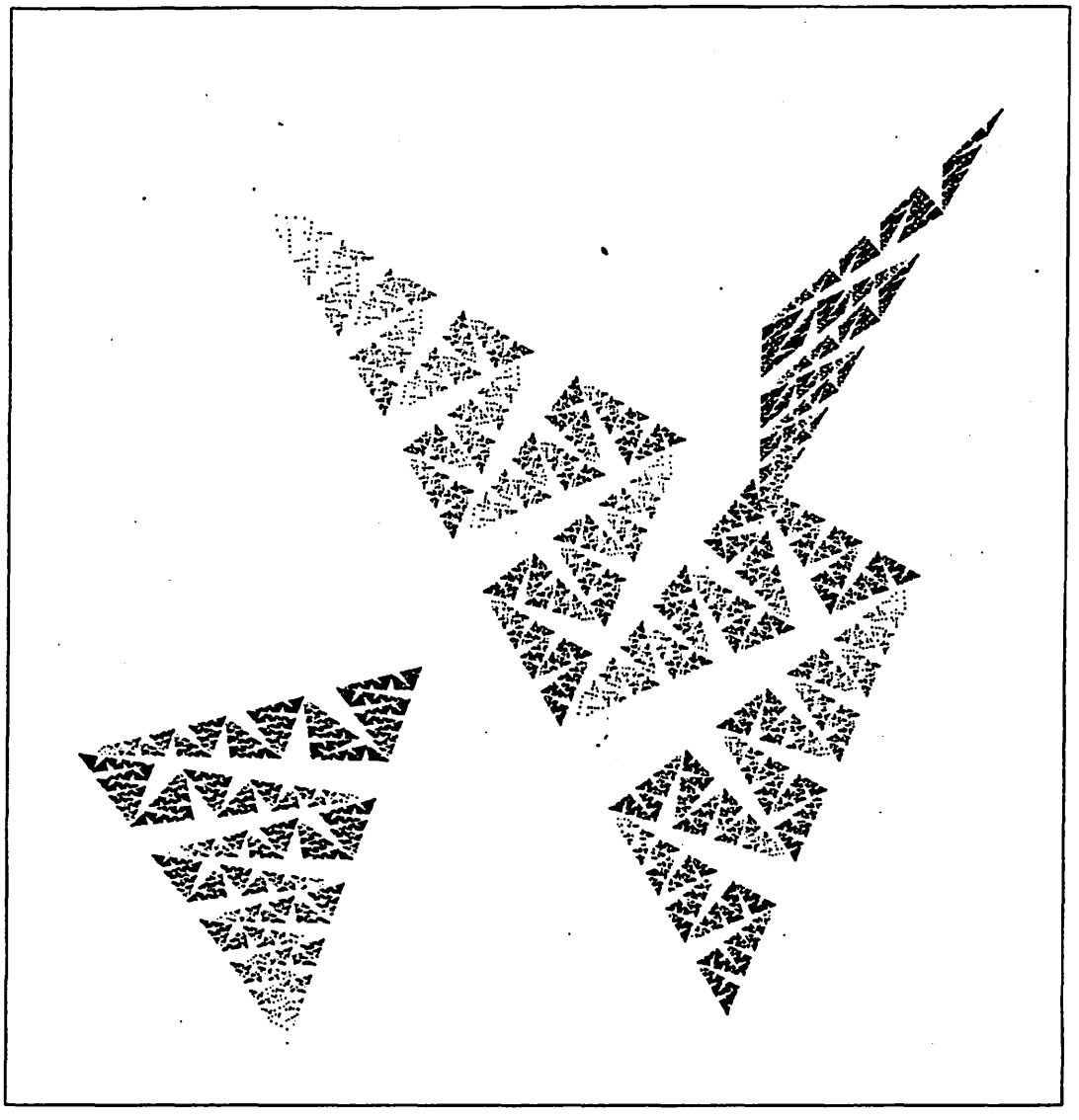
Guidance of library program use
(Numerical calculation: NUMPAC VOL. 3)

11. Table function 12. Orthogonal polynomial 13. Special functions

14. Bessel function and related function 15. Acceleration of sequence and series

16. Linear programming 17. Special data processing

18. Figure display application program 19. Others



29/9/92

Nagoya University Computer Center

(Supervision: Ichizo Ninomiya)

I. NUMPAC routine

Library programs of NUMPAC are roughly divided into two cathegories, ie., function subprograms and subroutine subprograms. There are some general rules for each of them and the rules are used in this manual for simple description. Please read the following explanations carefully before using NUMPAC.

(I) Function subprogram

(1) Function name and type

The function name of the real type follows the rule of the implicit type specification of FORTRAN.

Example : BJ0, ACND

The function name of the double precision real type consists of the function name of the corresponding real type with adding D to the head of it. The function name of the quadruple precision real number type (if exists) consists of the function name of the corresponding real type with adding Q to the head of it. However, there are some exceptions.

Example : SINHP, DSINHP, QSINHP

Example of exception : ALOG1, DLOG1, QLOG1

It is severely observed that the function name for double precision begins with D and that for quadruple precision begins with Q. Note that the function name should be declared with a suitable type in each program unit referring to the function.

```
Example : DOUBLE PRECISION DCOSHP, DJ1
          REAL*8 DCELI1, DCELI2
          REAL*16 QSINHP, QASINH
```

Because the function name of double precision always begins with D and that of quadruple precision with Q, it is convenient to use the IMPLICIT statement considering other variables.

```
Example : IMPLICIT REAL*8(D)
          IMPLICIT REAL*8(A-H, O-Z)
```

In this way, you need not declare the function name, separately.

(2) Accuracy of function value

Function routines are created aiming at the accuracy of full working precision as a rule. However, this cannot be achieved completely because of fundamental or technical difficulty [1].

4

Especially, it is not achieved for functions of two variables and functions of complex variable.

(3) Limit of argument

(a) The domain is limited.

Example : ALOG1

This function calculates $log(1+x)$. Therefore, $x>-1$ should be satisfied.

(b) The singular point exists.

Example : TANHP

This function calculates $\tan \pi x/2$. Therefore, an odd integer x is a sungularity.

(c) The function value overflows.

Example : BI0

This function is for modified Bessel function $I_0(x)$, and for big x, e^x is calculated referring to standard function EXP. Therefore, overflow limit $252 log_e 2 \fallingdotseq 174.673$ of EXP is the upper bound of the argument of this function.

(d) The function value becomes meaningless.

Example : BJ0

This function is for Bessel function $J_0(x)$, and standard functions SIN and COS are referred to for big x. Therefore, the argument limit $|x| \leqq 2^{18}\pi \fallingdotseq 8.23 \cdot 10^5$ of SIN and COS is the limit of the argument of this function.

There are many such examples. Note that the value $2^{18}\pi$ is not a sharp limit and that the number of significant digits for the function decreases gradually as approaching this limit even if within this limit.

When the function value underflows, it is set to 0 without special processing.

(4) Error processing

When the argument exceeds the limit, an message for the error is printed and the calculation is continued with the all function values set as 0. The message consists of the function name, the argument value, the function value (0) and the reason for the error.

Example : ALOG1 ERROR ARG=-0.2000000E+01 VAL=0.0 ARG.LT.-1

The error processing program counts the frequency of the errors and stops the calculation if the frequency exceeds a certain limit, considering the case that the calculation becomes meaningless when the error occurs one after another. Because all users do not want this, you can adopt or reject this processing including the print of the message. Subroutine FNERST is

provided for this purpose and you can use it in the following way.

CALL FNERST(IABORT, MSGPRT, LIMERR)

Argument	Type and kind	Attribute	Content
IABORT	Integer type	Input	IABORT=0 The calculation is not stopped. IABORT≠0 The calculation is stopped.
MSGPRT	Integer type	Input	MSGPRT=0 The message is not printed. MSGPRT≠0 The message is printed.
LIMERR	Integer type	Input	Upper bound of frequency of errors.

If this subroutine is not called, following values are set as a standard value.

IABORT=1, MSGPRT=1, LIMERR=10

(II) Subroutine subprogram

(1) Subroutine name and type

There is no meaning of the type in the head character of the subroutine name. Subroutines with the same purpose and the different type are distinguished by the ending character of the name. The principle is as follows.

Single precision : S Double precision : D Quadruple precision : Q	Complex number : C Double precision complex number : B Quadruple precision complex number : Z	Vector computer single precision : V Vector computer double precision : W Vector computer complex number : X Vector computer double precision complex number : Y

However, there are some exceptions.

Example	Example of exception
LEQLUS/D/Q/C/B RK4S/D/Q/C/B GJMNKS/D/Q	FFTR/FFTRD MINVSP/MINVDP

(2) Argument ··· The following four kinds are distinguished as an attribute of the argument.

Input	Users should set this data before calling the subroutine. As long as it is not especially noticed, the data is preserved as it is at the subroutine exit. This includes the case when the function name and the subroutine name are used as arguments. Note that those names should be declared with EXTERNAL.

6

Output	This data is created in the subroutine and is significant for the user.
Input/Output	Data is output in the same place as the input to save area. When input/output argument is a single variable, you should not specify a constant as a real argument. For instance, if LEQLUS is called with constant 1 specified in input/output argument and is ended normally, IND=0 is output, but all constants 1 are changed to 0.
Work area	It is an area necessary for calculation in a subroutine, and the content of the subroutine at exit is meaningless for users.

The type and attribute of the argument are explained for each subroutine group. The explanation is for single precision. For others, please read it with exchanging the type for the suitable one.

When a subroutine is called with an argument, but the argument is not used, the area for the argument need not be prepared, and anything can be written in that place. The same area can be allocated for the different arguments, only if it is pointed as it like SVDS. There is an example (FT235R) that special demand is requested for the argument.

It is requested for users to provide the function routine and the subroutine for the numerical integration routine and the routine for solving differential equations. In this case, the number, the type, and the order of the argument should be as specified. If parameters except a regulated argument are necessary, they are allocated in COMMON area to communicate with the main program. Refer to the explanation of an individual routine for the example.

---

1) Ichizo Ninomiya; "Current state, issues of mathematical software", information processing, Vol. 23 and pp. 109-117 (1982).

【 Opening source program to the public 】

The following source programs are published for users requesting them. Calculation can be requested directly, and the source list can be output or can be copied in the shared file. The copied program cannot be given to the third party without the permission of this center.

If you need to copy the source list in the card or the data set, please execute following procedures.

(1) Input the following command for TSS.

NLIBRARY ELM (library name) "DS (data set name)" "SLAVE(ON)"

When you need only the source list, you can omit DS and SLAVE. When SLAVE(ON) is specified, all slave routines of the program will be output.

(2) Execute the following job for BATCH.

//EXEC NLIBRARY, ELM=program names[, DS='data set names'][, SLAVE=ON]

You can have examples of the program usage with the following procedures.

(1) For TSS

EXAMPLE NAME (library name) [DS (data set name)]

(2) For BATCH

//EXEC EXAMPLE, NAME=program names[, DS='data set names']

Four kinds of manual listed below are prepared concerning library program.

Number	Manual title	Content
1	Library program and data list	All library programs and data which can be used in this center are listed. Additionally, "description format of the NUMPAC routine and notes on use", "How to choose the NUMPAC routine", and usage of error processing subroutine "FNERST" are described in this list.
2	Guidance to use library program (General volume : GENERAL VOL. 1)	This volume describes the general use of programs except NUMPAC, which can be used in this center.

8

3	Guidance to use library program (Numerical calculation : NUMPAC VOL. 1)	This volume describes how to use the following five kinds of programs. 1. Basic matrix operations 2. System of linear equations 3. Matrix inversion 4. Eigenvalue analysis 5. Polynomial equation and nonlinear equation
4	Guidance to use library program (Numerical calculation : NUMPAC VOL. 2)	This volume describes how to use the following five kinds of programs. 6. Interpolation, smoothing, and numerical differentiation and integration 7. Fourier analysis 8. Numerical quadrature 9. Ordinary differential equation 10. Elementary function
5	Guidance to use library program (Numerical calculation : NUMPAC VOL. 3)	This volume describes how to use the following nine kinds of programs. 11. Table functions 12. Orthogonal polynomial 13. Special functions 14. Bessel function and related function 15. Acceleration of convergence of sequences 16. Linear programming 17. Special data processing 18. Figure display application program 19. Others

All these manuals can be output by "MANUAL command". "PICKOUT command" is available if you need part of the usage of individual program.

For NUMPAC users

Please note the following and use NUMPAC effectively.

(1) The user has the responsibility for the result obtained by NUMPAC.

(2) When the trouble is found, please report it to the center program consultation corner (Extension 6530).

(3) Do not use NUMPAC in computer systems other than this center without permission.

(4) To publish the result obtained NUMPAC, the used program names (for instance, *** of NUMPAC) should be referred to.

This manual was translated using Fujitsu's machine translation system ATLAS.

10

II. Library and program itemized discussion

11. Table function

12

BERNO/DBERNO/QBERNO (Bernoulli's numbers)

Bernoulli Numbers

Programmed by	Ichizo Ninomiya, April 1981
Format	Function Language: FORTRAN; Size: 57, 57, and 57 lines respectively

(1) Outline

BERNO (DBERNO, QBERNO) calculates the Bernoulli's number B_{2n} for a positive integer n with single (double or quadruple) precision.

(2) Directions

1. BERNO(N), DBERNO(N), and QBERNO(N)

   N is an arbitrary expression of an integer type. DBERNO (QBERNO) requires the declaration of double (quadruple) precision.

2. Range of argument

   $1 \leqq N \leqq 48$

3. Error processing

   If an argument outside the range is given, an error message is printed, and the calculation is continued with the function value as 0. (See "FNERST.")

(3) Calculation method

The table precomputed with a sufficient precision is used.

(4) Note

1. There is the following relationship between the Bernoulli's number B_{2n} and Riemann Zeta function $\zeta(n)$.

$$B_{2n}=(-1)^{n+1}\frac{2(2n)!}{(2\pi)^{2n}}\zeta(2n)$$

2. If $n\to\infty$, $|B_{2n}|$ increases in about the same order as $(n/e\pi)^{2n}$.

Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., p.804 (1970).

(1987.07.07)

14

BETNO/DBETNO/QBETNO (Beta Numbers)

Beta Numbers

Programmed by	Ichizo Ninomiya, December 1987
Format	Function Language: FORTRAN; Size: 28, 55, and 84 lines respectively

(1) Outline

BETNO (DBETNO, QBETNO) calculates beta numbers $\beta(n)$ for a positive integer n with single (double or quadruple) precision.

$$\beta(n)=\sum_{k=0}^{\infty}\frac{(-1)^k}{(2k+1)^n}$$

(2) Directions

1. BETNO(N), DBETNO(N), and QBETNO(N)

   N is an arbitrary expression of an integer type. DBETNO (QBETNO) requires the declaration of double (quadruple) precision.

2. Range of argument

   $N \geqq 1$

3. Error processing

   If an argument outside the range is given, an error message is printed, and the calculation is continued with the function value as 0. (See "FNERST.")

(3) Calculation method

The table precomputed with a sufficient precision is used.

(4) Note

1. $\beta(n) \sim 1-3^{-n}$ for a sufficiently large number n.
2. The following expression gives the Euler number E_{2n}.

$$E_{2n}=(-1)^n\frac{2\cdot(2n)!}{(\pi/2)^{2n+1}}\beta(2n+1)$$

Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., p.807 (1970).

(1987.07.08) (1988.01.11)

16

EULNO/DEULNO/QEULNO (Euler Number)

Euler Number

Programmed by	Yasuyo Hatano and Kazuo Hatano, March 1983
Format	Function Language; FORTRAN

(1) Outline

EULNO (DEULNO, QEULNO) calculates the Euler number E2n for a positive integer n with single (double) precision.

(2) Directions

1. EULNO(N), DEULNO(N), QEULNO(N)

   N is an arbitrary expression of the integer type. DEULNO (QEULNO) requires the declaration of double (quadruple) precision.

2. Range of argument

   $1 \leqq N \leqq 31$

3. Error processing

   If an argument outside the range is given, an error message is printed, and the calculation is continued with the function value as 0.

(See FNERST.)

(3) Calculation method

The table calculated with sufficient precision in advance is used.

(4) Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., p.804 (1970).

(1987.08.07)

FCTRL/DFCTRL/QFCTRL, FFCTR/DFFCTR/QFFCTR, HFCTR/DHFCTR/QHFCTR (Factorials)

Factorials

Programmed by	Ichizo Ninomiya; December 1987, January 1980
Format	Function Language; FORTRAN Size; 25, 41, 66, 36, 62, 106, 67, 68, and 68 lines respectively

(1) Outline

FCTRL (DFCTRL, QFCTRL), FFCTR (DFFCTR, QFFCTR), and HFCTR (DHFCTR, QHFCTR) calculate factorial $n!$, double factorial $n!!$, and $\Gamma(n+1/2)/\Gamma(1/2)$ respectively, with single (double, quadruple) precision, for an integer n.

Where,

$$n! = \begin{cases} 1 & n=0 \\ n(n-1)\cdots\cdots 2\cdot 1 & n>0 \end{cases}$$

$$n!! = \begin{cases} 1 & n=0 \\ n(n-2)\cdots\cdots 4\cdot 2 & n\geqq 2 \quad \text{(even number)} \\ n(n-2)\cdots\cdots 3\cdot 1 & n\geqq 1 \quad \text{(odd number)} \end{cases}$$

$$\Gamma(n+1/2)/\Gamma(1/2) = \begin{cases} 0 & n=0 \\ 1/2\cdot 3/2\cdots(n-1/2) & n\geqq 1 \end{cases}$$

(2) Directions

1. FCTRL(N), DFCTRL(N), QFCTRL(N), FFCTR(N), DFFCTR(N), QFFCTR(N), HFCTR(N), DHFCTR(N), and QHFCTR(N)

   N is an arbitrary integer-type expression. The double (quadruple) precision function name needs to be declared as double (quadruple) precision.

2. Range of argument

   Factorial: $0\leqq N\leqq 57$. Double factorial: $0\leqq N\leqq 97$. HFCTR etc.: $0\leqq N\leqq 57$.

18

3. Error processing

If the specified argument is outside the range, an error message is printed but calculation continues with the function value assumed to be 0. (See PNERST.)

(3) Calculation method

A numerical table precomputed with sufficient precision is used.

(1987. 07. 03) (1988. 02. 15)

GAMCO/DGAMCO/QGAMCO (Coefficient of Tayler Series For $1/\Gamma(x)$)

Coefficients of Taylor Series for $1/\Gamma(x)$

Programmed by	Ichizo Ninomiya, April 1981
Format	Function Language: FORTRAN; Size: 91, 91, and 91 lines respectively

(1) Outline

GAMCO (DGAMCO, QGAMCO) calculates the value of coefficients of Tayler series C_n of $1/\Gamma(x)$ for positive integers n with single (double or quadruple) precision, where

$$1/\Gamma(x)=\sum_{n=1}^{\infty}C_n x^n$$

(2) Directions

1. GAMCO(N), DGAMCO(N), and QGAMCO(N)

   N is an arbitrary expression of an integer type. DGAMCO (QGAMCO) requires the declaration of double (quadruple) precision.

2. Range of argument

   $N \geqq 1$

3. Error processing

   If an argument outside the range is given, an error message is printed, and the calculation is continued with the function value as 0. (See "FNERST.")

(3) Calculation method

The table precomputed with a sufficient precision is used.

(4) Note

1. $C_n=0$ is assumed because the function underflow at $n \geqq 80$.
2. C_2 is the Euler's constant γ.

(1987.07.13)

20

HARMS/DHARMS/QHARMS (Partial sum of harmonic series)

Partial Sum of the Harmonic Series

Programmed by	Ichizo Ninomiya; December 1987
Format	Function Language; FORTRAN Size; 118, 122, and 134 lines respectively

(1) Outline

HARMS (DHARMS, QHARMS) calculates partial sums

$$\varphi(n)=\sum_{k=1}^{n}1/k$$

up to term n of the harmonic series, with single (double, quadruple) precision, for an integer n.

(2) Directions

1. HARMS(N), DHARMS(N), QHARMS(N)

   N is an arbitrary integer-type expression. DHARMS (QHARMS) needs to be declared for double (quadruple) precision.

2. Range of argument

   $N \geqq 0$

3. Error processing

   If the specified argument is outside the range, an error message is printed but calculation continues with the function value assumed to be 0. (See FNERST.)

(3) Calculation method

(1) In case of $n<0$, an error results.

(2) In case of $n=0$, $\varphi(n)=0$.

(3) In case of $1 \leq n \leq 100$, $\varphi(n)$ is read from the numerical table calculated beforehand.

(4) In case of $n>100$, the following asymptotic expansion is used:

$$\varphi(n)=\gamma+ln(n+1)-\frac{1}{2(n+1)}-\sum_{k=1}^{\infty}\frac{B_{2k}}{2k\cdot(n+1)^{2k}}$$

where γ is an Euler's constant.

Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., p.259 (1970).

(1987.08.05) (1988.02.15)

22

ZETNO/DZETNO/QZETNO (Riemann Zeta function)

Riemann Zeta Function

Programmed by	Ichizo Ninomiya; April 1981
Format	Function Language; FORTRAN Size; 42, 62, and 162 lines respectively

(1) Outline

ZETNO, DZETNO, and QZETNO each calculate Riemann Zeta function $\zeta(n)$ for a positive integer n with single, double, or quadruple precision.

$$\zeta(n)=\sum_{k=1}^{\infty}\frac{1}{k^n}$$

(2) Directions

1. ZETNO(N), DZETNO(N), and QZETNO(N)

   N is an arbitrary integer-type expression. DZETNO and QZETNO needs to be declared as double and quadruple precision respectively.

2. Range of argument

   $N\geqq 1$

3. Error processing

   If the specified argument is outside the range, an error message is printed but calculation continues with the function value assumed to be 0. (See FNERST.)

(3) Calculation method

A numerical table precomputed with sufficient precision is used.

(4) Notes

1. Since original $\zeta(1)$ is not defined for n=1, the Euler's constant is output instead:

$$\gamma=\lim_{k\to\infty}\left(\sum_{i=1}^{k}\frac{1}{i^n}-\log k\right)$$

2. $\zeta(n)\sim 1+2^{-n}$ for sufficiently large n.

3. The Bernoulli's constant B_{2n} is given by the following expression:

$$B_{2n}=(-1)^{n+1}\frac{2(2n)!}{(2\pi)^{2n}}\zeta(2n)$$

Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., p.804 (1970).

(1987.07.08)

24

12. Orthogonal polynomial

PLEGE/DPLEGE, PLEGA/DPLEGA, PLEGN/DPLEGN, PCHB1/DPCHB1, PCHB2/DPCHB2, PLAGU/DPLAGU, PLAGG/DPLAGG, and PHERM/DPHERM

(Orthogonal polynomials)

Orthogonal Polynomials

Programmed by	Ichizo Ninomiya and Yasuyo Hatano: March 1984, revised in December 1987
Format	Function Language; FORTRAN77 Size; 130 lines or less each

(1) Outline

When order n and argument x (auxiliary variables m and a) are given, each function calculates corresponding orthogonal polynomial as follows:

PLEGE(DPLEGE) ··· Legendre polynomial $P_n(X)$

PLEGA(DPLEGA) ··· Adjoint Legendre function $P_n^m(X)$

PLEGN(DPLEGN) ··· Normalized adjoint Legendre function $\overline{P}_n^m(X)$

PCHB1(DPCHB1) ··· Chebyshev polynomial of the first kind $T_n(X)$

PCHB2(DPCHB2) ··· Chebyshev polynomial of the second kind $U_n(X)$

PLAGU(DPLAGU) ··· Laguerre polynomial $L_n(X)$

PLAGG(DPLAGG) ··· Generalized Laguerre polynomial $L_n^{(a)}(X)$

PHERM(DPHERM) ··· Hermite polynomial $H_n(X)$

(2) Directions

1. PLEGE(N, X), DPLEGE(N, DX)

   PLEGA(N, M, X), DPLEGA(N, M, DX)

   PLEGN(N, M, X), DPLEGN(N, M, DX)

   PCHB1(N, X), DPCHB1(N, DX)

   PCHB2(N, X), DPCHB2(N, DX)

   PLAGU(N, X), DPLAGU(N, DX)

   PLAGG(N, A, X), DPLAGG(N, DA, DX)

   PHERM(N, X), DPHERM(N, DX)

N and M are arbitrary integer type expressions. X (DX) and A (DA) are arbitrary single (double)

precision expressions. A function whose name begins with D is used for double precision calculation. The function name needs to be declared as double precision.

2. Range of argument

$0 \leqq N$, $0 \leqq M \leqq N$, $-1 < A$ (DA)

PLEGA (DPLEGA): $M \leqq 49$

PLEGN (DPLEGN): $M \leqq 100$

3. Error processing

If the argument is outside the range, an error results with an error message printed, but calculation continues with the function value assumed to be 0. (Refer to FNERST.)

(3) Calculation method

Each function calculates a 3-term recurrence formula in the forward direction.

1. PLEGE (DPLEGE)

$$P_k(x)=\frac{1}{k}((2k-1)xP_{k-1}(x)-(k-1)P_{k-2}(x))$$

2. PLEGA (DPLEGA)

$$P_k^m(x)=\frac{1}{k-m}((2k-1)xP_{k-1}^m(x)-(k+m-1)P_{k-2}^m(x))$$

3. PLEGN (DPLEGN)

$$\rho_k^m(x)=((2k+1)/(k^2-m^2))^{1/2}$$

$$\times((2k-1)^{1/2}\ x\rho_{k-1}^m(x)-(((k-1)^2-m^2)/(2k-3))^{1/2}\rho_{k-2}^m(x))$$

4. PCHB1 (DPCHB1)

$$T_k(x)=2xT_{k-1}(x)-T_{k-2}(x)$$

5. PCHB2 (DPCHB2)

$$U_k(x)=2xU_{k-1}(x)-U_{k-2}(x)$$

6. PLAGU (DPLAGU)

$$L_k(x)=\frac{1}{k}((2k-1-x)L_{k-1}(x)-(k-1)L_{k-2}(x))$$

7. PLAGG (DPLAGG)

$$L_k^{(a)}(x)=\frac{1}{k}((2k-1+a-x)L_{k-1}^{(a)}(x)-(k-1+a)L_{k-2}^{(a)}(x))$$

8. PHERM (DPHERM)

$$H_k(x)=2xH_{k-1}(x)-2(k-1)H_{k-2}(x)$$

(4) Note

$$P_n^m(x)=|1-x^2|^{m/2}\ \frac{d^mP_n(x)}{dx^m},$$

$$\rho_n^m(x)=(-1)^m((2n+1)\cdot(n-m)!/(2\cdot(n+m)!))^{1/2}P_n^m(x).$$

Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., pp. 771-802 (1970).

(1989. 01. 25)

28

13. Special functions

ABRM0/DABRM0,ABRM1/DABRM1,ABRM2/DABRM2

(Abramowitz functions of the order 0, 1, and 2))

Abramowitz Functions of the Order 0, 1, and 2

Programmed by	Ichizo Ninomiya; December 1986
Format	Function Language; FORTRAN77 Size; 53, 82, 53, 76, 54, and 80 lines respectively

(1) Outline

ABRM0 (DABRM0), ABRM1 (DABRM1), and ABRM2 (DABRM2) calculate $f_0(x)$, $f_1(x)$, and $f_2(x)$ respectively for a single (double) precision real x with single (double) precision.

$f_m(x)$ by the Abramowitz function and is defined as follows:

$$f_m(x)=\int_0^\infty t^m e^{-(t^2+x/t)}dt \quad (m\geqq 0)$$

(2) Directions

1. ABRM0(X), ABRM1(X), ABRM2(X), DABRM0(D), DABRM1(D), DABRM2(D). X(D) is an arbitrary single (double) precision real type expression. DABRM0, DABRM1, and DABRM2 each needs to be declared as double precision.

2. Range of argument $0\leqq X$, $0\leqq D$

3. Error processing

If the argument is outside of the range, an error message is printed but calculation continues with the function value assumed to be 0. (See FNERST.)

(3) Calculation method

1. When $1.0\leqq x<1$, polynomials A0, A1, A2, and rational functions B0, B1, and B2 are used to calculate:

$$f_0(x)=\log x\cdot x\cdot A_0(x^2)+B_0(1-x)$$

$$f_1(x)=\log x\cdot x^2\cdot A_1(x^2)+B_1(1-x)$$

$$f_2(x)=\log x\cdot x^3\cdot A_2(x^2)+B_2(1-x)$$

2. When $1\leqq x\leqq 2$, rational approximations C0, C1, and C2 are used to calculate:

$$f_0(x)=C_0(x-1),\ f_1(x)=C_1(1-x),\ f_2(x)=C_2(1-x)$$

3. When x>2,

$$f_0(x)=e^{-v}D_0(1/v),\ f_1(x)=e^{-v}(x/2)^{1/3}D_1(1/v),\ f_2(x)=e^{-v}(x/2)^{2/3}D_2(1/v)$$

is calculated by polynomial approximations D0, D1, and D2.

Where, $v=3(x/2)^{2/3}$.

(4) Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., p.1001 (1970).

(1987.08.06)

ABRMW/DABRMW (Abramowitz functions of integral order)

Abramowitz Functions of the Integral Order

Programmed by	Ichizo Ninomiya; December 1986
Format	Function Language; FORTRAN77 Size; 36 and 37 lines respectively

(1) Outline

ABRAMW (DABRMW) calculates $f_n(x)$, where n is an integer and x is a single (double) precision real, with single (double) precision.

Where

$$f_m(x)=\int_0^\infty t^m e^{-(t^2+x/t)}dt \qquad (m\geqq 0)$$

(2) Directions

1. ABRMW(N, X), DABRMW(N, D)

   X (D) is an arbitrary single (double) precision expression.

   DABRMW needs be declared as a double precision real type.

2. Range of argument

   0<N, 0<X(D)

3. Error processing

   If the value given to the argument is outside of the range, an error message is output but operation continues with the function value assumed to be 0. (See FNERST.)

(3) Calculation method

Recurrence formula

$$f_m(x)=\frac{m-1}{2}f_{m-2}(x)+\frac{x}{2}f_{m-3}(x)$$

is used.

$f_0(x), f_1(x), f_2(x)$ are calculated by calling function routines ABRM0 (DABRM0), ABRM1 (DABRM1), and ABRM2 (DABRM2) respectively.

32

(4) Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., p.1001 (1970).

(1987.08.07)

ACND/DACND, ACNDC/DACNDC (The Inverses of the Cumulative Normal Distribution Function and its Complement)

The Inverses of the Cumulative Normal Distribution Function and its Complement

Programmed by	Ichizo Ninomiya, revised in April 1977, April 1981
Format	Function Language: FORTRAN; Size: 9, 10, 9, and 10 lines respectively

(1) Outline

ACND (DACND) calculates $\Phi^{-1}(x)$ for a single (double) precision real numbers x with single (double) precision, and ACNDC (DACNDC) calculates $\Psi^{-1}(x)$ similarly,

where,

$$\Phi(x) = \frac{1}{\sqrt{2\pi}}\int_0^x e^{-t^2/2}dt, \quad \Psi(x) = \frac{1}{\sqrt{2\pi}}\int_x^\infty e^{-t^2/2}dt = \frac{1}{2}-\Phi(x)$$

and $\Phi^{-1}(x), \Psi^{-1}(x)$ are the inverse functions of $\Phi(x), \Psi(x)$ respectively.

(2) Directions

1. ACND(X), DACND(D), ACNDC(X), and DACNDC(D)

   X (D) is an arbitrary expression of a single (double) precision real number type. DACND and DACNDC require the declaration of double precision.

2. Range of argument

   (1) ACND and DACND

   $$-\frac{1}{2} < X < \frac{1}{2}, \quad -\frac{1}{2} < D < \frac{1}{2}$$

   (2) ACNDC and DACNDC

   0<X<1, 0<D<1

3. Error processing

   If an argument outside the range is given, an error message is printed, and the calculation is continued with the function value as 0. (See "FNERST.")

34

(3) Calculation method

1. ACND (DACND)

   AERF (DAERF) is called using the relationship $\Phi^{-1}(x)=\sqrt{2}erf^{-1}(2x)$.

2. ACNDC (DACNDC)

   AERFC (DAERFC) is called using the relationship $\Psi^{-1}(x)=\sqrt{2}erfc^{-1}(2x)$.

(1987.07.31)

AERF/DAERF, AERFC/DAERFC (The Inverse of the Error Function and Its Complement)

The Inverse of the Error Function and Its Complement

Programmed by	Ichizo Ninomiya, April 1977
Format	Function Language: FORTRAN; Size: 21, 31, 37, and 63 lines respectively

(1) Outline

AERF (DAERF) calculates $erf^{-1}x$ for a single (double) precision numbers x with single (double) precision, and AERFC (DAERFC) calculates $erfc^{-1}x$ similarly,

where,

$$erf\ x = \frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}dt$$

$$erfc\ x = \frac{2}{\sqrt{\pi}}\int_x^\infty e^{-t^2}dt$$

$$erf\ x + erfc\ x = 1$$

. The functions above are the inverse functions of $erf^{-1}x$, $erfc^{-1}x$ respectively.

(2) Directions

1. AERF(X), DAERF(D), AERFC(X), and DAERFC(D)

   X (D) is an arbitrary expression of a single (double) precision real number type. DAERF and DAERFC require the declaration of double precision.

2. Range of argument

   -1<X<1 and -1<D<1 for the inverse of error functions.

   0<X<2 and 0<D<2 for the inverse of error functions and their complement.

3. Error processing

   If an argument outside the range is given, an error message is printed, and the calculation is continued with the function value as 0. (See "FNERST.")

(3) Calculation method

1. AERF (DAERF)

(1) If $|x| \geqq 1$, an error results.

(2) If $|x| \leqq 0.75$, $erf^{-1}x = xR_1(t)$ is calculated using the optimal, rational approximation R_1. Where, $t = (0.75+x) \cdot (0.75-x)$.

(3) If $|x| > 0.75$, $erf^{-1}x = erfc^{-1}(1-x)$ is calculated calling AERFC (DAERFC).

2. AERFC (DAERFC)

(1) If $x \leqq 0$ or $x \geqq 2$, an error results.

(2) If $0.25 \leqq x \leqq 1.75$, that is, if $|1-x| \leqq 0.75$, $erfc^{-1}x = erf^{-1}(1-x)$ is calculated calling AERF (DAERF).

(3) If $0.75 < |1-x| \leqq 0.9375$, $erfc^{-1}x = sign(1-x) \cdot (1-u) \cdot R_2(t)$ is calculated using the optimal, rational approximation R_2. Where, $u = min(2-x, x)$, $t = (u-0.0625) \cdot (1.9375-u)$.

(4) If $|1-x| > 0.9375$, $erfc^{-1}x = t \cdot R_3(t)$ is calculated using the optimal, rational approximation R_3.

Where, $t = \sqrt{-log(min(2-x, x))}$.

Bibliography

1) A. J. Strecok; "On the Calculation of the Inverse of the Error Function", Math. Comp., Vol. 22 (1968).

(1987. 07. 15)

AICGAM/DICGAM (Incomplete gamma functions)

Incomplete Gamma Functions

Programmed by	Toshio Yoshida: June 1985
Format	Function language; FORTRAN Size; 309 and 918 lines respectively

(1) Outline

AIGAM (DIGAM) calculates

$$\Gamma(\nu,x)=\int_x^\infty e^{-u}u^{\nu-1}du$$

for single (double) precision real numbers ν and x in single (double) precision.

(2) Directions

1. AICGAM(V, X), DICGAM(W, D)

V and W correspond to ν, and X and D correspond to x.

V and X (W and D) are arbitrary single (double) precision real type expressions. DICGAM requires declaration of double precision.

2. Range of argument

$V\geqq 0$, $X\geqq 0$ ($W\geqq 0$, $D\geqq 0$) excluding V=X=0 (W=D=0)

3. Error processing

If an argument outside the range is given, an error message is output and calculation is continued assuming the function value to be 0. (See the description of FNERST.)

(3) Calculation method

The calculation method of $\Gamma(\nu,x)$ differs depending on the size of x. When x is small and $0\leqq\nu<1$, the function value is calculated from the modification of the following expression in order to avoid loss of significant digits near $\nu\fallingdotseq 0$:

38

$$\Gamma(\nu,x)=\Gamma(\nu)-\int_0^x e^{-u}u^{\nu-1}du$$

$$=\Gamma(\nu)e^{-x}\left\{e^{x}-x^{\nu}\sum_{k=0}^{\infty}\frac{x^k}{\Gamma(k+1+\nu)}\right\}$$

When $1\leqq\nu\leqq2$, the function value is calculated in the same way. When $\nu>2$ for $\Gamma(\nu,x)$, the function is calculated by using the following recurrence relation:

$$\Gamma(\nu+1,x)=x^{\nu}e^{-x}+\nu\Gamma(\nu,x)$$

For details, refer to bibliography 1).

When x is large, calculation is done by the approximation of $f_\nu(1/x)$ in the following form:

$$\Gamma(\nu,x)=e^{-x}x^{\nu-1}f_\nu(\tfrac{1}{x})$$

Note that the approximation is obtained by applying the τ method to the following differential equation that satisfies $f_\nu(t)$:

$$t^2f_\nu''(t)+\left\{(3-\nu)t+1\right\}f_\nu'(t)+(1-\nu)f_\nu(t)=0$$

For details, refer to bibliography 2).

Bibliography

1) Toshio Yoshida and Ichizo Ninomiya: "Computation of incomplete gamma function $\Gamma(\nu,x)$ for small argument x", Transactions of Information Processing Soc. of Japan, Vol. 23, No. 5, pp. 522-528 (1982).

2) Toshio Yoshida and Ichizo Ninomiya: "Computation of incomplete gamma function $\Gamma(\nu,x)$ for large argument x", Transactions of Information Processing Soc. of Japan, Vol. 25, No. 2, pp. 306-312 (1984).

BETIC/DBETIC (Incomplete Beta Integral)

Title Incomplete Beta Integral

Programmed by	Ichizo Ninomiya, April 1983
Format	Function Language: FORTRAN; Size 34 and 35 lines respectively

(1) Outline

BETIC (DBETIC) calculates the incomplete beta integral

$$B(x,a,b)=\int_0^x t^{a-1}(1-t)^{b-1}dt$$

for single (double) precision real numbers x, a, and b with single (double) precision.

(2) Directions

1. BETIC(X, A, B)

   DBETIC (D, P, Q)

A, B, and C (D, P, Q) are arbitrary expressions of single (double) precision number type. DBETIC requires the declaration of double precision.

2. Range of argument

   $0\leq X\leq 1$, $0\leq A$, $0\leq B$, $A+B\leq 56$,

   $0\leq D\leq 1$, $0\leq P$, $0\leq Q$, $P+Q\leq 56$

3. Error processing

   If an argument outside the range is given, an error message is printed, and the calculation is continued with the function value as 0.

(3) Calculation method

1. If $0\leq x\leq 0.575$, the Taylor series

$$B(x,a,b)=\sum_{k=0}^{\infty}\frac{(-1)^k(b-1)(b-2)\cdots(b-k)x^{k+1}}{k!(a+k)}$$

is calculated.

2. If $0.575<5\leq 1$, the calculation of B(x, a, b) is reduced to that of B(1-x, b, a) using the

relationship of $B(x, a, b) = B(a, b) - B(1-x, b, a)$,

where, a complete Beta integral is calculated as.

$$B(a, b) = \begin{cases} \Gamma(a) - \Gamma(b) / \Gamma(a+b) & (a+b \neq 1) \\ \pi / \sin a\pi & (a+b=1) \end{cases}$$

(1987. 08. 11)

42

BLAS/DBLAS/BLASP/DBLASP (Solution of Blasius equation and its derivatives)

Solution of Blasius Equation and its Derivative

Programmed by	Ichizo Ninomiya; March 1987
Format	Function Language; FORTRAN77 Size; 31, 56, 32, and 55 lines respectively

(1) Outline

BLAS (DBLAS) calculates $f(x)$ for a single (double) precision real x, with single (double) precision.

BLASP (DBLASP) calculates $f'(x)$ for a single (double) precision real x, with single (double) precision.

Where, f satisfies $2f'''+ff''=0$, $f(0)=f'(0)=0$, $f'(\infty)=1$.

(2) Directions

1. BLAS(X), BLASP(X), DBLAS(D), DBLASP(D). X (D) is an arbitrary single (double) precision real-type expression. DBLAS and DBLASP need to be declared as double precision.

2. Range of argument X≧0 and D≧0.

3. Error processing

If the argument is outside of the range, an error message is printed but calculation continues with the function value assumed to be 0.

(See FNERST.)

(3) Calculation method

1. When 0≦x≦3.7490234375, polynomial approximation A and B are used to calculate:

$f(x)=x^2\cdot A(t)$,

$f'(x)=x\cdot B(t)$,

$t=\alpha x^3/2$, $\alpha=0.33205733621519629894$.

2. When 3.7490234375<X≦10(14), rational approximation C and D are used to calculate:

$f(x)=x-\beta+e^{-z^2}\cdot C(u)/z^2$,

$f'(x)=1+e^{-z^2}\cdot D(u)/z$,

$z=(x-\beta)/2$,

$u=3.7490234375/z$,

$\beta=1.7207876575205028196$.

3. When x>10(14), then

$f(x)=x-\beta$

$f'(x)=1$.

(4) Summary.

In the range of $3.7490234375<x\leqq 14$, precision of DBLASP is indicated by a relative error of about 10^{-15}.

(1987.08.07)

44

CELI1/DCELI1/QCELI1 and CELI2/DCELI2/QCELI2 (Complete Elliptic Integrals of the First and the Second Kind)

Complete Elliptic Integrals of the First and the Second Kind

Programmed by	Ichizo Ninomiya, April 1977
Format	Function Language: FORTRAN; Size: 36, 61, 40, and 64 lines respectively

(1) Outline

CELI1 (DCELI1, QCELI1) calculates $K(x)$ for a single (double) precision real numbers x with single (double) precision, and CELI2 (DCELI2, QCELI2) calculates $E(x)$ similarly,

where,

$$K(x)=\int_0^{\frac{\pi}{2}}\frac{d\theta}{\sqrt{1-x\sin^2\theta}},\qquad E(x)=\int_0^{\frac{\pi}{2}}\sqrt{1-x\sin^2\theta}\ d\theta$$

That is, x is a value that is usually written as k^2.

(2) Directions

1. CELI1(X), DCELI1(D), QCELI1(Q), CELI2(X), DCELI2(D), and QCELI2(Q)

   X (D) is an arbitrary expression of a single (double) precision real number type. DCELI1 and DCELI2 require the declaration of double precision.

2. Range of argument

   $0\leqq X<1$ and $0\leqq D<1$ for the complete elliptic integral of the first kind.

   $0\leqq X\leqq 1$ and $0\leqq D\leqq 1$ for the complete elliptic integral of the second kind.

3. Error processing

   If an argument outside the range is given, an error message is printed, and calculation is continued with the function value as 0. (See "FNERST.")

(3) Calculation method

1. CEL11 (DCEL11, QCEL11)

(1) If $x<0$ or $x \geqq 1$, an error results.

(2) If $0 \leqq x<1/2$, a rational approximation function of $y=1-x$ is used.

(3) If $1/2 \leqq x<1$, rational approximations P and Q are used to calculate $K(x)=P(y)-Q(y)\log y$, where $y=1-x$.

2. CEL12 (DCEL12, QCEL12)

(1) If $x<0$ or $x 1$, an error results.

(2) If $x=1$, $E(x)=1$.

(3) If $0 \leqq x<1/2$, a rational approximation function of $y=1-x$ is used.

(4) If $1/2 \leqq x<1$, rational approximations P and Q are used to calculate $E(x)=P(y)-Q(y)\log y$, where $y=1-x$.

(4) Note

Note that not the modulus k but k^2 is put in the argument x.

Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., p.587 (1970).

(1987.07.31)

46

CGAMMA/CDGAMA/CQGAMA (Gamma function for complex arguments)

Gamma Function for Complex Arguments

Programmed by	Ichizo Ninomiya; April 1977; Revised in December 1986
Format	Function Language; FORTRAN Size; 76, 77, and 120 lines respectively

(1) Outline

CGAMMA (CDGAMA, CQGAMA) calculates $\Gamma(z)$ for single (double, quadruple) precision complex number z, as a single (double, quadruple) precision complex number.

(2) Directions

1. CGAMMA(C), CDGAMA(B), CQGAMA(Z)

   C (B, Z) is an arbitrary expression of the single (double, quadruple) precision type. CGAMMA (CDGAMA, CQGAMA) needs be declared to be a single (double, quadruple) precision complex.

2. Range of argument

   The range of argument is a complex plane from which 0, negative integers, and an area where gamma function values overflow are excluded.

3. Error processing

   If the value given to the argument is outside of the range, an error message is printed but operation continues with the function value assumed to be 0. (See FNERST.)

(3) Calculation method

Suppose the argument is z=x+iy.

1. When z is 0 or a negative integer, an error results.
2. When $|x|>56$ or $|y|>56$, an error results.
3. When $|z|\leqq 1$ or $|x|\leqq 0.5$ and $|y|\leqq 1.0$, $1/\Gamma(z)$ is calculated by the Taylor series.
4. When $x<0$, it is converted into the case of $x>0$ using the inversion formula

$\Gamma(z)=\pi/(\Gamma(1-z)\cdot\sin\pi z)$.

5. When $x^2+y^2>16(32,144)$, $\mathrm{Log}\,\Gamma(z)$ is calculated by the partial sum of asymptotic series

$$\log\Gamma(z)=(z-1/2)\cdot\log z+\log\sqrt{2\pi}-z+\sum_{n=1}^{\infty}C_n/z^{2n-1}.$$

6. When $|y|\leq 1.0$ but the argument is outside of (1) and (3), the asymptotic expansion $\Gamma(z)=\Gamma(z+1)/z$ is repeated as many times as needed to reduce to the case of 3.

7. When $|y|>1.0$ and $x^2+y^2\leq 16(32,144)$, the asymptotic expansion $\Gamma(z+1)=z\,\Gamma(z)$ is repeated as many times as needed to reduce to the case of 3.

(4) Note

1. CDGAMA and CQGAMA can be replaced by DCGAMA and QCGAMA respectively.

(1987.08.07)

48

CLASN/DCLASN (Clausen's Integral)

Clausen's Integral

Programmed by	Ichizo Ninomiya, February 1985
Format	Function Language; FORTRAN77; Size; 29 and 43 lines respectively

(1) Outline

CLASN (DCLASN) calculates the Clausen's integral f(x) for a single (double) precision real number x with single (double) precision. Where,

$$f(x)=-\int_0^{\pi x} \log\left(2\sin\frac{t}{2}\right)dt=\sum_{k=1}^{\infty}\frac{\sin k\pi x}{k^2}$$

(2) Directions

1. CLASN(X), DCLASN(D)

   X(D) is arbitrary an expression of single (double) precision real type. DCLASN requires the declaration of double precision.

2. Range of argument $|X|\leqq 1$ and $|D|\leqq 1$

3. Error processing

   If an argument outside the range is given, an message is printed, and the calculation is continued with the function value as 0. (See FNERST.)

(3) Calculation method

1. If $|x|>1$, an error results.
2. If $|x|\leqq 1/2$, $f(x)=\pi x(A(x^2)-\log(\pi x))$ is calculated with the optimal approximation A.
3. If $1/2<|x|\leqq 1$, $f(x)=x\cdot B(x^2)$ is calculated with the optimal approximation B.

Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., p.1005 (1970)

(1989.01.25)

CND/DCND, CNDC/DCNDC (The Cumulative Normal Distribution Function and its Complement)

The Cumulative Normal Distribution Function and its Complement

Programmed by	Ichizo Ninomiya, revised in April 1977, April 1981
Format	Function Language: FORTRAN; Size: 5, 6, 5, and 6 lines respectively

(1) Outline

CND (DCND) calculates $\Phi(x)$ for a single (double) precision real numbers x with single (double) precision, and CNDC (DCNDC) calculates $\Psi(x)$ similarly,

where,

$$\Phi(x) = \frac{1}{\sqrt{2\pi}}\int_0^x e^{-t^2/2}dt$$

$$\Psi(x) = \frac{1}{\sqrt{2\pi}}\int_x^\infty e^{-t^2/2}dt = \frac{1}{2} - \Phi(x)$$

(2) Directions

1. CND(X), DCND(D), CNDC(X), and DCNDC(D)

   X (D) is an arbitrary expression of a single (double) precision real number type. DCND and DCNDC require the declaration of double precision.

2. Range of argument

   There is no limit on arguments.

(3) Calculation method

1. CND (DCND)

   The elementary external function ERF (DERF) is called using the relationship

$\Phi(x)=erf(x/\sqrt{2})/2$.

2. CNDC (DCNDC)

   The elementary external function ERFC (DERFC) is called using the relationship

$\Psi(x)=erfc(x/\sqrt{2})/2$.

50

(1987.07.31)

DAWSN/DDAWSN (Dawson's Integral)

Dawson's Integral

Programmed by	Ichizo Ninomiya, February 1985
Format	Function Language: FORTRAN77; Size: 62 and 125 lines respectively

(1) Outline

DAWSN (DDAWSN) calculates the Dawson's integral f(x) for a single (double) precision real number x with single (double) precision. Where,

$$f(x)=e^{-x^2}\int_0^x e^{t^2}dt$$

(2) Directions

1. DAWSN(X), DDAWSN(D)

X(D) is an arbitrary expression of single (double) precision real type. DDAWSN requires the declaration of double precision.

2. Range of arguments: None.

(3) Calculation method

1. DAWSN

(1) If $|x|\leqq 4$, the optimal rational approximations A is used according to the value of |x| to calculate $f(x)=x\cdot A(x^2)$.

(2) If $|x|>4$, we calculate $f(x)=P((4/x)^2)/x$ with the Optimal polynomial approximation P.

2. DDAWSN

(1) If $|x|\leqq 6$, the optimal rational approximation formula B is used according to the value of |x| to calculate $f(x)=x\cdot B(x^2)$.

(2) If $|x|>6$, we calculate $f(x)=Q((6/x)^2)/x$ with the optimal polynomial approximation Q.

52

Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., pp.297-329 (1970)

(1989.01.25)

DEBYE/DDEBYE (Debye Function)

Debye Function

Programmed by	Ichizo Ninomiya, December 1987
Format	Function Language: FORTRAN77; Size: 33 and 49 lines respectively

(1) Outline

DEBYE(DDEBYE) calculates the Debye function $Db(x)$ for a single (double) precision real numbers x with single (double) precision.

Where,

$$Db(x)=\int_0^x \frac{t\ dt}{e^t-1}$$

(2) Directions

1. DEBYE(X) and DDEBYE(D)

X(D) is an arbitrary expression of single (double) precision.

DDEBYE requires the declaration of a double precision real type.

2. Range of argument

No limit.

(3) Calculation method

(1) If $x<-18.42(-41.45)$, $Db(x)=-\pi^2/6-x^2/2$.

(2) If $-18.42(-41.45)\leqq x<-2$, $Db(x)=-\pi^2/6-ln^2(-t)/2-t\cdot A(t)$ is calculated using an optimal polynomial approximation A.

Where, $t=1/(1-e^{-x})$

(3) If $|x|\leqq 2$, $Db(x)=x\cdot(1-x/4+x^2\cdot B(x^2))$ is calculated using an optimal polynomial approximation B.

(4) If $2<x\leqq 18.42(41.45)$, $Db(x)=\pi^2/6+t\cdot x+t(1+t/4+t^2\cdot B(t^2))$ is calculated. Where, $t=ln(1-e^{-x})$

54

(5) If $x>18.42(41.45)$, $Db(x)=\pi^2/6$.

(4) Note

There is the relationship $Db(x)=Sp(-ln(1-x))$ with the functions Sp of Spence.

Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y. p.998 (1970)

(1987.08.07) (1988.02.15)

DIGAM/DDIGAM (Digamma Function)

Digamma Function

Programmed by	Ichizo Ninomiya, April 1981
Format	Function Language: FORTRAN; Size: 37 and 47 lines respectively

(1) Outline

DIGAM (DDIGAM) calculates the digamma function $\psi(x)=\Gamma'(x)/\Gamma(x)$ for a single (double) precision real numbers x with single (double) precision.

(2) Directions

1. DIGAM(X) and DDIGAM(D)

X (D) is an arbitrary expression of a single (double) precision real number type. DDIGAM requires the declaration of double precision.

2. Range of argument

$X \geqq -2^{18}$, $D \geqq -2^{50}$

However, 0 and negative integers are excluded.

3. Error processing

If an argument outside the range is given, an error message is printed, and the calculation is continued with the function value as 0. (See "FNERST.")

(3) Calculation method

1. If $x<0.5$, the argument is converted to $x \geqq 0.5$ using the inversion formula $\psi(1-x)=\psi(x)+\pi \cot \pi x$.

2. If $0.5 \leqq x \leqq 16$, the argument is converted to $0.5 \leqq x \leqq 1.5$ using the recurrence formula $\psi(x+1)=\psi(x)+1/x$, and $\psi(x)$ is calculated as $\psi(x)=R(t)+t/(1+t)-r, t=x-1$ using the optimal rational approximation formula R.

3. If $x>16$, the optimal polynomial approximation related to $1/x^2$ of $\psi(x)-log(x)+1/2x$ is used.

56

Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., p.253 (1970).

(1987.07.09)

DILOG/DDILOG/CDILOG/CDDILG (Dilogarithm)

Dilogarithm

Programmed by	Ichizo Ninomiya, December 1987
Format	Function Language: FORTRAN; Size: 8, 9, 5, and 5 lines respectively

(1) Outline

DILOG(DDILOG, CDILOG, CDDILG) calculates the dilogarithm

$$dilog(x)=-\int_x^1 \frac{lnt}{t-1}dt$$

for single precision real numbers (double precision real number, single precision complex number, double precision complex number) x with single precision real numbers (double precision real number, single precision complex number, double precision complex number).

(2) Directions

1. DILOG(X), DDILOG(D), CDILOG(C), and CDDILG(B)

   X, D, C, and B are arbitrary expressions of corresponding types. DDILOG, CDILOG, and CDDILG require the declaration of corresponding types.

2. Range of argument

   X>0, D>0

3. Error processing

   If an argument outside the range is given, an error message is printed, and the calculation is continued with the function value as 0. (See "FNERST.")

(3) Calculation method

Spence function routines are called using the relationship $dilog(x)=S_p(1-x)$.

Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., p.1005 (1970).

(1987.07.07) (1988.01.27)

58

ERFC1/DERFC1 (Integral of complementary error function)

Integral of the Complementary Error Function

Programmed by	Ichizo Ninomiya: May 1986
Format	Function Language; FORTRAN Size; 55 and 110 lines respectively

(1) Outline

ERFC1 (or DERFC1) calculates i^1erfcx with single (or double) precision for single (or double) precision real x, where,

$$i^1 erfcx = \int_x^\infty erfc\,t\,dt \ ,$$

$$erfcx = \frac{2}{\sqrt{\pi}} \int_x^\infty e^{-t^2} dt \ .$$

(2) Directions

1. ERFC1(X), DERFC1(D)

   X (or D) is an arbitrary single (or double) precision real type expression. DERFC1 requires declaration of double precision.

2. Range of argument

   X≧0, D≧0

3. Error processing

   If the argument is outside the range, an error message is printed but calculation continues with the function value assumed to be 0. (Refer to FNERST.)

(3) Calculation method

(1) When $x<0$, an error results.

(2) When $0 \leqq x \leqq 2$, to calculate a rational function $f(x)$ is used.

$$i^1 erfcx = (f(x))^8$$

(3) When $2 < x \leqq 13.1$, a rational function g is used to calculate

$$i^1erfcx=e^{-x^2}g(4/x^2)$$

(4) When $x>13.1$,

$$i^1erfcx=0$$

Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., p.299 (1970).

(1987.08.05)

60

EXI/DEXI and EI/DEI (Exponential integral)

Exponential Integral

Programmed by	Ichizo Ninomiya; April 1985
Format	Function language; FORTRAN77, Size: 57 and 113 lines respectively

(1) Outline

For single (double) precision real x, EXI (DEXI) and EI (DEI) each calculate the following exponential integrations with single (double) precision:

$$Ei(x)=\int_{-\infty}^{x}\frac{e^t}{t}dt \quad ,E_1(x)=\int_{x}^{\infty}\frac{e^{-t}}{t}dt \qquad \left(E_1(x)=-Ei(-x)\right)$$

If x, $Ei(x)$ is regarded as the Cauchy's principal value.

(2) Directions

1. EXI (X), DEXI (D), EI (X), DEI (D)

   X (D) is an arbitrary single (double) precision real type expression. DEXI and DEI require declaration for double precision.

2. Range of argument: X≠0, X≦174.673 D≠0, and D≦174.673 for EXI and DEXI

   X≠0, X≧-174.673 D≠0, and D≧-174.673 for EI and DEI

3. Error processing

   If an argument outside the range is given, an error message is printed and calculation is continued with the function value put to 0.

(3) Calculation method

1. EXI (DEXI)

   (1) When $x<0$, $Ei(x)=-E_1(-x)$ is calculated by function routine EI or DEI.

   (2) When $0<x\leqq 6$, an optimal rational approximation A is used as follows:

$$Ei(x)=(x-z)\cdot A(6-x)+log(x/z)$$

Here, z=0.37250741078136663446 is the zero of $Ei(z)$.

(3) When $6<x\leqq 12$, an optimal rational approximation B is used as follows:

$$Ei(x)=B(6/x-1/2)\cdot e^x/x$$

(4) When $12<x\leqq 24$, an optimal polynomial approximation C is used as follows:

$$Ei(x)=C(1-12/x)\cdot e^x/x$$

(5) When $24<x\leqq 174.673$, an optimal rational approximation D is used as follows:

$$Ei(x)=D(1-24/x)\cdot e^x/x$$

2. Calculation method of EI(DEI)

(1) When $x<0$, the function routine EXI or DEXI is used to calculate;

$$E_1(x)=-Ei(-x)$$

(2) When $0<x\leqq 1$, a rational approximation P is used as follows:

$$E_1(x)=P(x)-logx$$

(3) When $1<x\leqq 2$, a rational approximation Q is used as follows:

$$E_1(x)=Q(x-1)$$

(4) When $2<x\leqq 175.040$, a rational approximation R is used as follows:

$$E_1(x)=R(2/x)e^{-x}/x$$

(5) When $x>$ 175.040, we put $E_1(x)=0$.

(4) Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y. p.228 (1970)

(1987.08.05) (1987.08.11) (1987.08.24)

(2

FRESS/DFRESS and FRESC/DFRESC (Fresnel Sine and Cosine Integrals)

Fresnel Sine and Cosine Integrals

Programmed by	Ichizo Ninomiya, April 1977
Format	Function Language: FORTRAN; Size: 72, 128, 72, and 128 lines respectively

(1) Outline

FRESS (DERRESS) calculates $S(x)$ and for a single (double) precision real numbers x with single (double) precision, and FRESC (DFRESC) calculates $C(x)$ similarly,

where,

$$S(x) = \frac{1}{\sqrt{2\pi}}\int_0^x \frac{\sin t}{\sqrt{t}}dt$$

$$C(x) = \frac{1}{\sqrt{2\pi}}\int_0^x \frac{\cos t}{\sqrt{t}}dt$$

(2) Directions

1. FRESS(X), DFRESS(D), FRESC(X), and DFRESC(D)

X (D) is an arbitrary expression of a single (double) precision real number type. The function name of double precision requires the declaration of double precision.

2. Range of argument

$0 \leqq X$, $0 \leqq D$

3. Error processing

If an argument outside the range is given, an error message is printed, and the calculation is continued with the function value as 0. (See "FNERST.")

(3) Calculation method

1. If $0 \leqq x \leqq 2$,

$S(x)=x\cdot\sqrt{x}A_s(x^2)$, $C(x)=\sqrt{x}A_c(x^2)$ are calculated using the polynomial approximations A_s, A_c.

2. If $2<x\leqq 4$,

$S(x)=B_s(x-3)$, $C(x)=B_c(x-3)$ are calculated using the polynomial approximations (rational) B_s, B_c.

3. If $4<x\leqq 6$,

$S(x)=C_s(x-5)$, $C(x)=C_c(x-5)$ are calculated using the polynomial approximations C_s, C_c.

4. If $6<x\leqq 8$,

$S(x)=D_s(x-7)$, $C(x)=D_c(x-7)$ are calculated using the polynomial approximations D_s, D_c.

5. If $8<x\leqq 8.23\cdot 10^5$ $(3.53\cdot 10^{15})$,

$$R(x)=E(t),\quad \varphi(x)=t\cdot F(t),\quad t=8/x,$$

$$S(x)=\frac{1}{2}-\cos(x-\varphi)\cdot\sqrt{\varphi\cdot t},$$

$$C(x)=\frac{1}{2}+\sin(x-\varphi)\cdot\sqrt{\varphi\cdot t}$$

are calculated using the polynomial approximations (rational) E, F.

6. If $x>8.23\cdot 10^5$ $(3.53\cdot 10^{15})$,

$$S(x)=\frac{1}{2},\quad C(x)=\frac{1}{2}$$

64

(4) Note

1. Note that Fresnel integrals have a different definition.

Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., P.300.

(1987.07.15) (1987.08.11)

HYPGM/DHYPGM/QHYPGM, CHPGM/DCHPGM/QCHPGM (Hypergeometric series and confluent hypergeometric series)

Title Hypergeometric Series and Confluent Hypergeometric Series

Programmed by	Ichizo Ninomiya; December 1987
Format	Function Language; FORTRAN Size; 21, 22, 22, 17, 18, and 18 lines respectively

(1) Outline

HYPGM, DHYPGM, and QHYPGM each calculate the following hypergeometric series, in single, double, or quadruple precision with the accuracy given by ε, for single, double, or quadruple precision real numbers a, b, c and x:

$$F(a,b,c,x)=\sum_{n=0}^{\infty}\frac{(a)_n(b)_n x^n}{(c)_n n!}$$

CHPGM, DCHPGM, and QCHPGM each calculate the following confluent hypergeometric series, in single, double, or quadruple precision with the accuracy given by ε, for single, double, and quadruple precision real numbers a, b and x:

$$M(a,b,x)=\sum_{n=0}^{\infty}\frac{(a)_n x^n}{(b)_n n!}$$

where $(a)_n=a\cdot(a+1)\cdots(a+n-1)$.

(2) Directions

1. HYPGM(A, B, C, X, E), DHYPGM(DA, DB, DC, DX, DE), QHYPGM(QA, QB, QC, QX, QE)

   CHPGM(A, B, X, E), DCHPGM(DA, DB, DX, DE), QCHPGM(QA, QB, QX, QE)

A, B, C, X, and E (DA, DB, DC, DX, DE; QA, QB, QC, QX, QE) are arbitrary single (double; quadruple) precision expressions. DHYPGM and DCHPGM (QHYPGM and QCHPGM) need to be declared as

66

double (quadruple) precision.

2. Range of argument

HYPGM (DHYPGM, QHYPGM): $0 \leqq |X|, |DX|, |QX| \leqq 1$

3. Error processing

If the specified argument is outside the range or if no convergence occurs after calculation of 1,000 terms, an error message is printed but calculation continues with the function value assumed to be 0.

(3) Calculation method

The series are accumulated sequentially starting from the first term. When the absolute value of the term added last is smaller than ϵ, then convergence is assumed to have occurred. The partial sum is used a the function value.

(4) Summary

Notice that some combinations of a, b, c, x and ϵ may cause overflow or underflow or nonconvergence.

Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y. p.505, p.556 (1970)

(1987.08.11) (1988.02.15)

ICEILS/D (Incomplete Elliptic Integrals of the First and Second Kind)

Incomplete Elliptic Integrals of the First and Second Kind

Programmed by	Ichizo Ninomiya, February 1982
Format	Subroutine Language: FORTRAN; Size: 74 and 79 lines respectively

(1) Outline

If the upper limit x and parameter m are given, ICEILS/D calculates the first and second kind incomplete elliptic integrals

$$F(x,m)=\int_0^x \frac{dt}{\sqrt{(1-t^2)(1-mt^2)}}=\int_0^\varphi \frac{d\theta}{\sqrt{1-m\sin^2\theta}}$$

and

$$E(x,m)=\int_0^x \sqrt{\frac{1-mt^2}{1-t^2}}\,dt=\int_0^\varphi \sqrt{1-m\sin^2\theta}\,d\theta$$

where,

$$0\leq x\leq 1,\ 0\leq\varphi\leq\pi/2,\ 0\leq m\leq 1,\ (x=\sin\varphi)$$

(2) Directions

CALL ICEILS/D(X, AM, F, E, CF, CE, IND)

Argument	Type and kind (*1)	Attribute	Content
X	Real type	Input	Upper limit of integral. $0\leq x\leq 1$
AM	Real type	Input	Parameter m($=k^2$). $0\leq m\leq 1$
F	Real type	Output	Incomplete elliptic integral of the first kind F(x,m).
E	Real type	Output	Incomplete elliptic integral of the second kind E(x,m).
CF	Real type	Output	Complete elliptic integral of the first kind K(m).
CE	Real type	Output	Complete elliptic integral of the second kind E(m).

18

Argument	Type and kind (*1)	Attribute	Content
IND	Integer type	Input/output	Input: IND=0: Only F and CF are calculated. IND≠0: F, E, CF, and CE are calculated. Output: IND=0: Normal termination. IND=10000: x=0, x=1, m=0, or m=1 occurred. The result is normal. IND=20000: Arithmetico-geometrical mean method did not converge even though the calculation was repeated 20 times. IND=30000: x<0, x>1, m<0, or m>1 occurred. The calculation is interrupted.

*1 For ICEILD, all real types should be changed to double precision real types.

(3) Calculation method

Gauss's arithmetico-geometrical mean method is used.

1. Starting from $a_0=1, b_0=\sqrt{1-m}, c_0=\sqrt{m}$, a sequence $(a_i, b_i, c_i), i=1,2,\cdots$ is generated as follows:

$$a_{i+1}=(a_i+b_i)/2,$$

$$b_{i+1}=\sqrt{a_i b_i},$$

$$c_{i+1}=(a_i-b_i)/2,$$

If $c_N \cong 0$ is reached with a sufficient precision, the generation is stopped. Then, the complete elliptic integrals are given by

$$K(m)=\pi/(2a_N),$$

$$E(m)=K(m)\left\{1-\frac{1}{2}(c_0^2+2c_1^2+2^2c_2^2+\cdots+2^N c_N^2)\right\}$$

2. Starting from $\varphi_0=\sin^{-1}x$, a sequence $\varphi_1, \varphi_2, \cdots, \varphi_N$ is generated as follows:

$$\tan(\varphi_{i+1}-\varphi_i)=(b_i/a_i)\tan\varphi_i, i=0,1,\cdots,N-1$$

where, $\varphi_{n+1}>\varphi_n$,

$$|\varphi_{n+1}-2\varphi_n|<\pi.$$

Then, the incomplete elliptic integrals are given as

$$F(x,m)=\varphi_N/(2^N a_N)$$

$$E(x,m)=(E(m)/K(m))\cdot F(x,m)+c_1\sin\varphi_1+\cdots+c_N\sin\varphi_N$$

3. If $m=1$,

$$CF=0,\ (K(m)=\infty)$$

$$CE=1$$

$$F=\frac{1}{2}\log\frac{1+x}{1-x}=\log M\left(\tan\left(\frac{\varphi}{2}+\frac{\pi}{4}\right)\right)$$

$$E=x$$

(4) Note

1. This subroutine is used to calculate the standard incomplete elliptic integral of the first and second kinds. It also outputs the complete elliptic integral of the first and second kinds as a by-product. When only the complete elliptic integral is to be calculated, however, it is more reasonable to use the special function routines CELI1 (DCELI1) and CELI2 (DCELI2).

2. This routine memorizes the AM value of the last call. If the same AM value is subsequently input, part of the step in 1 in "Calculation method" above is omitted. Thus, it is more reasonable to call this routine with the AM value left unchanged and only the X value changed.

3. If P is a cubic or quartic polynomial of t, and R is an arbitrary rational function, the general elliptic integral

$$\int_a^b R(\sqrt{P})\,dt$$

can be reduced to the three standard types: first kind $F(x,m)$, second kind $E(x,m)$, and third kind $\Pi(n;x,m)$ through appropriate variable transformation.

Where,

$$\Pi(n;x,m)=\int_0^x \frac{dt}{(1-nt^2)\sqrt{(1-t^2)(1-m^2t^2)}}$$

Bibliography

70

1) Handbook of Mathematical Functions, Dover, N.Y., pp.589-626 (1970).

(1989.01.20)

JACELS/D/Q (Jacobian Elliptic Functions sn,cn,dn sn,cn,dn))

Jacobian Elliptic Functions sn,cn,dn sn,cn,dn)

Programmed by	Ichizo Ninomiya, April 1977
Format	Subroutine Language: FORTRAN; Size: 54 and 55 lines respectively

(1) Outline

JACELS (JACELD) is a subroutine subprogram for calculating the Jacobian elliptic function $sn(u,k^2),cn(u,k^2),dn(u,k^2)$ and complete elliptic integral of the first kind $K(k^2)$ for single (double) real numbers u,k^2 with single (double) precision.

(2) Directions

CALL JACELS/D/Q(U, AK, SN, CN, DN, QP, ILL)

Argument	Type and kind (*1)	Attribute	Content
U	Real type	Input	Variable u. $\|U\| \leqq QP$
AK	Real type	Input	Square k^2 of the modulus k. $0 \leqq AK \leqq 1$
SN	Real type	Output	The value of the sn function is output.
CN	Real type	Output	The value of the cn function is output.
DN	Real type	Output	The value of the dn function is output.
QP	Real type	Output	The value of K is output.
ILL	Integer type	Output	ILL=0: Calculation is normally executed. ILL=30000: AK<0 or AK>1. ILL=1: |U|>QP occurred.

*1 For double precision subroutines, all real types should be changed to double precision real types.

(3) Calculation method

Salzer's arithmetico-geometrical mean method [1] is used.

72

(4) Note

Note that not the modulus k but k^2 is input in AK.

Bibliography

1) H. E. Salzer; "Quick Calculation of Jacobian Elliptic Functions", CACM, Vol. 5, p. 399, (1962).

(1987. 06. 26) (1987. 08. 21)

PN/DPN, PNM/DPNM (Legendre and Adjoint Legendre Polynomial)

Legendre and Adjoint Legendre Polynomial

Programmed by	Ichizo Ninomiya and Yasuyo Hatano, revised in April 1977; December 1987
Format	Function Language: FORTRAN; Size: 18, 19, 31, and 32 lines respectively

(1) Outline

PN (DPN) calculates $P_n(x)$ for an integer n and a single (double) precision real number x with single (double) precision.

PNM (DPNM) calculates $P_n^m(x)$ for integers n, m and a single (double) precision real numbers x with single (double) precision, where, $P_n(x)$ is a Legendre polynomial, and $P_n^m(x)$ is a Legendre adjoint polynomial.

(2) Directions

1. PN(N, X), DPN(N, D), PNM(N, M, X), and DPNM(N, M, D)

   N and M are arbitrary expressions of an integer type, and X (D) is an arbitrary expression of a single (double) precision real number type. DPN and DPNM require the declaration of double precision.

2. Range of argument

   $0 \leq N$, $0 \leq M \leq N$.

3. Error processing

   If an argument outside the range is given, an error message is printed, and the calculation is continued with the function value as 0. (See "PNERST.")

(3) Calculation method

1. PN (DPN)

   (1) If $n<0$, an error results.

   (2) If $n=0$, $P_n(x)=1$.

   (3) If $n=1$, $P_n(x)=x$.

74

(4) If $n \geqq 2$, the recurrence formula

$$P_k(x)=\frac{2k-1}{k}xP_{k-1}(x) - \frac{k-1}{k}P_{k-2}(x)$$

is sequentially applied to k $(2\sim n)$ beginning from $P_0(x)=1, P_1(x)=x$.

2. PNM (DPNM)

(1) If $n \geqq 0$ and $0 \leqq m \leqq n$ are not met, an error results.

(2) If $m=0$, $P_n^m(x)=P_n(x)$.

(3) If $m>0$, $P_n^m(x)=|1-x^2|^{m/2}F_n^m(x)$.

(4) If $n=m$, $F_n^m(x)=1\cdot 3\cdot\cdots\cdot(2n-1)$.

(5) If $n=m+1$, $F_n^m(x)=1\cdot 3\cdot\cdots\cdot(2n-1)\cdot x$.

(6) If $n \geqq m+2$, the recurrence formula

$$F_n^m(x)=\frac{2k-1}{k-m}xF_{k-1}^m(x) - \frac{k+m-1}{k-m}F_{k-2}^m(x)$$

is sequentially applied to k $(m+2\sim n)$ beginning from

$F_n^m(x)=1\cdot 3\cdot\cdots\cdot(2m-1)$, $F_{m+1}^m(x)=1\cdot 3\cdot\cdots\cdot(2m+1)\cdot x$.

(4) Note

The definition of Legendre adjoint polynomials of this routine is

$$P_n^m(x)=|1-x^2|^{\frac{m}{2}}\frac{d^mP_n(x)}{dx^m}$$

Note that there are different definitions.

(1987. 07. 03) (1988. 01. 08)

QN/DQN and QNM/DQNM (Legendre Functions and Adjoint Legendre of the Second Kind)

Legendre Functions and Adjoint Legendre Functions of the Second Kind

Programmed by	Ichizo Ninomiya, December 1987
Format	Function Language: FORTRAN; Size: 60, 61, 93, and 94 lines respectively

(1) Outline

QN(DQN) calculates $Q_n(x)$ for an integer n and a single (double) precision real number x with single (double) precision.

QNM(DQNM) calculates $Q_n^m(x)$ for integers n,m and a single (double) precision real numbers x with single (double) precision. Where, $Q_n(x)$ is the Legendre function of the second kind, and $Q_n^m(x)$ is the adjoint Legendre function of the second kind.

(2) Directions

1. QN(N, X), DQN(N, D), QNM(N, M, X), and DQNM(N, M, D)

   N and M are of an integer type, and X(D) is an arbitrary expression of a single (double) precision real number type. DQN and DQNM require the declaration of double precision.

2. Range of argument

   $0 \leq N$, $0 \leq M \leq N$. $|X| \neq 1$, $|D| \neq 1$

3. Error processing

   If an argument outside the range is given, an error message is printed, and the calculation is continued with the function value as 0. (See "FNERST.")

(3) Calculation method

1. QN(DQN)

   (1) If $n<0$ or $|x|=1$, an error results.

   (2) If $|x| \leq 0.9$, do the following:

   (i) If $n=0$, $Q_0(x)=1/2 \ln 1+x/1-x$.

   (ii) If $n=1$, $Q_1(x)=x \cdot Q_0(x)-1$.

(iii) If $n \geqq 2$, the recurrence formula $Q_k(x)=((2k-1)\cdot x\cdot Q_{k-1}(x)-(k-1)Q_{k-2}(x))/k$ is sequentially applied to k $(2\sim n)$ starting from $Q_0(x), Q_1(x)$.

(3) If $|x|>0.9$, the calculation is reduced to the case of $x>0$ using $Q_n(-x)=(-1)^{n+1}Q_n(x)$

(4) If $0.9<x\leq 1.05$, the calculation conforms to the theoretical formula

$$Q_n(x)=\frac{1}{2}\ln\left|\frac{1+x}{1-x}\right|\cdot\sum_{k=0}^{n}\frac{(-n)_k(n+1)_k}{(k!)^2}\left(\frac{1-x}{2}\right)^k$$

$$-\sum_{k=0}^{n}(\varphi(n)-\varphi(k))\frac{(-n)_k(n+1)_k}{(k!)^2}\left(\frac{1-x}{2}\right)^k$$

Where,

$$\varphi(n)=\sum_{k=1}^{n}\frac{1}{k}$$

(5) If $x>1.05$, do the following:

(i) Take a sufficiently large integer N (details omitted), and set $F_N=10^{-70}, F_{N-1}=2xF_N$.

(ii) Apply the opposite direction recurrence formula

$$F_k(x)=((2k+3)xF_{k+1}-(k+2)\cdot F_{k+2})/(k+1)$$

to k $(N-1\sim 0)$ sequentially.

(iii) $Q_n(x)=F_n\cdot Q_0(x)/F_0$. Where,

$$Q_0(x)=\frac{1}{2}\ln\left|\frac{1+x}{1-x}\right|$$

2. QNM(DQNM)

(1) If $m<0, n<m$ or $|x|=1$, an error results.

(2) If $|x|\leq 0.8$, do the following:

(i) Set $F_0^0=1/2\cdot ln(1+x/1-x), F_1^0=x\cdot Q_0(x)-1$.

(ii) F_m^m, F_{m+1}^m are given by applying the two recurrence formulas

$$F_k^k=(xF_k^{k-1}-(2k-1)F_{k-1}^{k-1})/(1-x^2),\ F_{k+1}^k=xF_k^k+2kF_k^{k-1}$$

alternatively to k $(k=1\sim m-1)$.

(iii) If $n=m$, $Q_n^m(x)=(1-x^2)^{m/2}F_m^m$.

(iv) If $n=m+1$, $Q_n^m(x)=(1-x^2)^{m/2}F_{m+1}^m$.

(v) If $n\geq m+2$, $Q_n^m(x)=(1-x^2)^{m/2}F_n^m$ is given by applying the recurrence formula

$F_k^m=((2k-1)xF_{k-1}^m-(k+m-1)F_{k-1}^m)/(k-m)$ to k $(k=m+2\sim n)$.

(3) If $|x|>0.8$, the calculation is reduced to the case of $x>0$ using $Q_n^m(-x)=(-1)^{n+m+1}Q_n^m(x)$

.

(4) If $0.8<x\leq 1.1$, the calculation conforms to the theoretical formula

$$Q_n^m(x)=\frac{|1-x^2|^{m/2}}{2^{m+1}}\left\{\frac{(n+m)!}{(n-m)!m!}\sum_{k=0}^{n-m}(Q_0(x)-\varphi(n-m)+\varphi(k))\cdot\right.$$

$$\frac{(-n+m)_k(n+m+1)_k}{(m+1)_k\ k!}t^k+\left(\frac{2}{1+x}\right)^m\sum_{k=0}^{n}(Q_0(x)-\varphi(n+m)+\varphi(m+k))\cdot$$

$$\left.\frac{(-n)_k(n+1)_k}{(m+1)_k k!}t^k\right\}+\frac{2^{m-1}}{(1-x)^m}\sum_{k=0}^{m-1}\frac{(-n)_k(n+1)_k(m-1-k)!}{k!}(-t)^k$$

.

(5) If $x>1.1$, do the following:

(i) Take a sufficiently large (details omitted) integer N, and set $F_N^N=10^{-70}, F_{N-1}^N=2xF_N^N$.

(ii) F_m^m, F_{m+1}^m is given by applying the opposite direction recurrence formula to k $(k=N-1\sim m)$ sequentially.

(iii) The two recurrence formulas

$$F_{k+1}^k=(F_{k+2}^{k+1}-xF_{k+1}^{k+1})/(2k+2),$$

and

$$F_k^k=(xF_{k+1}^k+(1-x^2)F_{k+1}^{k+1})/(2k+1)$$

are alternatively applied to k $(k=m-1\sim 0)$ starting from F_m^m, F_{m+1}^m.

(iv) $Q_n^m(x)=(x^2-1)^{m/2}F_n^m\cdot Q_0(x)/F_0^0$.

78

(4) Summary

This routine conforms to

$$Q_n^m(x) = |1-x^2|^{m/2} \frac{d^m Q_n(x)}{dx^m}$$

. Note that there are other definitions.

Bibliography

1) Handbook of Mathematical Functions, Dover N.Y., p.332 (1970).

(1987.07.03) (1988.01.08) (1988.02.15)

QNOME/DQNOME (The Nome of Elliptic θ Functions)

The Nome of Elliptic θ Functions

Programmed by	Ichizo Ninomiya, April 1981
Format	Function Language: FORTRAN; Size: 23 and 32 lines respectively

(1) Outline

QNOME (DQNOME) calculates the nome $q(x)$ of elliptic θ functions for a single (double) precision real numbers x with single (double) precision, where, x represents the parameter $m=k^2$ of elliptic functions.

(2) Directions

1. QNOME(X) and DQNOME(D)

X (D) is an arbitrary expression of a single (double) precision real number type. DQNOME requires the declaration of double precision.

2. Range of argument

$0 \leqq X \leqq 1$, $0 \leqq D \leqq 1$

3. Error processing

If an argument outside the range is given, an error message is printed, and calculation is continued with the function value as 0. (See "FNERST.")

(3) Calculation method

1. If $0 \leqq x \leqq 0.5$, $q(x)$ is calculated using the optimal rational approximation formula.

2. If $0.5 < x \leqq 1$, the calculation is converted to that of the complementary nome $q(1-x)$ using the relation $\log q(x) \log q(1-x) = \pi^2$.

Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., p.591 (1970).

(1987.06.25)

80

RGAMA/DRGAMA (Reciprocal of Gamma Function)

Reciprocal of Gamma Function

Programmed by	Ichizo Ninomiya, April 1981
Format	Function Language: FORTRAN; Size: 41 and 54 lines respectively

(1) Outline

RGAMA (DRGAMA) calculates $1/\Gamma(x)$ for a single (double) precision real numbers x with single (double) precision.

(2) Directions

1. RGAMA(X) and DRGAMA(D)

X (D) is an arbitrary expression of a single (double) precision real number type. DRGAMA requires the declaration of double precision.

2. Range of argument

$X \geqq -56$, $D \geqq -56$

3. Error processing

If an argument outside the range is given, an error message is printed, and the calculation is continued with the function value as 0. (See "FNERST.")

(3) Calculation method

1. If $-56 \leqq x < 0$, the calculation is reduced to that of $1/\Gamma(1-x)$ using the following inversion formula.

$$\frac{1}{\Gamma(x)} = \frac{\Gamma(1-x)\sin\pi x}{\pi}$$

2. If $0 \leqq x \leqq 32$, the argument is converted to that in the range of $1 \leqq x \leqq 2$ as required using the recurrence formula $\Gamma(1+x)=x\Gamma(x)$, and $1/\Gamma(x)=P(t), t=x-1$ is calculated with the optimal approximation polynomial P.

3. If $32 < x \leqq 57$, $\log\Gamma(x)$ is computed using an approximation formula and $1/\Gamma(x)$ is computed as follows:

$$\frac{1}{\Gamma(x)}=e^{-\log\Gamma(x)}$$

4. If $x>57$,

$$\frac{1}{\Gamma(x)}=0$$

(4) Note

1. $\Gamma(x)$ has the pole of the order 1 at $x=0, -1, -2, \cdots\cdots$. However, $1/\Gamma(x)$ has no singularity other than infinity. It is a so-called entire function.

2. For the calculation of rational functions that have a Gamma function in the denominator, it is more reasonable to use this function program rather than the Gamma function program.

Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., p.253 (1970).

(1987.07.31)

82

SI/DSI and CI/DCI (Sine and Cosine Integrals)

Sine and Cosine Integrals

Programmed by	Ichizo Ninomiya, May 1983
Format	Function Language: FORTRAN; Size: 75, 130, 74, and 137 lines respectively

(1) Outline

SI(DSI) calculates $S_i(x)$ for a single (double) precision real numbers x with single (double) precision, and CI(DCI) calculates $C_i(x)$ similarly,

where,

$$S_i(x)=\int_0^x \frac{\sin t}{t}dt$$

$$C_i(x)=-\int_x^\infty \frac{\cos t}{t}dt$$

(2) Directions

1. SI(X), CI(X), DSI(D), and DCI(D)

X (D) is an arbitrary expression of a single (double) precision real number type. The function name of double precision requires the declaration of double precision.

2. Range of argument

$0 \leq X$ and $0 \leq D$ for sine integral.

$0 < X$ and $0 < D$ for cosine integral.

3. Error processing

If an argument outside the range is given, an error message is printed, and the calculation is continued with the function value as 0. (See "FNERST.")

(3) Calculation method

1. If $0 \leqq x \leqq 2$ ($0 < x \leqq 2$ in case of cosine integral),

$$S_i(x) = x \cdot A_s(x^2), \quad C_i(x) = A_c(x^2) + \log x$$

is calculated using the polynomial approximations A_s, A_c.

2. If $2 < x \leqq 4$,

$$S_i(x) = B_s(x-3), \quad C_i(x) = B_c(x-3)$$

is calculated using the polynomial approximations B_s, B_c.

3. If $4 < x \leqq 6$,

$$S_i(x) = C_s(x-5), \quad C_i(x) = C_c(x-5)$$

is calculated using the polynomial approximations C_s, C_c.

4. If $6 < x \leqq 8$,

$$S_i(x) = D_s(x-7), \quad C_i(x) = D_c(x-7)$$

is calculated using the polynomial approximations D_s, D_c.

5. If $8 < x \leqq 8.23 \cdot 10^5$ $(3.53 \cdot 10^{15})$,

$$R(x) = E(t), \quad \varphi(x) = t \cdot F(t), \quad t = 8/x \text{ and}$$

$$S_i(x) = \frac{\pi}{2} - R \cdot t \cdot \cos(x-\varphi), \quad C_i(x) = R \cdot t \cdot \sin(x-\varphi)$$

are calculated using the polynomial approximations (rational approximations) E, F.

6. If $x > 8.23 \cdot 10^5$ $(3.53 \cdot 10^{15})$,

$$S_i(x) = \frac{\pi}{2}, \quad C_i(x) = 0$$

Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., p.231.

(1987.07.31) (1987.08.11)

84

SPENC/DSPENC and CSPENC/CDSPEN (Spence Function)

Spence Function

Programmed by	Ichizo Ninomiya, December 1987
Format	Function Language: FORTRAN77; Size: 38, 58, 41, and 62 lines respectively

(1) Outline

SPENC(DSPENC, CSPENC, CDSPEN) calculates the Spence function

$$Sp(x)=-\int_0^x \frac{ln(1-t)}{t}dt$$

for single precision real numbers (double precision real number, single precision complex number, double precision complex number) x with single precision real numbers (double precision real number, single precision complex number, double precision complex number).

(2) Directions

1. SPENC(X), DSPENC(D), CSPENC(C), CDSPEN(B)

   X(D, C, B) is an arbitrary expression of corresponding type. DSPENC, CSPENC, and CDSPEN require the declaration of corresponding types.

2. Range of argument

   No limit.

(3) Calculation method

1. Calculation method of SPENC(DSPENC)

   (1) If $|x| \leq 1/4$, $Sp(x)=x \cdot A(x)$ is calculated using an optimal polynomial approximation A.

   (2) If $1-e^2 \leq x \leq 1-e^{-2}$, $Sp(x)=t \cdot (1-t/4+t^2 \cdot B(t^2))$ is calculated using an optimal

polynomial approximation B.

Where, $t=-ln(1-x)$

(3) If $1-e^{-2}<x\leqq 2$, $Sp(x)=\pi^2/6+t(1+t/4+t^2\cdot B(t^2))-t\cdot ln|1-x|$ is calculated.

Where, $t=lnx$

(4) If $x<1-e^2$, $Sp(x)=-\pi^2/6-ln^2(-x)/2-t\cdot A(t)$ is calculated. Where, $t=1/x$.

(5) If $x=1$, $Sp(x)=\pi^2/6$.

(6) If $x>2$, $S_p(x)=\pi^2/3+t\cdot(1+t/4+t^2B(t^2))-ln^2x/2$ is calculated. Where, $t=ln(1-1/x)$

2. Calculation method of CSPENC(CDSPEN)

(1) If $|z|<0.1$, $Sp(z)=z\sum_{n=1}^{6(16)}z^n/n^2$ is calculated.

(2) If $|z|\leqq 1, Re(z)\leqq 1/2$, $Sp(z)=t\cdot(1-t/4+\sum_{n=1}^{4(11)}B_{2n}t^{2n}/(2n+1)!)$ is calculated.

Where, $t=-ln(1-z)$

(3) If $|z-1|\leqq, Re(z)>1/2$,

$Sp(z)=\pi^2/6-t\cdot ln(1-z)+t(1+t/4+\sum_{n=1}^{4(11)}B_{2n}t^{2n}/(2n+1)!)$ is calculated.

Where, $t=lnz$

(4) If $|z|>1$, $|z-1|>1$,

$Sp(z)=-\pi^2/6-ln^2(-z)/2+t(1+t/4+\sum_{n=1}^{4(11)}B_{2n}t^{2n}/(2n+1)!)$ is calculated.

(4) Note

1. There is the relationship $dilog(1-z)=Sp(z)$ with Dilogarithm $dilog$ of Euler. Therefore, the value of Spence functions can be calculated even with the function routine DILOG. However, it is more effective to use SPENC in this section.

2. There is the relationship $Db(-ln(1-x))=Sp(x)$ with the functions Db of Debye.

Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y. pp.997-1005 (1970).

(1987.08.05) (1987.08.11) (1987.08.24) (1988.02.15)

86

TMFRM/DTMFRM/TMFMP/DTMFMP (Solution of Thomas-Fermi Equation and its derivative)

Solution of Thomas-Fermi Equation and its Derivative

Programmed	Ichizo Ninomiya: April 1985
Format	Function Language; FORTRAN Size: 38, 59, 36, and 54 lines respectively

(1) Outline

TMFRM (or DTMFRM) calculates $\phi(x)$ with single (or double) precision for single (or double) precision real x.

TMFMP (or DTMFMP) calculates $\phi'(x)$ with single (or double) precision for single (or double) precision real x,

$\phi''(x)=\phi(x)**(3/2)\cdot x**(-1/2)$ where ϕ satisfies the Thomas-Fermi equation.

(2) Directions

1. TMFRM(X), TMFMP(X), DTMFRM(D), DTMFMP(D)

   X (or D) is an arbitrary single (or double) precision real type expression. DTMFRM and DTMFMP require declaration of double precision.

2. Range of argument $X \geqq 0$ and $D \geqq 0$

3. Error processing

   If the argument is outside the range, an error message is printed but calculation continues with the function value assumed to be 0.

(Refer to FNERST.)

(3) Calculation method

1. When $0 \leqq x \leqq 1$, optimal rational approximations A and B are used to calculate:

   $\phi(x)=A(x**(1/2))$

   $\phi'(x)=B(x**(1/2))$

2. When $1 \leqq x \leqq a$, optimal rational approximations C and D are used to calculate:

   $\phi(x)=C(x**\nu)**4/x**3$

   $\phi'(x)=\phi(x)\cdot D(x**\nu)/x$

   where, $\nu=-0.77200187265876558394$, and

   $a=5.4\cdot 10**9\ (9.5\cdot 10**23)$

3. When $a \leqq x \leqq b$, then $\phi(x)=144/x**3$ is calculated.

When $a \leqq x \leqq b'$, then $\phi'(x) = -432/x^{**}4$ is calculated.

Where, $b = 2.43 \cdot 10^{**}26$ and $b' = 1.68 \cdot 10^{**}20$.

4. When $x > b$, then $\phi(x) = 0$. When $x > b'$, then $\phi'(x) = 0$.

(4) Bibliography

1) H. Krutter; "Numerical Integration of the Thomas-Fermi Equation from Zero to Infinity," J. of Comp. Physics, Vol. 47, pp. 308-312 (1982).

(1987. 08. 07)

88

ZETA/DZETA (Riemann zeta function)

Title: Riemann Zeta Function

Programmed by	Ichizo Ninomiya: May 1984
Format	Function language: FORTRAN Size: 94 and 213 lines respectively

(1) Outline

When single (double) precision real x is given, ZETA (DZETA) calculates the following Riemann zeta functions in single (double) precision.

$$\zeta(x)=\sum_{k=0}^{\infty}k^{-x},\ x>1$$

$$\zeta(x)=\left(\sum_{k=0}^{\infty}(-1)^{k-1}k^{-x}\right)/(1-2^{1-x}),$$

$$\zeta(x)=2^{x}\pi^{x-1}\sin(\pi x/2)\Gamma(1-x)\zeta(1-x),\ x\leq 0$$

(2) Directions

1. ZETA(X) DZETA(D)

Use an arbitrary single (X) or double (D) precision real type expression. DZETA needs to be declared as double precision.

2. Range of argument: X≧-97 D≧-97

3. Error processing

If an argument outside the range is given, an error message is printed but calculation continues assuming the function value to be 0.

(3) Calculation method

1. In case of X≧0, optimal polynomial or rational approximation is used depending on the range of X.

2. In case of X<0, it is reduced to a case of X≧0 by the following inversion formula:

$$\zeta(x)=2^x\pi^{x-1}\sin(\Pi x/2)\Gamma(1-x)\zeta(1-x)$$

Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., p.807 (1970).

(1987.07.27)

90

14. Bessel function and related function

AI/DAI, AIP/DAIP, BI/DBI, BIP/DBIP (Airy functions and their derivatives)

Airy Functions and Their Derivatives

Programmed by	Ichizo Ninomiya; April 1981
Format	Function Language; FORTRAN Size; 74, 131, 72, 130, 90, 160, 93, and 159 lines respectively

(1) Outline

AI (DAI), AIP (DAIP), BI (DBI), and BIP (DBIP) calculate $A_i(x)$, $A_i'(x)$, $B_i(x)$, and $B_i'(x)$ respectively, with single (double) precision, for a single (double) precision real number x.

(2) Directions

1. AI(X), DAI(D), AIP(X), and DAIP(D)

   BI(X), DBI(D), BIP(X), and DBIP(D)

   X and D are arbitrary single and double precision real-type expressions respectively. DAI, DAIP, DBI, and DBIP need to be declared as double precision.

2. Range of argument

(1) AI, DAI, AIP, and DAIP

$$X \geq -1.15130 \cdot 10^4 \quad \left(\frac{2}{3}|X|^{3/2} \leq 2^{18}\pi\right)$$

$$D \geq -3.04201 \cdot 10^{10} \quad \left(\frac{2}{3}|D|^{3/2} \leq 2^{50}\pi\right)$$

(2) BI, DBI, BIP, and DBIP

$$-1.15130 \cdot 10^4 \leq X \leq 40.946 \begin{cases} X<0 \cdots \frac{2}{3}|X|^{3/2} \leq 2^{18}\pi \\ X>0 \cdots \frac{2}{3}|X|^{3/2} \leq 252 \log 2 \end{cases}$$

92

$$-3.04201\cdot 10^{10}\leq D\leq 40.946\begin{cases}D<0\cdots\frac{2}{3}|D|^{3/2}\leq 2^{18}\pi\\ D>0\cdots\frac{2}{3}|D|^{3/2}\leq 252\log 2\end{cases}$$

3. Error processing

If the specified argument is outside the range, an error message is printed but calculation continues with the function value assumed to be 0. (See FNERST.)

(3) Calculation method

1. AI and DAI

(1) When $x<-9^{1/3}$, optimal rational approximations C_1, S_1 are used to calculate:

$A_i(x)=|x|^{-1/4}(C_1(2/z)\cos z+S_1(2/z)\sin z)$ where $z=2/3|x|^{3/2}$.

(2) When $-9^{1/3}\leq x<0$, optimal polynomial approximations P_1, Q_1 are used to calculate

$A_i(x)=P_1(x^3)+|x|Q_1(x^3)$

(3) When $0\leq x\leq 9^{1/3}$, optimal rational approximation R_1 is used to calculate $A_i(x)=R_1(x)$.

(4) When $9^{1/3}<x<41.808(2<2/3x^{3/2}<260\log 2)$, optimal rational approximation E_1 is used to calculate:

$A_i(x)=e^{-z}|x|^{-1/4}E_1(2/z)$ where $z=2/3x^{3/2}$.

(5) When $x\geq 41.808$, $A_i(x)=0$.

2. AIP and DAIP

(1) When $x<-9^{1/3}$, optimal rational approximations C_2, S_2 are used to calculate:

$A_i'(x)=|x|^{1/4}(C_2(2/z)\cos z+S_2(2/z)\sin z)$ where $z=2/3|x|^{3/2}$.

(2) When $-9^{1/3}\leq x<0$, optimal polynomial approximations P_2, Q_2 are used to calculate

$A_i'(x)=x^2P_2(x^3)+Q_2(x^3)$.

(3) When $0\leq x\leq 9^{1/3}$, optimal rational approximation R_2 is used to calculate $A_i'(x)=R_2(x)$.

(4) When $9^{1/3}<x<41.808$, optimal rational approximation E_2 is used to calculate:

$A_i'(x)=x^{1/4}e^{-z}E_2(2/z)$ where $z=2/3x^{3/2}$.

(5) When $x\geq 41.808$, $A_i'(x)=0$.

3. BI and DBI

(1) When $x<-9^{1/3}$, optimal rational approximations C_3, S_3 are used to calculate:

$B_i(x)=|x|^{-1/2}(C_3(2/z)\cos z + S_3(2/z)\sin z)$ where $z=2/3|x|^{3/2}$.

(2) When $-9^{1/3}\leq x<0$, optimal polynomial approximations P_3, Q_3 are used to calculate

$B_i(x)=P_3(x^3) + xQ_3(x^3)$.

(3) When $0\leq x\leq 144^{1/3}$, optimal rational approximations A_1, B_1 are used to calculate

$B_i(x)=A_1(x^3) + xB_1(x^3)$.

(4) When $144^{1/3}<x$, optimal rational approximation E_3 is used to calculate:

$B_i(x)=x^{-1/2}e^z E_3(1-8/z)$ where $z=2/3x^{3/2}$.

4. BIP and DBIP

(1) When $x<-9^{1/3}$, optimal rational approximations C_4, S_4 are used to calculate:

$B_i'(x)=|x|^{1/4}(C_4(2/z)\cos z+S_4(2/z)\sin z)$ where $z=2/3|x|^{3/2}$.

(2) When $-9^{1/3}\leq x<0$, optimal polynomial approximations P_4, Q_4 are used to calculate

$B_i'(x)=x^2P_4(x^3)+Q_4(x^3)$.

(3) When $0\leq x\leq 144^{1/3}$, optimal rational approximations A_2, B_2 are used to calculate

$B_i'(x)=x^2A_2(x^3)+B_2(x^3)$.

(4) When $144^{1/3}<x$, optimal rational approximation E_4 is used to calculate:

$B_i'(x)=x^{1/4}e^z E_4(1-8/z)$ where $z=2/3x^{3/2}$.

Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., p.446 (1970).

(1989.01.13)

94

BERO/DBERO,BEIO/DBEIO,BKERO/DKERO,BKEIO/DKEIO,
BER1/DBER1,BEI1/DBEI1,BKER1/DKER1,BKEI1/DKEI1

(Kelvin functions of the order 0 and 1)

Kelvin Functions of the Order 0 and 1

Programmed by	Ichizo Ninomiya: September 1984
Format	Function language; FORTRAN Size; All 150 lines or less

(1) Outline

BERO(DBERO) $ber_0 x$

BEIO(DBEIO) $bei_0 x$

BKERO(DKERO) $ker_0 x$

BKEIO(DKEIO) $kei_0 x$

calculate for single or double precision real x respectively.

BER1(DBER1) $ber_1 x$

BEI1(DBEI1) $bei_1 x$

BKER1(DKER1) $ker_1 x$

BKEI1(DKEI1) $kei_1 x$

(2) Directions

1. BERO(X), BEIO(X), BKERO(X), and BKEIO(X), etc.

   DBERO(D), DBEIO(D), DKERO(D), and DKEIO(D), etc.

   X is an arbitrary expression of single precision real type. D is an arbitrary expression of double precision real type. The name of a double precision function needs to be declared as double precision.

2. Range of argument

   $0 \leq X \leq 247.02264$ for ber and bei functions.

   $0 \leq D \leq 247.02264$.

   $0 < X$, $0 < D$ for ker and kei functions.

3. Error processing

   If an argument outside the range is given, an error message is printed and calculation is

continued with the function value assumed to be 0. (Refer to FNERST.)

(3) Calculation method

1. BER0(DBER0), BEI0(DBEI0), BER1(DBER1), and BEI1(DBEI1)

(1) If x is less than 0, an error results.

(2) If $0 \leqq x \leqq 2$, $ber_0 x = A_0 r(x^4)$, $bei_0 = x^2 A_0 i(x^4)$, $ber_1 x = x A_1 r(x^2)$, $bei_1 = x A_1 i(x^2)$ are calculated by the optimal polynomial approximations $A_0 r, A_0 i, A_1 r, A_1 i$.

(3) If $2 < x \leqq 4$, $ber_0 x = B_0 r(x-3)$, $bei_0 x = B_{0i}(x-3)$, $ber_1 x = B_1 r(x-3)$, $bei_1 x = B_1 i(x-3)$ are calculated by the optimal polynomial approximations $B_0 r, B_0 i, B_1 r, B_1 i$.

(4) If $4 < x \leqq 6$, $ber_0 x = C_0 r(x-5)$, $bei_0 x = C_{0i}(x-5)$, $ber_1 x = C_1 r(x-5)$, $bei_1 x = C_1 i(x-5)$ are calculated by the optimal polynomial approximations $C_0 r, C_0 i, C_1 r, C_1 i$.

(5) If $6 < x \leqq 8$, $ber_0 x = D_0 r(x-7)$, $bei_0 x = D_0 i(x-7)$, $ber_1 x = D_1 r(x-7)$, $bei_1 x = D_1 i(x-7)$ are calculated by the optimal polynomial approximations $D_0 r, D_0 i, D_1 r, D_1 i$.

(6) If $8 < x \leqq 247.02264$,

$$ber_0 x = e^{x/\sqrt{2}} \cdot \sqrt{E_0(8/x) \cdot (8/x)} \cos(G_0(8/x) + x/\sqrt{2}) - \frac{1}{\pi} kei_0 x$$

$$bei_0 x = e^{x/\sqrt{2}} \cdot \sqrt{E_0(8/x) \cdot (8/x)} \sin(G_0(8/x) + x/\sqrt{2}) + \frac{1}{\pi} ker_0 x$$

$$ber_1 x = e^{x/\sqrt{2}} \cdot \sqrt{E_1(8/x) \cdot (8/x)} \cos(G_1(8/x) + x/\sqrt{2}) - \frac{1}{\pi} kei_1 x$$

91

$$bei_1x=e^{x/\sqrt{2}}\cdot\sqrt{E_{10}(8/x)\cdot(8/x)}\sin(G_1(8/x)+x/\sqrt{2})+\frac{1}{\pi}ker_1x$$

are calculated by the optimal polynomial approximations E_0, G_0, E_1, G_1.

(7) If x>247.02264, an error results.

2. BKER0(DKER0), BKEI0(DKEI0), BKER1(DKER1), BKEI1(DKEI1)

(1) If x≦0, an error results.

(2) If 0<x≦2,

$$ker_0x=A_0r(x^2)-logx\cdot ber_0x$$

$$kei_0x=A_0i(x^2)-logx\cdot bei_0x$$

$$ker_1x=A_1r(x^2)/x-logx\cdot ber_1x$$

$$kei_1x=A_1i(x^2)/x-logx\cdot bei_1x$$

are calculated by the optimal polynomial approximations A_0r, A_0i, A_1r, A_1r.

(2) If 2<x≦4, $ker_0x=B_0r(x-3), kei_0x=B_0i(x-3), ker_1x=B_1r(x-3), kei_1x=B_1i(x-3)$ are calculated by the optimal rational approximations B_0r, B_0i, B_1r, B_1i.

(3) If 4<x≦254.8646,

$$ker_0x=e^{-x/\sqrt{2}}\sqrt{E_0(4/x)\cdot(4/x)}\cos(G_0(4/x)-x/\sqrt{2})$$

$$kei_0x=e^{-x/\sqrt{2}}\sqrt{E_0(4/x)\cdot(4/x)}\sin(G_0(4/x)-x/\sqrt{2})$$

$$ker_1x=e^{-x/\sqrt{2}}\sqrt{E_1(4/x)\cdot(4/x)}\cos(G_1(4/x)-x/\sqrt{2})$$

$$kei_1x=e^{-x/\sqrt{2}}\sqrt{E_1(4/x)\cdot(4/x)}\sin(G_1(4/x)-x/\sqrt{2})$$

are calculated by the optimal polynomial (rational) approximations E_0, G_0, E_1, G_1.

(4) If x>254.8646, $ker_0 x = kei_0 x = ker_1 x = kei_1 x = 0$ results.

Bibliography

(1) Handbook of Mathematical Functions, Dover, N.Y., pp.379-385 (1970).

(1989.01.17)

98

BESJFC/B and BESIFC/B (Bessel Functions of Fractional Order with Complex Argument)

Bessel Functions of Fractional Order with Complex Argument

Programmed by	Toshio Yoshida
Format	Function Language: FORTRAN; Size: 187, 189, 231, and 232 lines respectively

(1) Outline

BESJFC (BESJFB) is a function subprogram for obtaining the first kind Bessel function $J_\nu(z)$ of the ν-th order (real number) of complex variables z by single (double) precision.

BESIFC (BESIFB) is a function subprogram for obtaining the first kind modified Bessel function $I_\nu(z)$ of the ν-th order (real number) of complex variables z by single (double) precision.

(2) Directions

1. BESJFC(N, Z), BESJFB(N, Z)

   BESIFC(N, Z), BESIFB(N, Z)

   Declaration such as COMPLEX*8 BESJFC, Z, or COMPLEX*16 BESIFB, Z is required for function names and the argument Z.

2. Range of argument

   $|Re(z)| \leqq 174.673$

   $|Im(z)| \leqq 174.673$

3. Error processing

   If an argument outside the range is given, an error message is output, and the calculation is continued with the function value as 0. (See FNERST.)

(3) Calculation method

The cutting plane line is a negative real axis.

The Taylor expansion is used in $|Re(z)|+|Im(z)| \leqq 1$, and the method of using recurrence

technique is used in other cases. For details, see "Bibliography."

Bibliography

1) Toshio Yoshida et al., "Recurrence Techniques for the Calculation of Bassel Function In(z) with Complex Argument." Information Processing, Vol. 14, No. 1, and pp. 23-29 (1973).

(1987. 08. 07)

/00

BESJNC/B and BESINC/B (Bessel Functions of Integral Order with Complex Argument)

Bessel Functions of Integral Order with Complex Argument

Programmed by	Toshio Yoshida, June 1973, Revised in June 1985
Format	Function Language: FORTRAN; Size: 152, 153, 145, and 146 lines respectively

(1) Outline

BESJNC (BESJNB) is a function subprogram for obtaining the Bessel function of first kind of the n-th (integer) order of the complex variable z with single (double) precision.

BESINC (BESINB) is a function subprogram for obtaining the modified Bessel function of first kind of the n-th (integer) order of the complex variable z with single (double) precision.

(2) Directions

1. BESJNC(N, Z), BESJNB(N, Z)

   BESINC(N, Z), BESINB(N, Z)

   Declarations such as COMPLEX*8 BESJNC, Z, or COMPLEX*16 BESJNB, Z are required for function names and the argument Z.

2. Range of arguments: $|\mathrm{Real}(z)| \leqq 174.673$ and $|\mathrm{Imag}(z)| \leqq 174.673$

3. Error processing

   If an argument outside the range is given, an error message is output, and the calculation is continued with the function value as 0. (See FNERST.)

(3) Calculation method

Recurrence formulas are used for calculation. For details, see "Bibliography."

(4) Note

Because this calculation uses recurrence formulas, the computation time becomes longer as $|z|$ becomes larger. If $|z|>100$, it is efficient to use the asymptotic expansions.

Bibliography

1) Toshio Yoshida; "Bessel Function Subprogram of Complex Variable: In (z) and Jn (z)," Nagoya University Computer Center News, Vol. 5, No. 3, pp. 179-185 (1974).

2) Toshio Yoshida et al.; "Reccurence Techniques for the Calculation of Bassel Funciton In(z) with Complex Argument," Information Processing, Vol. 14, No. 1, pp. 23-29 (1973).

(1987. 08. 07)

102

BESKNC/B (Modified Bessel Functions of the Second Kind of Integral Order with Complex Argument)

Modified Bessel Functions of the Second Kind of Integral Order with Complex Argument

Programmed by	Toshio Yoshida, June 1985
Format	Function Language: FORTRAN; Size: 149 and 211 lines respectively

(1) Outline

BESKNC (BESKNB) calculates $K_n(z)$ with single (double) precision for an integer n and single (double) precision complex number z.

(2) Directions

1. BESKNC(N, Z), BESKNB(N, Z)

COMPLEX*8 BESKNC, Z, or COMPLEX*16 BESKNB, Z should be declared for function names and the argument Z.

2. Range of argument

$|Z| \neq 0$ and

$0 \leqq \mathrm{Re}(Z) \leqq 174.673$ and

$|Im(z)| < \{8.23\times10^{5}$ (single precision) and 3.53×10^{15} (double precision)$\}$

Or

$-174.673 \leqq \mathrm{Re}(z) < 0$ and $\left|\mathrm{Im}(z)\right| \leqq 174.673$

3. Error processing

If an argument outside the range is given, an error message is output, and the calculation is continued with the function value as 0. (See FNERST.)

(3) Calculation method

From the relation K-n(z)=Kn(z), the case n<0 can be reduced to the case n≧0.

1. If Re(z)≧0, Kn(z) (n≧2) is calculated using the recurrence formula

$$K_{k+1}(z)=(2k/z)K_k(z)+K_{k-1}(z) \quad K=1,2,\cdots,n-1$$

from K0(z) and K1(z).

K0(z) and K1(z) are calculated using

$$K_0(z)=-\{\gamma+log(z/2)\}I_0(z)+2\sum_{k=1}^{\infty}I_{2k}(z)/k$$

and

$$k_1(z)=(1/z-I_1(z)K_0(z))/I_0(z)$$

when |Im(z)|<{-2.25Re(z)+4.5(single precision), -4Re(z)+8(double precision)}.

However, IK(z) is calculated using recurrence formulas. Except for the above case, the calculation is executed with the approximation formula of fn(1/z) as the form of

$$K_n(z)=\sqrt{\pi/(2z)}\,e^{-z}f_n(1/z)$$

Note that the approximation is obtained by applying τ method to the differential equation

$$t^2f_n''(t)+2(t+1)f_n'(t)-(n^2-\tfrac{1}{4})f_n(t)=0$$

that fn(t) satisfies. For details, refer to "Bibliography."

2. If Re(Z)<0, the cutting plane line of Kn(Z) should be selected for the negative real axis. Therefore, the value of Kn(z) is obtained by using the relational expression

$$K_n(z)=(-1)^nK_n(-z)-\pi iI_n(-z)$$

if Im(z)≧0, and the relational expression

104

$$K_n(z)=(-1)^n K_n(-z)+\pi i I_n(-z)$$

if $\mathrm{Im}(z)<0$.

Bibliography

1) Toshio Yoshida and Ichizo Ninomiya; "Computation of Bassel Function Kn(z) with Complex Argument by Using the τ-Method" Information Processing. Vol. 14, No. 8, pp. 569-575 (1973).

(1987. 08. 07)

BESYNC/B (Bessel Function of the Second Kind of Integral Order with Complex Argument)

Bessel Functions of the Second Kind of Integral Order with Complex Argument

Programmed by	Toshio Yoshida, June 1985
Format	Function Language: FORTRAN; Size: 45 and 50 lines respectively

(1) Outline

BESYNC (BESYNB) calculates Yn(z) for the argument n and single (double) precision complex number z with single (double) precision.

(2) Directions

1. BESYNC(N, Z), BESYNB(N, Z)

   COMPLEX*8 BESYNC, Z, or COMPLEX*16 BESYNB, Z should be declared for function names and the argument Z.

2. Range of argument

   $|z| \neq 0$

   $|\mathrm{Re}(z)| \leq 174.673$ and $|\mathrm{Im}(z)| \leq 174.673$

3. Error processing

   When an argument outside the range is given, an error message is output, and the calculation is continued with the function value as 0. (See FNERST.)

(3) Calculation method

From

$$Y_{-n}(z)=(-1)^n Y_n(z)$$

and

$$Y_n(conjg(z))=conjg(Y_n(z))$$

, the case n<0 or Im(z)<0 can be reduced to the case n>0 and Im(z)>0.

Yn(z) is obtained by using the relational expression

$$Y_n(z)=i^{n+1}I_n(-iz)-\frac{2}{\pi}i^n(-1)^nK_n(-iz)$$

Where, the value of In(-iz) is obtained by using BESINC/B, and the value of Kn(-iz) is obtained by using BESKNC/B.

(1987.08.07)

BH0/DH0/BH1/DH1 (Struve Functions of the Order 0 and 1)

Struve Functions of the Order 0 and 1

Programmed by	Ichizo Ninomiya, April 1983
Format	Function Language; FORTRAN

(1) Outline

BH0 (DH0) calculates the Struve function H_0 of the 0-th order for a single (double) precision real number x by single (double) precision.

BH1 (DH1) calculates the Struve function H_1 of the 1-st order for a single (double) precision real number x by single (double) precision.

(2) Directions

1. BH0(X), BH1(X), DH0(D), DH1(D)

X(D) is arbitrary an expression of single (double) precision real type. DH0 and DH1 require the declaration of double precision.

2. Range of argument

$0 \leqq X \leqq 8.23E+5$, $0 \leqq D \leqq 3.53D+15$

3. Error processing

If an argument outside the range is given, an error message is printed, and the calculation is continued with the function value as 0.

(See FNERST.)

(3) Calculation method

1. If $0 \leqq x \leqq 2$, $H_0(x)=x \cdot P_1(x^{**}2)$ and $H_1(x)=x^{**}2 \cdot Q_1(x^{**}2)$ are calculated with the polynomial approximations P_1 and Q_1.

2. If $2<x \leqq 4$, $H_0(x)=P_2(x-3)$ and $H_1(x)=Q_2(x-3)$ are calculated with the polynomial approximations P_2 and Q_2.

108

3. If $4<x\leqq6$, $H_0(x)=P_3(x-5)$ and $H_1(x)=Q_3(x-5)$ are calculated with the polynomial approximations P_3 and Q_3.

4. If $6<x\leqq8$, $H_0(x)=P_4(x-7)$ and $H_1(x)=Q_4(x-7)$ are calculated with the polynomial approximations P_4 and Q_4.

5. If $8<x\leqq8.25$ (3.53D+15), $L_0(x)=P_5((8/x)**2)/x+Y_0(x)$ and $L_1(x)=Q_5((8/x)**2)$ are calculated with the polynomial approximations (rational approximations) P_5 and Q_5.

(4) Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., p.228 (1970).

(1989.01.25)

BIO/DIO/QIO, BI1/DI1/QI1, BKO/DKO/QKO, and BK1/DK1/QK1 (Modified Bessel functions of order 0 and 1)

Modified Bessel Functions of the Order 0 and 1

Making	Ichizo Ninomiya; May 1983
Format	Function language; FORTRAN Size; 47, 72, 139, 50, 72, 139, 54, 73, 162, 55, 73, and 160 lines respectively

(1) Outline

BIO (DIO, QIO), BI1 (DI1, QI1), BKO (DKO, QKO), and BK1 (DK1, QK1) calculate $I_0(x)$, $I_1(x)$, $K_0(x)$, and $K_1(x)$ respectively for single (double, quadruple) precision real x with single (double, quadruple) precision.

(2) Directions

1. BIO(X), BI1(X), BKO(X), BK1(X)

   DIO(D), DI1(D), DKO(D), DK1(D)

   QIO(Q), QI1(Q), QKO(Q), QK1(Q)

   X (D, Q) is an arbitrary single (double, quadruple) precision real expression.

2. Range of argument

   $0 \leq X < 174.673$ for BIO(X), BI1(X), DIO(X), DI1(X), QIO(X), and QI1(X)

   $0 < X$ for BKO(X), BK1(X), DKO(X), DK1(X), QKO(X), and QK1(X)

3. Error processing

   If an argument outside the range is given, an error message is printed. Calculation is continued with the function value assumed to be 0. (See FNERST.)

(3) Calculation method

1. BIO and BI1 (DIO and DI1)

110

(1) When $0 \leqq x \leqq 6$, polynomial approximations A_0, A_1 are used as follows:

$$I_0(x)=A_0(x^2),\quad I_1(x)=x \cdot A_1(x^2)$$

(2) When $6 < x \leqq 8$, polynomial approximations B_0, B_1 are used as follows:

$$I_0(x)=B_0((x-6)(x+6)),\quad I_1(x)=B_1((x-6)(x+6))$$

(3) When $8 < x \leqq 174.673$, rational approximations C_0, C_1 are used as follows:

$$I_0(x)=C_0(1-8/x) \cdot e^x/\sqrt{x},\quad I_1(x)=C_1(1-8/x) \cdot e^x/\sqrt{x}$$

2. BK0 and BK1 (DK0 and DK1)

(1) When $0 < x \leqq 1$, polynomial approximations A_0, A_1, B_0, B_1 are used as follows:

$$K_0(x)=x^2A_0(x^2)+(log2-\gamma-logx) \cdot B_0(x^2),$$

$$K_1(x)=A_1(x^2)/x+(log2-\gamma-logx) \cdot x \cdot B_1(x^2),$$

(2) When $1 < x \leqq 2$, rational approximations C_0, C_1 are used as follows:

$$K_0(x)=C_0(x-1),\quad K_1(x)=C_1(x-1)$$

(3) When $2 < x \leqq 177.850$, rational approximations E_0, E_1 are used as follows:

$$K_0(x)=E_0(2/x)e^{-x}/\sqrt{x},\quad K_1(x)=E_1(2/x)e^{-x}/\sqrt{x}$$

(4) When $x > 177.850$, we put $K_0(x)=0$, $K_1(x)=0$.

3. QI0 and QI1

(1) When $0 \leqq x \leqq 12$, polynomial approximations A_0, A_1 are used as follows:

$$I_0(x)=A_0(x^2),\quad I_1(x)=x \cdot A_1(x^2)$$

(2) When $12 < x \leqq 16$, polynomial approximations B_0, B_1 are used as follows:

$I_0(x)=B_0((x-12)(x+12))$, $I_1(x)=B_1((x-12)(x+12))$

(3) When $16<x\leqq 32$, rational approximations C_0, C_1 are used as follows:

$I_0(x)=C_0(1-16/x)e^x/\sqrt{x}$, $I_1(x)=C_1(1-16/x)e^x/\sqrt{x}$

(4) When $32<x\leqq 174.673$, rational approximations E_0, E_1 are used as follows:

$I_0(x)=E_0(1-32/x)e^x/\sqrt{x}$, $I_1(x)=E_1(1-32/x)e^x/\sqrt{x}$

4. QK0 and QK1

(1) When $0<x\leqq 1$, polynomial approximations A_0, A_1, B_0, B_1 are used as follows:

$K_0(x)=A_0(x^2)-B_0(x^2)\log x$, $K_1(x)=A_1(x^2)/x+B_1(x^2)\cdot x\cdot\log x$

(2) When $1<x\leqq 2$, rational approximations C_0, C_1 are used as follows:

$K_0(x)=C_0(x-1)$, $K_1(x)=C_1(x-1)/x$

(3) When $2<x\leqq 4$, rational approximations E_0, E_1 are used as follows:

$K_0(x)=E_0(2/x)e^{-x}/\sqrt{x}$, $K_1(x)=E_1(2/x)e^x/\sqrt{x}$

(4) When $4<x\leqq 8$, rational approximations G_0, G_1 are used as follows:

$K_0(x)=G_0(4/x)e^{-x}/\sqrt{x}$, $K_1(x)=G_1(4/x)e^{-x}/\sqrt{x}$

(5) When $8<x\leqq 177.850$, rational approximations P_0, P_1 are used as follows:

$K_0(x)=P_0(8/x)e^{-x}/\sqrt{x}$, $K_1(x)=P_1(8/x)e^{-x}/\sqrt{x}$

(6) When $x>177.850$, we put $K_0(x)=0$, $K_1(x)=0$.

Bibliography

112

1) Handbook of Mathematical Functions, Dover, N.Y., p.374.

(1987.07.06) (1987.08.12) (1987.08.24)

BIOI0/DIOI0, BIOI1/DIOI1, BKOI0/DKOI0, BKOI1/DKOI1 (Integrals of modified Bessel functions)

Integrals of Modified Bessel Functions

Programmed by	Ichizo Ninomiya: August 1978
Format	Function Language; FORTRAN Size; 39, 67, 40, 71, 44, 66, 42, and 64 lines respectively

(1) Outline

$$\begin{cases} BIOI0(DIOI0) \\ BIOI1(DIOI1) \\ BKOI0(DKOI0) \\ BKOI1(DKOI1) \end{cases} \qquad \begin{cases} \int_0^x I_0(t)dt \\ \int_0^x \frac{I_0(t)-1}{t}dt \\ \int_0^x K_0(t)dt \\ \int_x^\infty \frac{K_0(t)}{t}dt \end{cases}$$

Each function routine calculates the definite integral corresponding to single or double precision real number x, with single or double precision respectively.

(2) Directions

1. BIOI0(X), BIOI1(X), BKOI0(X), BKOI1(X)

   DIOI0(D), DIOI1(D), DKOI0(D), DKOI1(D)

   X and D are arbitrary single and double precision real-type expressions respectively. The double precision function name needs to be declared as double precision.

2. Range of argument

   BIOI0 and BIOI1: $0 \leqq X \leqq 174.673$

   DIOI0 and DIOI1: $0 \leqq D \leqq 174.673$

   BKOI0 and BKOI1: $0 \leqq X$

   DKOI0 and DKOI1: $0 \leqq D$

3. Error processing

   If the specified argument is outside the range, an error message is printed but calculation continues with the function assumed to be 0. (See FNERST.)

114

(3) Calculation method

1. For BI0I0, DI0I0, BI0I1, and DI0I1

(1) In case of $x<0$ or $x>174.673$, an error results.

(2) In case of $0\leqq x\leqq 16$, polynomial approximations P_0, P_1 are used to calculate:

$$\int_0^x I_0(t)dt=xP_0(x^2)$$

$$\int_0^x \frac{I_0(t)-1}{t}dt=P_1(x^2)$$

(3) In case of $16<x\leqq 174.673$, rational approximations R_0, R_1 are used to calculate:

$$\int_0^x I_0(t)dt=\frac{e^x}{\sqrt{x}}R_0\left(\frac{x-16}{x}\right)$$

$$\int_0^x \frac{I_0(t)-1}{t}dt=\frac{e^x}{x^{3/2}}R_1\left(\frac{x-16}{x}\right)-\log\frac{x}{16}$$

2. For BK0I0 and DK0I0

(1) In case of $x<0$, an error results.

(2) In case of $0\leqq x\leqq 2$, polynomial approximations P_0, Q_0 are used to calculate:

$$\int_0^x K_0(t)dt=xP_0(x^2)+xQ_0(x^2)\log\frac{x}{2}$$

(3) In case of $2<x\leqq 180.218$, rational approximations R_0 are used to calculate:

$$\int_0^x K_0(t)dt=\frac{\pi}{2}-\frac{e^{-x}}{\sqrt{x}}R_0\left(\frac{2}{x}\right)$$

(4) In case of $x>180.218$, $\int_0^x K_0(t)dt=\pi/2$.

3. For BK0I1 and DK0I1

(1) In case of $x\leqq 0$, an error results.

(2) In case of $0<x\leq 2$, polynomial approximations P_1, Q_1 are used to calculate:

$$\int_x^\infty \frac{K_0(t)}{t}dt=\left(P_1(x^2)+\frac{1}{2}\log\frac{x}{2}\right)\cdot\log\frac{x}{2}+Q_1(x^2)$$

(3) In case of $2<x\leq 180.218$, rational approximation R_1 is used to calculate:

$$\int_x^\infty \frac{K_0(t)}{t}dt=\frac{e^{-x}}{x^{3/2}}R_1\left(\frac{2}{x}\right)$$

(4) In case of $x\geq 180.218$,

$$\int_x^\infty \frac{K_0(t)}{t}dt=0$$

(1989. 01. 13)

116

BIOML0/DIOML0/BI1ML1/DI1ML1

(Modified Bessel Functions Minus Modified Struve Function (Order 0, 1))

Modified Bessel Function Minus Modified Struve Function (Order 0, 1)

Programmed by	Ichizo Ninomiya, November 1983
Format	Function Language; FORTRAN

(1) Outline

BIOML0 (DIOML0) obtains the difference between the 0-th order modified Bessel function I0 and the 0-th order modified Struve function L0 for the single (double) precision real number x with single (double) precision. BI1ML1 (DI1ML1) calculates the difference between the 1-st order modified Bessel function I1 and the 1-st order modified Struve function L1 for the single (double) precision real number x with single (double) precision.

(2) Directions

1. BIOML0(X), BI1ML1(X), DIOML0(D), DI1ML1(D), and X(D) are arbitrary expressions of single (double) precision real type. DIOML0 and DI1ML1 require the declaration of double precision.
2. Range of argument $0 \leqq X$, $0 \leqq D$
3. Error processing

If an argument outside the range is given, an error message is printed, and the calculation is continued with the function value as 0. (See FNERST.)

(3) Calculation method

1. If $0 \leqq x \leqq 8$, $I0(x)-L0(x)=R1(x)$ and $I1(x)-L1(x)= x \cdot S1(x)$ are calculated with the rational approximations R1 and S1.

2. If $8 \leqq x \leqq 16$, $I0(x)-L0(x)=R2(x-8)$ and $I1(x)-L1(x)=S2(x)$ are calculated with the rational approximations R2 and S2.

3. If $x>16$, $I0(x)-L0(x)=R3((16/x)**2)$ and $I1(x)=L1(x)=S3((16/x)**2)/x$ are calculated with the

polynomial approximations R3 and S3.

(4) Note

This function program should be used to calculate the difference between modified Bessel and Struve functions. If the difference is found by calculating the modified Bessel and Struve functions separately, precision cannot be attained because of a severe cancellation.

(5) Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., p.498(1970).

(1987.08.07)

118

BIF/DIF (Modified Bessel functions of the first kind of fractional orders)

Modified Bessel Functions of the First Kind of Fractional Orders

Programmed by	Ichizo Ninomiya; September 1981
Format	Function Language; FORTRAN Size; 129 and 188 lines respectively

(1) Outline

BIF (DIF) calculates $I_u(x)$ with single (double) precision for single (double) precision real numbers u and x.

(2) Directions

1. BIF(U, X) and DIF(W, D)

   U and X are arbitrary single precision real-type expressions. W and D are arbitrary double precision real-type expressions. DIF needs to be declared as double precision.

2. Range of argument

   $0 \leqq U (0 \leqq W)$, $0 \leqq X$, $(0 \leqq D)$ However, any region where the function value overflows is excluded.

3. Error processing

   If the specified argument is outside the range, an error message is printed but calculation continues with the function value assumed to be 0. (See PNERST.)

(3) Calculation method

1. BIF(DIF)

(1) In case of $x<0$ or $u<0$, an error results.

(2) In case of $(x/2)^2 \leqq u+1$, the following Taylor series is calculated:

$$I_u(x)=\sum_{k=0}^{\infty}\frac{(x/2)^{u+2k}}{k!\Gamma(u+k+1)}$$

(3) In case of $x \geqq 10(18)$ and $x \geqq 0.55u^2$, the following asymptotic expansion is used:

$$I_u(x)=\frac{e^x}{\sqrt{2\pi x}}\left\{1-\frac{4u^2-1}{8x}+\frac{(4u^2-1)(4u^2-9)}{2!(8x)^2}\right.$$

$$-\frac{(4u^2-1)(4u^2-9)(4u^2-25)}{3!(8x)^3}+\cdots\cdots\}$$

(4) In case of $u \geqq 10(35)$ in a region other than (2) and (3), the following uniform asymptotic expansion is used:

$$I_u(x)=\frac{1}{\sqrt{2\pi u}}\frac{e^{u\eta}}{(1+\xi^2)^{1/4}}\{1+\sum_{k=1}^{\infty}u^{-k}u_k(t)\}$$

This calculation is based on the following:

$$\xi=x/u,\ t=1/\sqrt{1+\xi^2},\quad \eta=\sqrt{1+\xi^2}+\log[\ \xi/(1+\sqrt{1+\xi^2})\ ]$$

where $u_k(t)$ is a kth order polynomial of t.

(5) In case of an area other than the above, the following recurrence formula is used:

$$I_{v-1}(x)=2vI_v(x)/x+I_{v+1}(x)$$

(4) Note

Both BIF and DIF involve a great amount of calculation and are time consuming. So, it is better not to use these function programs for those function that can be calculated in another method. For instance, $I_0(x)$ can be calculated by using any of BI0(X), BIN(0, X), and BIF(0.0, X). Among them, however, BI0(X) is the fastest in calculation with better precision. Generally speaking, BI0 and BI1 (DI0 and DI1) should be used to calculate modified Bessel functions of orders 0 and 1; BIN (DIN) should be used for other functions of integral orders. As for Bessel functions of half odd order, it is better to calculate them via the spherical Bessel function. For $I_{5/2}(x)$, for instance, it is more reasonable to calculate SIK(2, X) and multiply it by $\sqrt{2x/\pi}$ rather than to calculate BIF(2.5, X).

Bibliography

1) Ichizo Ninomiya; "Calculation of Bessel functions by recurrence formula", Numerical method II for computers, Baifu-kan (1966).

120

2) D. E. Amos, S. L. Daniel and M. K. Weston; "CDC6600 Subroutines IBESS and JBESS for Bessel Functions $Iv(x)$ and $Jv(x)$, $x \geq 0, v \geq 0$, ACM Trans. on Math. Software, Vol. 3, No. 1, pp. 76-92 (1977).

(1989. 01. 25)

BIN/DIN, BKN/DKN (Modified Bessel functions of integer orders)

Modified Bessel Functions of Integral Orders

Programmed by	Ichizo Ninomiya; September 1981
Format	Function Language; FORTRAN Size; 123, 183, 49, and 50 lines respectively

(1) Outline

BIN (DIN) and BKN (DKN) calculate $I_n(x)$ and $K_n(x)$ respectively, with single (double) precision, for an integer n and a single (double) precision real number x.

(2) Directions

1. BIN(N, X), DIN(N, D), BKN(N, X), and DKN(N, D)

   N is an arbitrary integer-type expression. X and D are arbitrary single (double) precision real-type expressions respectively. The double precision function name needs to be declared as double precision.

2. Range of argument

   Modified Bessel function of the first kind $I_n(x)$: $0 \leqq N$, $0 \leqq X$ ($0 \leqq D$)

   However, any region where the function value overflows is excluded.

   Modified Bessel function of the second kind $K_n(x)$: $0 \leqq N$, $0 < X$ ($0 < D$)

   However, any region where the function value overflows is excluded.

3. Error processing

   If the specified argument is outside the range, an error message is printed but calculation continues with the function value assumed to be 0. (See PNERST.)

(3) Calculation method

1. BIN (DIN)

   (1) In case of $x<0$ or $n<0$, an error results.

(2) In case of $(x/2)^2 \leq n+1$, the following Taylor series is calculated:

$$I_n(x)=\sum_{k=0}^{\infty}\frac{(x/2)^{n+2k}}{k!\,(n+k)!}$$

(3) In case of $x \geq 10(18)$ and $x \geq 0.55n^2$, the following asymptotic expansion is used:

$$I_n(x)=\frac{e^x}{\sqrt{2\pi x}}\left\{1-\frac{4n^2-1}{8x}+\frac{(4n^2-1)(4n^2-9)}{2!\,(8x)^2}\right.$$

$$\left.-\frac{(4n^2-1)(4n^2-9)(4n^2-25)}{3!\,(8x)^3}+\cdots\cdots\right\}$$

(4) In case of $n \geq 10(35)$ in a region other than (2) and (3), the following uniform asymptotic expansion is used:

$$I_n(x)=\frac{1}{\sqrt{2\pi n}}\frac{e^{n\eta}}{(1+\xi^2)^{1/4}}\left\{1+\sum_{k=1}^{\infty}n^{-k}u_k(t)\right\}$$

This calculation is based on the following:

$$\xi=x/n,\ t=1/\sqrt{1+\xi^2},\quad \eta=\sqrt{1+\xi^2}+\log\left[\ \xi/(1+\sqrt{1+\xi^2})\ \right]$$

where $u_k(t)$ is a kth order polynomial of t.

(5) In case of an area other than the above, the following recurrence formula is used:

$$I_{k-1}(x)=2kI_k(x)/x+I_{k+1}(x)$$

2. BKN (DKN)

(1) In case of $n<0$ or $x \leq 0$, an error results.

(2) In case of $x>180.218$, $K_n(x)=0$ is assumed.

(3) In case of $n=0$, BK0 (DK0) is called to calculate $K_0(x)$.

(4) In case of $n=1$, BK1 (DK1) is called to calculate $K_1(x)$.

(5) In case of $n \geq 2$, the following recurrence formula is repeatedly calculated, starting with $K_0(x)$, $K_1(x)$, to obtain $K_n(x)$:

$$K_{k+1}(x)=2kK_k(x)\frac{1}{x}+K_{k-1}(x)$$

(4) Note

For calculation of modified Bessel functions of order 0 or 1, the functions specific to them are superior in calculation time and precision to the above functions. For instance, BI0(X) is better than DIN(0, X).

Bibliography

1) Ichizo Ninomiya; "Calculation of Bessel functions by recurrence formula", Numerical method II for computers, Baifu-kan (1966).

2) D. E. Amos et al.; "CDC 6600 Subroutines IBESS and JBESS for Bessel Functions $I\upsilon(x)$ and $J\upsilon(x), x\geq 0, \upsilon\geq 0$", ACM Trans. on Math. Software, Vol. 3, No. 1, pp. 76-92 (1977).

(1989. 01. 25)

124

BJ0/DJ0/QJ0, BJ1/DJ1/QJ1, BY0/DY0/QY0, BY1/DY1/QY1

BJ2/DJ2, BJ3/DJ3, and BJ4/DJ4 (Bessel function of the order 0 and 1)

Bessel Functions of the Order 0 and 1

Programmed	Ichizo Ninomiya May, 1983
Format	Function Language; FORTRAN Size; 76, 112, 248, 74, 112, 248, 83, 132, 262, 82, 132, 261 lines

(1) Outline

Given real number x with single (double, quadruple) precision, BJ0(DJ0, QJ0), BJ1(DJ1, QJ1), BY0(DY0, QY0) and BY1(DY1, QY1) calculate $J_0(x)$, $J_1(x)$, $Y_0(x)$ and $Y_1(x)$ in single (double, quadruple) precision respectively.

Similary BJ2(DJ2), BJ3(DJ3), and BJ4(DJ4) calculate $J_2(x)$, $J_3(x)$, and $J_4(x)$ respectively.

(2) Directions

1. BJ0(X), BJ1(X), BY0(X), and BY1(X)

   DJ0(D), DJ1(D), DY0(D), DY1(D)

   QJ0(Q), QJ1(Q), QY0(Q), QY1(Q)

   X(D, Q) is an arbitrary expression of real type with single (double, quadruple) precision. The function name of double (quadruple) precision needs the declaration of the double (quadruple) precision.

   BJ2(DJ2) etc. are the same as BJ0(DJ0).

2. Range of argument

For the first kind Bessel function, $0 \leq X \leq 8.23 \cdot 10^5$, $0 \leq D \leq 3.53 \cdot 10^{15}$, $0 \leq Q \leq 10^{30}$.

For the second kind Bessel function, $0 < X \leq 8.23 \cdot 10^5$, $0 < D \leq 3.53 \cdot 10^{15}$, $0 < Q \leq 10^{30}$.

3. Error processing

   When the argument is outside the range, it is handled as an error, and the message is printed. The calculation continues with the function value as 0. (Refer to FNERST)

(3) Calculation method

1. Calculation methods for BJ0, BJ1, BY0, BY1(DJ0, DJ1, DY0, DY1), and BJ2(DJ2)

(1) If $0 \leqq x \leqq 2$ ($0 < x \leqq 2$ for the second kind of functions), calculation is done using polynomial approximation $A_{ji}, i=0, \cdots, 4, A_{y0}, A_{y1}, P_0, P_1$ as follows:

$$J_i(x)=x^i A_{ji}(x^2), i=0,1,\cdots,4$$

$$Y_0(x)=A_{y0}(x^2) + P_0(x^2)\log x,$$

$$Y_1(x)=x \cdot A_{y1}(x^2) - 2/(\pi x) + P_1(x^2)\log x,$$

(2) If $2 < x \leqq 4$, calculation is done using polynomial approximation $B_{ji}, i=0, \cdots, 4, B_{y0}, B_{y1}$ as follows:

$$J_i(x)=B_{ji}(x-3), i=0,1,\cdots,4$$

$$Y_0(x)=B_{y0}(x-3),\ Y_1(x)=B_{j1}(x-3)$$

(3) If $4 < x \leqq 6$, calculation is done using polynomial approximation $C_{ji}, i=0,1,\cdots,4, C_{y0}, C_{y1}$ as follows:

$$J_i(x)=C_{ji}(x-5)$$

$$Y_0(x)=C_{y0}(x-5),\ Y_1(x)=C_{y1}(x-5)$$

(4) If $6 < x \leqq 8$, calculation is done using polynomial approximation $D_{ji}, i=0,1,\cdots,4, D_{y0}, D_{y1}$ as follows:

$$J_i(x)=D_{ji}(x-7)$$

$$Y_0(x)=D_{y0}(x-7),\ Y_1(x)=D_{y1}(x-7)$$

(5) If $8 < x \leqq 8.23 \cdot 10^5 (3.53 \cdot 10^{15})$, calculation is done using polynomial (rational) approximation $E_i, G_i, i=0,1,\cdots,4$ as follows:

126

$$R_i(x)=E_i(y^2),\ \varphi_i(x)=yG_i(y^2),$$

$$J_i(x)=\sqrt{yR_i}\cos(\varphi_i+\frac{\pi i}{2}-\frac{\pi}{4}+x),\ i=0,1,\cdots,4$$

$$Y_i(x)=\sqrt{yR_i}\sin(\varphi_i+\frac{\pi i}{2}-\frac{\pi}{4}+x),\ i=0,1$$

Where, $y=8/x$.

2. Calculation method of QJ0, QJ1, QY0, and QY1

(1) If $0\leq x\leq 2$ ($0<x\leq 2$ for the second kind of functions), calculation is done using polynomial approximation $A_{j0}, A_{j1}, A_{y0}, A_{y1}$ as follows:

$$J_0(x)=A_{j0}(x^2),\ J_1(x)=xA_{j1}(x^2),$$

$$Y_0(x)=A_{y0}(x^2)+\frac{2}{\pi}J_0(x)\log x,$$

$$Y_1(x)=xA_{y1}(x^2)+\frac{2}{\pi}(J_1(x)\log x-1/x)$$

(2) In the range of $2<x\leq 12$, four functions are calculated by using the polynomial approximation which centers on each middle point of five intervals
$2<x\leq 4$, $4<x\leq 6$, $6<x\leq 8$, $8<x\leq 10$, $10<x\leq 12$.

(3) If $12<x\leq 10^{30}$, calculation is done using rational approximation P_0, P_1, S_0, S_1 as follows:

$$R_0(x)=P_0(y^2),\ R_1(x)=P_1(y^2),$$

$$\varphi_0(x)=yS_0(y^2)\ -\frac{\pi}{4}+x,\ \varphi_1(x)=yS_1(y^2)-\frac{\pi}{4}+x$$

$$J_0(x)=\sqrt{yR_0}\cos\varphi_0$$

$$Y_0(x)=\sqrt{yR_0}\sin\varphi_0$$

$$J_1(x)=\sqrt{yR_1}\sin\varphi_1$$

$$Y_1(x)=-\sqrt{yR_1}\cos\varphi_1$$

Where, $y=12/x$.

(4) Note

The second kind Bessel function $N_0(x), N_1(x)$ is the same as $Y_0(x), Y_1(x)$ respectively.

Bibliography

1) Handbook of Mathematical Functions, Dover, N. Y., p. 358

2) J. F. Hart; "Computer Approximations" and J. Wiley (1968).

(1989. 01. 25)

128

BJOI0/DJOI0, BJOI1/DJOI1, BYOI0/DYOI0, BYOI1/DYOI1 (Integrals of Bessel functions)

Integrals of Bessel Functions

Programmed by	Ichizo Ninomiya: January 1978
Format	Function Language; FORTRAN Size; 46, 78, 43, 79, 55, 93, 52, and 94 lines respectively

(1) Outline

$$\begin{cases} BJOI0\,(DJOI0) \\ BJOI1\,(DJOI1) \\ BYOI0\,(DYOI0) \\ BYOI1\,(DYOI1) \end{cases} \qquad \begin{cases} \displaystyle\int_0^x J_0(t)dt \\ \displaystyle\int_x^\infty \frac{J_0(t)}{t}dt \\ \displaystyle\int_0^x Y_0(t)dt \\ \displaystyle\int_x^\infty \frac{Y_0(t)}{t}dt \end{cases}$$

Each function routine calculates the definite integral corresponding to single or double precision real number x, with single or double precision respectively.

(2) Directions

1. BJOI0(X), BJOI1(X), BYOI0(X), BYOI1(X)

   DJOI0(D), DJOI1(D), DYOI0(D), DYOI1(D)

   X and D are arbitrary single and double precision real-type expressions respectively. The double precision function name needs to be declared as double precision.

2. Range of argument

   BJOI0 and BYOI0: $0 \leqq X \leqq 8.23 \cdot 10^5$.

   DJOI0 and DYOI0: $0 \leqq D \leqq 3.53 \cdot 10^{15}$.

   BJOI1 and BYOI1: $0 < X \leqq 8.23 \cdot 10^5$.

   DJOI1 and DYOI1: $0 < D < 3.53 \cdot 10^{15}$.

3. Error processing

If the specified argument is outside the range, an error message is printed but calculation continues with the function value assumed to be 0. (See FNERST.)

(3) Calculation method

1. In case of $0<x\leqq 4$, polynomial approximations $P_0, Q_0, R_0, P_1, Q_1, R_1$ are used to calculate:

$$\int_0^x J_0(t)dt = xP_0(x^2)$$

$$\int_0^x Y_0(t)dt = xQ_0(x^2) + xR_0(x^2)\log\frac{x}{2}$$

$$\int_x^\infty \frac{J_0(t)}{t}dt = P_1(x^2) - \log\frac{x}{2}$$

$$\int_x^\infty \frac{Y_0(t)}{t}dt = \left(Q_1(x^2) - \frac{1}{\pi}\log\frac{x}{2}\right)\log\frac{x}{2} + R_1(x^2)$$

2. In case of $4<x<8.23\cdot 10^5$, rational approximations S_0, T_0, S_1, T_1 are used to calculate:

$$\int_0^x J_0(t)dt = \frac{1}{\sqrt{x}}\left(S_0\left(\frac{4}{x}\right)\cos\left(x-\frac{\pi}{4}\right) - T_0\left(\frac{4}{x}\right)\sin\left(x-\frac{\pi}{4}\right)\right) + 1$$

$$\int_0^x Y_0(t)dt = \frac{1}{\sqrt{x}}\left(S_0\left(\frac{4}{x}\right)\sin\left(x-\frac{\pi}{4}\right) + T_0\left(\frac{4}{x}\right)\cos\left(x-\frac{\pi}{4}\right)\right)$$

$$\int_x^\infty \frac{J_0(t)}{t}dt = x^{-\frac{3}{2}}\left(S_1\left(\frac{4}{x}\right)\cos\left(x-\frac{\pi}{4}\right) - T_1\left(\frac{4}{x}\right)\sin\left(x-\frac{\pi}{4}\right)\right)$$

$$\int_x^\infty \frac{Y_0(t)}{t}dt = x^{-\frac{3}{2}}\left(S_1\left(\frac{4}{x}\right)\sin\left(x-\frac{\pi}{4}\right) - T_1\left(\frac{4}{x}\right)\cos\left(x-\frac{\pi}{4}\right)\right)$$

(1989.01.17)

130

BJF/DJF (Bessel functions of the first kind of fractional orders)

Bessel Functions of the First Kind of Fractional Orders

Making	Ichizo Ninomiya; September 1981
Format	Function language; FORTRAN Size; 100 lines each

(1) Outline

BJF (DJF) calculates $J_u(x)$ for single (double) precision real numbers u,x with single (double) precision.

(2) Directions

1. BJF (U, X) and DJF (W, D)

$U(W),X(D)$ is an arbitrary single (double) precision real expression. DJF requires declaration for double precision.

2. Range of argument

$U \geqq 0 (W \geqq 0), 0 \leqq X \leqq 8.23 \cdot 10^5 (0 \leqq D \leqq 3.53 \cdot 10^{15})$

excluding $X>200, U>1.384\sqrt{X}$ $(D>200, W>1.384\sqrt{D})$.

3. Error processing

If an argument outside the range is given, an error message is printed. Calculation is continued with the function value assumed to be 0. (See FNERST.)

(3) Calculation method

1. BJF(DJF)

(1) When $u<0$, $x<0$, or $x>8.23 \cdot 10^5 (3.53 \cdot 10^{15})$, error processing starts.

(2) When $(x/2)^2 \leqq u+1$, Taylor series

$$J_u(x)=\sum_{k=0}^{\infty} \frac{(-1)^k (x/2)^{u+2k}}{k!\Gamma(u+k+1)}$$

is calculated.

(3) If $x \geqq 10(18)$ and $x \geqq 0.55u^2$, asymptotic expansion

$$J_u(x)=\sqrt{2/\pi x}\,\{P(u,x)\cos\phi-Q(u,x)\sin\phi\}$$

is used.

where

$$\phi=x-(u/2+1/4)\pi,$$

$$P(u,x)=1-\frac{(4u^2-1)(4u^2-9)}{2!(8x)^2}+\cdots,$$

$$Q(u,x)=\frac{4u^2-1}{8x}-\frac{(4u^2-1)(4u^2-9)(4u^2-25)}{3!(8x)^3}+\cdots$$

(4) For an area which is not covered by (2) and (3) but satisfies $x\leqq 200$, the following recurrence equation is used:

$$J_{v-1}(x)=2vJ_v(x)/x-J_{v+1}(x)$$

(5) Any other area causes an error.

(4) Note

BJF and DJF each require a large amount of calculation and take time. Therefore, these function programs should not be used for a function which can be calculated by other methods. For instance, $J_0(x)$ can be calculated by any of BJ0(X), BJN(0, X), and BJF(0.0, X), but BJ0(X) is the best in both speed and precision. Generally, BJ0 or BJ1 (DJ0, DJ1) should be used to calculate 0-th and 1st order Bessel functions. Bessel functions of half odd number order should be calculated via a spherical Bessel function. For $J_{5/2}(x)$, for instance, it is more reasonable to multiply SJN(2, X) by $\sqrt{2x/\pi}$ than to calculate BJF(2.5, X).

Bibliography

1) Ichizo Ninomiya; "Calculation of Bessel functions by recurrence formulas," Numerical Analysis II for Computer, Baifuukan (1966)

(1987.07.07)

132

BJN/DJN, BYN/DYN (Bessel functions of integral orders)

Bessel Functions of Integral Orders

Programmed by	Ichizo Ninomiya: September 1981
Format	Function Language; FORTRAN Size; 91, 91, 46, and 47 lines respectively

(1) Outline

BJN (DJN) and BYN (DYN) calculate $J_n(x)$ and $Y_n(x)$ respectively, with single (double) precision, for an integer n and a single (double) precision real number x.

(2) Directions

1. BJN(N, X), DJN(N, D), BYN(N, X), and DYN(N, D)

N is an arbitrary integer-type expression. X and D are arbitrary single and double precision real-type expressions respectively. The double precision function name needs to be declared as double precision.

2. Range of argument

Bessel function of the first kind $J_n(x)$: $N \geq 0, 0 \leq X \leq 8.23 \cdot 10^5 (0 \leq D \leq 3.53 \cdot 10^{15})$

$X > 200, N > 1.384\sqrt{X}$ $(D > 200, N > 1.384\sqrt{D})$ are excluded.

Bessel function of the second kind $Y_n(x)$: $N \geq 0, 0 < X \leq 8.23 \cdot 10^5 (0 < D \leq 3.53 \cdot 10^{15})$

Any region where the function value overflows is excluded.

3. Error processing

If the specified argument is outside the range, an error message is printed but calculation continues with the function value assumed to be 0. (See FNERST.)

(3) Calculation method

1. BJN(DJN)

(1) In case of $n<0$, $x<0$, or $x>8.23 \cdot 10^5 (3.53 \cdot 10^{15})$, an error results.

(2) In case of $0 \leqq n \leqq 4$, special routines BJ0 (DJ0), BJ1 (DJ1), BJ2 (DJ2), BJ3 (DJ3), or BJ4 (DJ4) are called.

(3) In case of $(x/2)^2 \leqq n+1$, Taylor series

$$J_n(x)=\sum_{k=0}^{\infty}\frac{(-1)^k(x/2)^{n+2k}}{k!\cdot(n+k)!}$$

is calculated.

(4) In case of $x \geqq 10(18)$ and $x \leqq 0.55n^2$, the following asymptotic expansion is used:

$$J_n(x)=\sqrt{2/\pi x}\ \{P(n,x)\cos\phi - Q(n,x)\sin\phi\}$$

where

$$\phi = x-(n/2+1/4)\pi,$$

$$P(n,x)=1-\frac{(4n^2-1)(4n^2-9)}{2!(8x)^2}+\cdots\cdots\cdots,$$

$$Q(n,x)=\frac{4n^2-1}{8x}-\frac{(4n^2-1)(4n^2-9)(4n^2-25)}{3!(8x)^3}+\cdots\cdots\cdots$$

(5) In case of $x \leqq 200$ in a region other than (3) and (4), the following recurrence formula is used:

$$J_{k-1}(x)=2kJ_k(x)/x-J_{k+1}(x)$$

(6) In case of a region other than the above, an error results.

2. BYN (DYN)

(1) In case of $n<0$, $x \leqq 0$, or $x>8.23\cdot10^5(3.53\cdot10^{15})$, an error results.

(2) In case of $n=0$, BY0 (DY0) is called to calculate $Y_0(x)$.

(3) In case of $n=1$, BY1 (DY1) is called to calculate $Y_1(x)$.

(4) In case of $n \geqq 2$, the following recurrence formula is calculated repeatedly starting with $Y_0(x), Y_1(x)$ to determine $Y_n(x)$:

$$Y_{k+1}(x)=2kY_k(x)/x - Y_{k-1}(x)$$

(4) Note

The Bessel function of the second kind $N_n(x)$ is the same as $Y_n(x)$.

139

Bibliography

1) Ichizo Ninomiya; "Calculation of Bessel functions by recurrence formula", Numerical method II for computers, Baifukan (1966).

(1989. 01. 25)

BKF/DKF (Modified Bessel Functions of the Second Kind of Fractional Order)

Modified Bessel Functions of the Second Kind of Fractional Order

Programmed by	Toshio Yoshida, June 1985
Format	Function Language; FORTRAN; Size; 786 and 1508 lines respectively

(1) Outline

BKF (DKF) calculates $K_\nu(x)$ for the single (double) precision real numbers ν and x with single (double) precision.

(2) Directions

1. BKF(V, X), DKF(W, D)

   V and W correspond to ν, and X and D correspond to x.

   V and X (W, D) are expressions of single (double) precision real type. DKF requires the declaration of double precision.

2. Range of argument

   X>0 (D>0)

   However, the region where function values overflow is excluded.

   W>0 0<D<3.53×10

3. Error processing

   If an argument outside the range is given, an error message is output, and the calculation is continued with the function value as 0. (See FNERST.)

(3) Calculation method

$K_\nu(x)$ should be defined as

$$K_\nu(x)=\frac{\pi}{2}\cdot\frac{I_{-\nu}(x)-I_\nu(x)}{\sin\nu\pi}$$

The case $\nu>0$ can be reduced to the case $\nu<0$ from the relation $K_{-\nu}(x)=K_\nu(x)$.

136

In this method, the value of $K\nu(x)$ is directly calculated at $0 \leqq \nu \leqq 2.5$.

At $\nu > 2.5$, the value of $K\nu(x)$ is calculated with the recurrence formula

$$K_{\nu+1}(x)=\frac{2\nu}{x}K_{\nu}(x)+K_{\nu-1}(x)$$

The calculation method of $K\nu(x)$ at $0 \leqq \nu \leqq 2.5$ depends on the value of x. If x is small, the calculation is executed with the previous definition formula of $K\nu(x)$. However, the calculation is executed so that the number of significant digits is not reduced even if ν is near the integer. For details, see 1) in "Bibliography." If the value of x is large, the calculation is executed with the approximation to $f\nu(1/x)$ in the form of

$$K_{\nu}(x)=\sqrt{\frac{\pi}{2x}}e^{-x}f_{\nu}(\frac{1}{x})$$

However, the approximation formula is obtained by applying τ method to the differential equation

$$t^2f_N''(t)+2(t+1)f_{\nu}'(t)-(\nu^2-\frac{1}{4})f_{\nu}(t)=0$$

that $f\nu(t)$ satisfies.

For details, see 2) in "Bibliography."

Bibliography

1) Toshio Yoshida and Ichizo Ninomiya: "Computation of Modified Bessel function $K\nu(x)$ with Small Argument x." Transactions of Information Processing Soc. of Japan, Vol. 21, No. 3, pp. 238-245 (1980).

2) Toshio Yoshida and Ichizo Ninomiya: "Computation of Modified Bessel function $K\nu(x)$ with Large

Argument x." Transactions of Information Processing Soc. of Japan, Vol. 22, No. 4, pp. 312-319 (1981).

(1987. 08. 07)

138

BL0/DL0/BL1/DL1 (Modified Struve Functions of the Order 0 and 1)

Modified Struve Functions of the Order 0 and 1

Programmed by	Ichizo Ninomiya, April 1983
Format	Function Language; FORTRAN

(1) Outline

BL0 (DL0) calculates the modified Struve function L_0 of the 0-th order for a single (double) precision real number x with single (double) precision.

BL1 (DL1) calculates the modified Struve function L_1 of the 1-st order for a single (double) precision real number x with single (double) precision.

(2) Directions

1. BL0(X), BL1(X), DL0(D), DL1(D)

   X(D) is arbitrary an expression of single (double) precision real type. DL0 and DL1 require the declaration of double precision.

2. Range of argument

   $0 \leq X,\ D \leq 174.673$

3. Error processing

   If an argument outside the range is given, an error message is printed, and the calculation is continued with the function value as 0.

(See FNERST.)

(3) Calculation method

1. If $0 \leq x \leq 16$, $L_0(x)=x \cdot P_1(x^{**}2)$ and $L_1(x)=x^{**}2 \cdot Q_1(x^{**}2)$ are calculated with the polynomial approximations P_1 and Q_1.

2. If $16 < x \leq 174.673$, $L_0(x)=P_2((16/x)^{**}2)/x+I_0(x)$ and $L_1(x)=Q_2((16/x)^{**}2)+I_1(x)$ are calculated with the polynomial approximations P_2 and Q_2.

(4) Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., p.498 (1970).

(1989.01.25)

140

BYF/DYF (Bessel Functions of the Second Kind of Fractional Order)

Bessel Functions of the Second Kind of Fractional Order

Programmed by	Toshio Yoshida, June 1985
Format	Function Language: FORTRAN; Size: 713 and 1061 lines respectively

(1) Outline

BYF (DYF) calculates $Y_\nu(x)$ for the single (double) precision real numbers ν and x with single (double) precision.

(2) Directions

1. BYF(V, X), DYF(W, D)

   V and W correspond to ν, and X and D correspond to x.

   V and X(W, D) are expressions of single (double) precision real type. DYF requires the declaration of double precision.

2. Range of argument

   V>0 $0<X<8.23\times10^5$

   W>0 $0<D<3.53\times10^{15}$

3. Error processing

   If an argument outside the range is given, an error message is output, and the calculation is continued with the function value as 0. (See FNERST.)

(3) Calculation method

$Y_\nu(x)$ should be defined as

$$Y_\nu(x)=\frac{J_\nu(x)\cos\nu\pi-J_{-\nu}(x)}{\sin\nu\pi}$$

In this calculation method, the value of $Y_\nu(x)$ is directly calculated at $0\leqq\nu\leqq2.5$.

At $\nu>2.5$, the value of $Y_\nu(x)$ is calculated with the recurrence relation

$$Y_{\nu+1}(x)=\frac{2\nu}{x}Y_{\nu}(x)-Y_{\nu-1}(x)$$

The calculation method of $Y\nu(x)$ at $0\leqq\nu\leqq2.5$ depends on the value of x. If x is small, the calculation is executed with the previous definition formula of $Y\nu(x)$. However, the calculation is executed so that the number of significant digits is not reduced even if ν is near the integer. For details, see bibliography [1]. If x is large, $Y\nu(x)$ is calculated by the imaginary part of the approximation to the first kind Hankel function $H\nu^{(1)}(x)$ that is obtained by applying τ method to the differential equation that $H\nu^{(1)}(x)$ satisfies.

For details, see Bibliography [2].

Bibliography

1) Toshio Yoshida and Ichizo Ninomiya: "Computation of Bessel Function $Y\nu(x)$ for Small Argument x." Transactions of Information Processing Soc. of Japan, Vol. 23, No. 3, pp. 296-303 (1982).

2) Toshio Yoshida and Ichizo Ninomiya: "Computaion of Bessel Function $Y\nu(x)$ for Large Argument x." Transactions of Information Processing Soc. of Japan, Vol. 24, No. 4, pp. 436-443 (1983).

(1989. 01. 25)

142

JOYOS/D and J1Y1S/D (Bessel functions of order 0 and 1)

Bessel Functions of the Order 0 and 1

Programmed by	Ichizo Ninomiya; May 1983
Format	Subroutine Language; FORTRAN77 Size; 105, 188, 102, and 186 lines respectively

(1) Outline

JOYOS(D) and J1Y1S(D) are subroutine subprograms that calculate 0th and 1st order Bessel functions $J_0(x)$ and $Y_0(x)$, and $J_1(x)$ and $Y_1(x)$ for a single (double) precision real number x, with single (double) precision.

(2) Directions

CALL JOYOS/D(X, VJ, VY, ILL)

CALL J1Y1S/D(X, VJ, VY, ILL)

Argument	Type and kind (*1)	Attribute	Content
X	Real type	Input	Value of variable x. 0≦X≦8.23·10**5(3.53·10**15)
VJ	Real type	Output	Value of Bessel function of the first kind. $J_0(x)$ or $J_1(x)$
VY	Real type	Output	Value of Bessel function of the second kind. $Y_0(x)$ or $Y_1(x)$

Argument	Type and kind (*1)	Attribute	Content
ILL	Integer type	Output	ILL=0: Normal termination. ILL=30000: Argument X is outside of the range. VJ=0.0 and VY=0.0.

*1 For double precision subroutines, all real types should be changed to double precision real types.

(3) Calculation method

1) When $x \leqq 0$ or $x > 8.23 \cdot 10^{**}5$ ($3.53 \cdot 10^{**}5$), an error results.

2) When $0 < x \leqq 2$, optimal polynomial approximations A_0, A_1, B_0, and B_1 are used to calculate:

$$J_0(x) = A_0(x^*x)$$

$$Y_0(x) = 2/\pi * J_0(x)\log(x) + B_0(x^*x),$$

$$J_1(x) = x^*A_1(x^*x)$$

$$Y_1(x) = 2/\pi * J_1(x)\log(x) + x^*B_1(x^*x) - 2/(\pi x)$$

3) When $2 < x \leqq 4$, optimal polynomial approximations C_1, C_2, C_3, and C_4 are used to calculate:

$$J_0(x) = C_1(x-3),$$

$$Y_0(x) = C_2(x-3),$$

$$J_1(x) = C_3(x-3),$$

$$Y_1(x) = C_4(x-3) - 2/(\pi x)$$

4) When $4 < X \leqq 6$, optimal polynomial approximations D_1, D_2, D_3, and D_4 are used to calculate:

$$J_0(x) = D_1(x-5),$$

$$Y_0(x) = D_2(x-5),$$

$$J_1(x) = D_3(x-5),$$

$$Y_1(x) = D_4(x-5).$$

5) When $6 < X \leqq 8$, optimal polynomial approximations E_1, E_2, E_3, and E_4 are used to calculate:

144

$J_0(x)=E_1(x-7)$,

$Y_0(x)=E_2(x-7)$,

$J_1(x)=E_3(x-7)$,

$Y_1(x)=E_4(x-7)$.

6) When $8<x\leqq 8.23\cdot 10^{**}5(3.53\cdot 10^{**}15)$, optimal polynomial approximations (rational functions) F_0, F_1, G_0, and G_1 are used to calculate:

$M_0=(F_0((8/x)^{**}2)*8/x)^{1/2}$,

$M_1=(F_1((8/x)^{**}2)*8/x)^{1/2}$,

$P_0=G_0((8/x)^{**}2)*8/x-\pi/4+x$,

$P_1=G_1((8/x)^{**}2)*8/x-\pi/4+x$,

$J_0(x)=M_0*\cos(P_0)$,

$Y_0(x)=M_0*\sin(P_0)$,

$J_1(x)=M_1*\sin(P_1)$,

$Y_1(x)=-M_1*\cos(P_1)$.

(4) Note

This routine uses the entire common part of the calculation method of $J_0(x)$ and $Y_0(x)$, or $J_1(x)$ and $Y_1(x)$. Therefore, when both of these functions are calculated, it is more advantageous to use this routine than to use individual functions to calculate them separately.

(1989.01.20)

SIO/DSIO, SI1/DSI1, SKO/DSKO, SK1/DSK1 (Modified spherical Bessel functions of the order 0 and 1)

Modified Spherical Bessel Functions of the Order 0 and 1

Programmed by	Ichizo Ninomiya: April 1977
Format	Function Language; FORTRAN Size; 18, 22, 18, 22, 23, 31, 24, and 31 lines respectively

(1) Outline

SIO (DSIO), SI1 (DSI1), SKO (DSKO), and SK1 (DSK1) calculate $i_0(x)$, $i_1(x)$, $k_0(x)$, and $k_1(x)$ respectively, with single (double) precision, for a single (double) precision real number x.

Where,

$$i_n(x)=\sqrt{\frac{\pi}{2x}}I_{n+1/2}(x),\quad k_n(x)=\sqrt{\frac{\pi}{2x}}K_{n+1/2}(x)$$

(2) Directions

1. SIO(X), SI1(X), SKO(X), SK1(X)

   DSIO(D), DSI1(D), DSKO(D), DSK1(D)

   X and D are arbitrary single and double precision real-type expressions respectively. The double precision function name needs to be declared as double precision.

2. Range of argument

   For modified spherical Bessel function of the first kind: $|X|<174.673$ and $|D|<174.673$.

   For modified spherical Bessel function of the second kind: $0<X$, $0<D$.

3. Error processing

   If the specified argument is outside the range, an error message is printed but calculation continues with the function value assumed to be 0. (See FNERST.)

146

(3) Calculation method

1. SI0 and DSI0

(1) In case of $|x| \geq 174.673$, an error results.

(2) In case of $|x| < 1$, $i_0(x)$ is calculated by a polynomial approximation.

(3) In case of $|x| \geq 1$, $i_0(x) = sinhx/x$ is calculated.

2. SI1 and DSI1

(1) In case of $|x| \geq 174.673$, an error results.

(2) In case of $|x| < 1$, $i_1(x)$ is calculated by a polynomial approximation.

(3) In case of $|x| \geq 1$, $i_1(x) = 1/x(coshx - sinhx/x)$ is calculated.

3. SK0 and DSK0

(1) In case of $x \leq 0$, an error results.

(2) In case of $x > 180.218$, $k_0(x) = 0$.

(3) In case of $0 < x < 1$, $k_0(x)$ is calculated by a rational approximation.

(4) In case of $1 \leq x \leq 180.218$, $k_0(x) = \pi/2 \cdot e^{-x}/x$ is calculated.

4. SK1 and DSK1

(1) In case of $x \leq 0$, an error results.

(2) In case of $x > 180.218$, $k_1(x) = 0$.

(3) In case of $0 < x < 1$, $k_1(x)$ is calculated by a rational approximation.

(4) In case of $1 \leq x \leq 180.218$, $k_1(x) = \pi/2(1+1/x)e^{-x}/x$ is calculated.

(4) Notes

The modified spherical Bessel function can easily be defined by exponential and hyperbolic functions. If it is calculated as defined, however, severe cancellation occurs near the origin, resulting in deteriorated precision. Using the above functions can eliminate such problem and also save the calculation time.

(1987.07.31)

SIK/DSIK and SKN/DSKN (Modified spherical Bessel functions of integral orders)

Modified Spherical Bessel Functions of Integral Orders

Programmed by	Ichizo Ninomiya; September 1981
Format	Function Language; FORTRAN Size; 125, 188, 49, and 50 lines respectively

(1) Outline

SIK (DSIK) and SKN (DSKN) calculate $i_n(x)$ and $k_n(x)$ respectively, with single (double) precision, for an integer n and a single (double) precision real number x.

Where,

$$i_n(x)=\sqrt{\frac{\pi}{2x}}I_{n+1/2}(x), \quad k_n(x)=\sqrt{\frac{\pi}{2x}}K_{n+1/2}(x)$$

(2) Directions

1. SIK(N, X) and SKN(N, X)

   DSIK(N, D) and DSKN(N, D)

   N, is an arbitrary integer-type expression. X and D are arbitrary single and double precision real-type expressions respectively. The double precision function name needs to be declared as double precision.

2. Range of argument

   Modified Bessel function of the first kind $i_n(x)$:

   N≧0, X≧0 (D≧0)

   However, any region where the function value overflows is excluded.

   Modified Bessel function of the second kind:

   N≧0, X>0 (D>0)

   However, any region where the function value overflows is excluded.

3. Error processing

148

If the specified argument is outside the range, an error message is printed but calculation continues with the function value assumed to be 0. (See FNERST.)

(3) Calculation method

1. SIK (DSIK)

(1) In case of $x<0$ or $n<0$, an error results.

(2) In case of $(x/2)^2 \leqq n+3/2$, the following Taylor series is calculated:

$$i_n(x)=\frac{x^n}{1\cdot 3\cdots(2n+1)}\left\{1+\frac{x^2/2}{1!\,(2n+3)}+\frac{(x^2/2)^2}{2!\,(2n+3)(2n+5)}+\cdots\right\}$$

(3) In case of $x \geqq 10(18)$ and $x \geqq 0.55(n+1/2)^2$, the following asymptotic expansion is used:

$$i_n(x)=\frac{e^x}{2x}\left\{1-\frac{\mu-1}{8x}+\frac{(\mu-1)(\mu-9)}{2!\,(8x)^2}-\frac{(\mu-1)(\mu-9)(\mu-25)}{3!\,(8x)^3}+\cdots\cdots\right\}$$

where $\mu=(2n+1)^2$.

(4) In case of $n+1/2 \geqq 10(35)$ in a region other than (2) and (3), the following uniform asymptotic expansion is used:

$$i_n(x)=\frac{1}{2\sqrt{nx}}\frac{e^{(n+\frac{1}{2})\eta}}{(1+\xi^2)^{1/4}}\left\{1+\sum_{k=1}^{\infty}\left(n+\frac{1}{2}\right)^{-K}u_k(t)\right\}$$

This calculation is based on the following:

$$\xi=x/(n+\frac{1}{2}),\ t=1/\sqrt{1+\xi^2},\quad \eta=\sqrt{1+\xi^2}+\log\left[\ \xi/(1+\sqrt{1+\xi^2})\ \right]$$

where $u_k(t)$ is a kth polynomial of t.

(5) In case of an area other than the above, the following recurrence formula is used :

$$i_{k-1}(x)=(2k+1)\,i_k(x)/x+i_{k+1}(x)$$

2. SKN (DSKN)

(1) In case of $n<0$ or $x\leqq 0$, an error results.

(2) In case of $x>180.218$, $k_n(x)=0$.

(3) In case of $n=0$, SK0 (DSK0) is called to calculate $k_0(x)$.

(4) In case of $n=1$, SK1 (DSK1) is called to calculate $k_1(x)$.

(5) In case of $n\geqq 2$, the following recurrence formula is repeatedly calculated, starting with $k_0(x), k_1(x)$, to obtain $k_1(x)$:

$$k_{l+1}(x)=\frac{(2l+1)}{x}k_l(x)+k_{l+1}(x)$$

Bibliography

1) Ichizo Ninomiya; "Calculation of Bessel functions by recurrence formula", Numerical method II for computers, Baifu-kan (1966).

2) D. E. Amos et al; "CDC 6600 Subroutines IBESS and JBESS for Bessel Functions $I\upsilon(x)$ and $J\upsilon(x), x\geqq 0, \upsilon\geqq 0$", ACM Trans. on Math. Software, Vol. 3, No. 1, pp. 76-92 (1977).

(1989. 01. 25)

150

SJ0/DSJ0, SJ1/DSJ1, SY0/DSY0, SY1/DSY1 (Spherical Bessel functions of the order 0 and 1)

Spherical Bessel Functions of the Order 0 and 1

Programmed by	Ichizo Ninomiya; April 1977
Format	Function Language; FORTRAN Size; 18, 22, 18, 22, 19, 25, 19, and 25 lines respectively

(1) Outline

SJ0 (DSJ0), SJ1 (DSJ1), SY0 (DSY0), and SY1 (DSY1) calculate $j_0(x)$, $j_1(x)$, $y_0(x)$, and $y_1(x)$ respectively, with single (double) precision, for a single (double) precision real number x.

Where,

$$j_n(x)=\sqrt{\frac{\pi}{2x}}J_{n+1/2}(x),\quad y_n(x)=\sqrt{\frac{\pi}{2x}}Y_{n+1/2}(x)$$

(2) Directions

1. SJ0(X), SJ1(X), SY0(X), SY1(X),

   DSJ0(D), DSJ1(D), DSY0(D), DSY1(D)

   X and D are arbitrary single and double precision real-type expressions respectively. The double precision function name needs to be declared as double precision.

2. Range of argument

   For Spherical Bessel function of the first kind: $|X|\leq 8.23\cdot 10^5$, $|D|\leq 3.53\cdot 10^{15}$.

   For Spherical Bessel function of the second kind:

   $0<|X|\leq 8.23\cdot 10^5$, $0<|D|\leq 3.53\cdot 10^{15}$.

3. Error processing

   If the specified argument is outside the range, an error message is printed but calculation continues with the function value assumed to be 0. (See FNERST.)

(3) Calculation method

1. SJ0(DSJ0)

   (1) In case of $|x| \geq 8.23 \cdot 10^5$ ($|x| \geq 3.53 \cdot 10^{15}$), an error results.

   (2) In case of $|x|<1$, $j_0(x)$ is calculated by a polynomial approximation.

   (3) In case of $|x| \geq 1$, $j_0(x)=sinx/x$ is calculated.

2. SJ1(DSJ1)

   (1) In case of $|x| \geq 8.23 \cdot 10^5$ ($|x| \geq 3.53 \cdot 10^{15}$), an error results.

   (2) In case of $|x|<1$, $j_1(x)$ is calculated by a polynomial approximation.

   (3) In case of $|x| \geq 1$, $j_1(x)=1/x(sinx/x-cosx)$ is calculated.

3. SY0(DSY0)

   (1) In case of $x=0$ or $|x| \geq 8.23 \cdot 10^5$ ($|x| \geq 3.53 \cdot 10^{15}$), an error results.

   (2) In case of $|x|<1$, $y_0(x)$ is calculated by a polynomial approximation.

   (3) In case of $|x| \geq 1$, $y_0(x)=-cosx/x$ is calculated.

4. SY1(DSY1)

   (1) In case of $x=0$ or $|x| \geq 8.23 \cdot 10^5$ ($|x| \geq 3.53 \cdot 10^{15}$), an error results.

   (2) In case of $|x|<1$, $y_1(x)$ is calculated by a polynomial approximation.

   (3) In case of $|x| \geq 1$, $y_1(x)=-1/x^2 \cdot (cosx+xsinx)$ is calculated.

(4) Notes

1. The spherical Bessel function can easily be defined using a trigonometric function. If it is calculated as defined, however, severe cancellation occurs near the origin, resulting in deteriorated precision. Using the above functions can eliminate such problem and also save the calculation time.

2. The symbol $n_n(x)$ may be used instead of $y_n(x)$.

(1987.07.09)

152

SJN/DSJN, SYN/DSYN (Spherical Bessel functions of integer orders)

Spherical Bessel Functions of Integral Orders

Programmed by	Ichizo Ninomiya; September 1981
Format	Function Language; FORTRAN Size; 93, 98, 46, and 48 lines respectively

(1) Outline

SJN (DSJN) and SYN (DSYN) calculate $j_n(x)$ and $y_n(x)$ respectively, with single (double) precision, for an integer n and a single (double precision real number x.

Where,

$$j_n(x)=\sqrt{\frac{\pi}{2x}}J_{n+1/2}(x),\quad y_n(x)=\sqrt{\frac{\pi}{2x}}Y_{n+1/2}(x)$$

(2) Directions

1. SJN(N, X), DSJN(N, D), SYN(N, X), and DSYN(N, D)

   N is an arbitrary integer-type expression. X and D are arbitrary single and double precision real-type expressions respectively. The double precision function name needs to be declared as double precision.

2. Range of argument

   Spherical Bessel function of the first kind $j_n(x)$:

   $N\geq 0,\ 0\leq X\leq 8.23\cdot 10^5 (0\leq D\leq 3.53\cdot 10^{15})$

   $X>200,\ N>1.384\sqrt{X}\ (D>200,\ N\geq 1.384\sqrt{D})$ are excluded.

   Spherical Bessel function of the second kind $y_n(x)$:

   $N\geq 0,\ 0<X\leq 8.23\cdot 10^5 (0<X\leq 3.53\cdot 10^{15})$

   However, any region where the function value overflows is excluded.

3. Error processing

   If the specified argument is outside the range, an error message is printed but calculation continues with the function value assumed to be 0. (See FNERST.)

(3) Calculation method

1. SJN(DSJN)

(1) In case of $n<0$, $x<0$, or $x>8.23\cdot10^5(3.53\cdot10^{15})$, an error results.

(2) In case of $(x/2)^2\leqq n+3/2$, Taylor series

$$j_n(x)=\frac{x^n}{1\cdot3\cdots(2n+1)}\left\{1-\frac{x^2/2}{1!\,(2n+3)}+\frac{(x^2/2)^2}{2!\,(2n+3)(2n+5)}-\cdots\right\}$$

is calculated.

(3) In case of $x\geqq10(18)$ and $n+1/2\leqq1.384\sqrt{x}$, the following expansion formula is used:

$$j_n(x)=\{P(n,x)\sin\phi+Q(n,x)\cos\phi\}/x$$

where

$$\phi=x-n\pi/2,$$

$$P(n,x)=1-\frac{(\mu-1)(\mu-9)}{2!\,(8x)^2}+\cdots\cdots,$$

$$Q(n,x)=\frac{\mu-1}{x}-\frac{(\mu-1)(\mu-9)(\mu-25)}{3!\,(8x)^3}+\cdots\cdots,$$

$$\mu=(2n+1)^2$$

(4) In case of $x\leqq200$ in a region other than (2) and (3), the following recurrence formula is used:

$$j_{k-1}(x)=(2k+1)j_k(x)/x-j_{k+1}(x)$$

(5) In case of a region other than the above, an error results.

2. SYN (DSYN)

(1) In case of $n<0$, $x\leqq0$, or $x>8.23\cdot10^5(x>3.53\cdot10^{15})$, an error results.

(2) In case of $n=0$, SY0(DSY0) is called to calculate $y_0(x)$.

(3) In case of $n=1$, SY1(DSY1) is called to calculate $y_1(x)$.

(4) In case of $n \geqq 2$, the following recurrence formula is repeatedly calculated, starting with $y_0(x), y_1(x)$, to obtain $y_n(x)$:

$$y_{k+1}(x)=\frac{(2k+1)}{x}y_k(x) - y_{k-1}(x)$$

(4) Note

Spherical Bessel function of the second kind $n_n(x)$ is the same as $y_n(x)$.

Bibliography

1) Ichizo Ninomiya; "Calculation of Bessel functions by recurrence formula", Numerical method II for computers, Baifukan (1966)

(1989. 01. 12)

ZBJ0/DZBJ0, ZBJ1/DZBJ1, ... , ZBJ15/DZBJ15 (Positive zeroes of J_0~J_{15})

Positive Zeroes of Bessel Functions J_0~J_{15}

Programmed by	Ichizo Ninomiya: December 1983, revised in July 1986 and December 1988
Format	Function Language; FORTRAN Size; 120 lines

(1) Outline

Each of functions ZBJ0 (or DZBJ0) to ZBJ15 (or DZBJ15) calculates the n-th positive zero in J_0 to J_{15} for positive integer n, with single (or double) precision.

(2) Directions

2.1 {ZBJ0(N)~ZBJ15(N)} {DZBJ0(N)~DZBJ15(N)}

N is an arbitrary integer type expression. Each function whose name begins with D requires declaration of double precision.

2.2 Range of argument

$1 \leqq N \leqq 1000$ for ZBJ0 and ZBJ1 (DZBJ0 and DZBJ1); $1 \leqq N \leqq 100$ for other functions

2.3 Error processing

If the argument is outside the range, an error message is printed but calculation continues with the function value assumed to be 0.

(3) Calculation method

When $1 \leqq N \leqq 100$, a numerical table calculated beforehand with sufficient precision is used.

When $100 < N \leqq 1000$, an asymptotic expansion formula is used.

Bibliography

1) Watson, G. N., Theory of Bessel Functions, Cambridge University Press, pp. 503-506 (1922)

(1989. 01. 25)

ZBJOS/D and ZBJ1S/D (Zeros and derivatives of Bessel functions J_0 and J_1)

Zeroes and Deviatives of Bessel Functions J_0 and J_1

Programmed by	Ichizo Ninomiya: December 1983
Format	Subroutine Language; FORTRAN Size: 210 lines

(1) Outline

ZBJOS (or ZBJOD) and ZBJ1S (or ZBJ1D) each calculate the n-th positive zeros {jon, j1n} and {J_0 (jon), J_1 (j1n)} of Bessel functions {J_0, J_1} for positive integer n with single (or double) precision.

(2) Directions

CALL ZBJOS/D(N, ZERO, DERIV, ILL)

CALL ZBJ1S/D(N, ZERO, DERIV, ILL)

Argument	Type and kind (*1)	Attribute	Content
N	Integer type	Input	Number of positive zero: $1 \leqq N \leqq 100$
ZERO	Real type	Output	N-th zero jon (j1n) of Bessel function J_0 (J_1)
DERIV	Real type	Output	Derivative J_0 (jon) (J_1 (j1n)) at zero jon (j1n)
ILL	Integer type	Output	ILL=0: Normal ILL=30000: N<1 or N>100.

*1 For double precision subroutines, all real types should be changed to double precision real types.

(3) Calculation method

A numerical table calculated beforehand with sufficient precision is used.

(4) Note

$J_0(jon) = -J_1(jon)$ and $J_1(j1n) = J_0(j1n) = -J_2(j1n)$.

(1989. 01. 25)

158

ZBJN/DZBJN (Positive Zeroes of Bessel Functions J_0~J_{15})

Positive Zeroes of Bessel Functions J_0~J_{15}

Programmed by	Ichizo Ninomiya, December 1983, revised in July 1986
Format	Function Language: FORTRAN; Size: 42 and 43 lines respectively

(1) Outline

{ZBJN(DZBJN)} calculates the k-th zero of J_n for positive integers n and k by single (double) precision.

(2) Directions

2.1 {ZBJN(N, K), DZBJN(N, K)}

An arbitrary expression of the integer type can be written in N and K. DZBJN requires the declaration of double precision.

2.2 Range of argument

$0 \leqq N \leqq 15$, $1 \leqq K \leqq 100$. However, $1 \leqq K \leqq 1000$ for $0 \leqq N \leqq 1$.

2.3 Error processing

If an argument outside the range is given, a message is printed as an error and the calculation is continued with the function value as 0.

(3) Calculation method

A numerical table precomputed with a sufficient precision is used for $1 \leqq N \leqq 100$.

An asymptotic expansion formula is for $100 < N \leqq 1000$.

Bibliography

1) Watson, G. N., Theory of Bessel Functions, Cambridge University Press, pp. 503-506 (1922)

(1989. 01. 25)

15. Acceleration of sequences and series

160

ACCELS/D, LEVNTS/D, LEVNUS/D, WYNNES/D, WYNNRS/D, EULERS/D, BRZSKS/D (Acceleration of convergence of sequence or series)

Acceleration of Convergence of Sequences or Series

Programmed by	Ichizo Ninomiya: October 1984
Format	Subroutine Language; FORTRAN Size; 220 and 224 respectively

(1) Outline

Each of these subroutines calculates a limit value with the requested precision by accelerating convergence of sequence $S_1, S_2, \cdots, S_n, \cdots$ or series $a_1+a_2+\cdots+a_n+\cdots$, which generally shows slow convergence.

The series (a sequence is regarded as series $a_n=\Delta S_{n-1}$) is divided into the following five types.

1. Alternating series

This type of series satisfies $a_{n+1}/a_n<0$. Example: $a_n=(-1)^{n-1}/n$

2. Series of linear convergence

This type of series satisfies $a_{n+1}/a_n \rightarrow \lambda$, $|\lambda|<1$. Example: $an=0.5^n+(-0.8)^n$

3. Series of logarithmic convergence

This type of series satisfies $a_{n+1}/a_n \rightarrow 1$. Example: $a_n=1/n^2$

4. Irregular sign series

This type of series has a_n whose sign is irregular. Example: $a_n=sinnx/n^2$

5. Others

Depending on the type of series given, each of these subroutines performs Levin's t-, Levin's u-, Wynn's ε-, or Wynn's ρ-transformation as follows:

1. Alternating series : t-transformation.
2. Series of linear convergence : ε-transformation.
3. Series of logarithmic convergence : ρ-transformation.
4. Irregular sign series : ε-transformation.
5. Others : u-transformation.

162

(2) Directions

CALL ACCELS/D(ICAT, ISER, FUNTERM, EPSA, EPSR, NMAX, V, ERR, NTERM, ILL)

Argument	Type and kind (*1)	Attribute	Content
ICAT	Integer type	Input	Type of series (a sequence is regarded as a series with term $\triangle$Sn-1) is specified. ICAT=1: Alternating series. u-transformation is used. ICAT=2: Series of linear convergence. ε-transformation is used. ICAT=3: Series of logarithmic convergence. ρ-transformation is used. ICAT=4: Irregular sign series. ε-transformation is used. ICAT=5: Type unknown. u-transformation is used. A value other than the above receives the same treatment as ICAT=5.
ISER	Integer type	Input	ISER=0 indicates a sequence. ISER$\neq$0 indicates a series.
FUNTRM	Real type Function subprogram	Input	General term an or Sn is given as a function of number n of the term. This subprogram must be declared in the EXTERNAL statement in the calling program.
EPSA	Real type	Input	Absolute tolerance εa.
EPSR	Real type	Input	Relative tolerance εr.
NMAX	Integer type	Input	Maximum allowable number of terms. NMAX$\geqq$4.
V	Real type	Output	Limit value. The target value is $\lvert V-Vo \rvert \leqq \max(\lvert Vo \rvert \varepsilon r, \varepsilon a)$, where Vo is the true value.
ERR	Real type	Output	Estimated value of $\lvert V-Vo \rvert$.
NTERM	Integer type	Output	Number of terms used.
ILL	Integer type	Output	Error code. ILL=0: Normal termination. ILL=10000: The convergence condition has not been established but there is no room for further improvement because of loss of significant digits. ILL=20000: Convergence has not occurred though the maximum number of terms was reached. ILL=30000: NMAX<4.

*1 For double precision subroutines, all real types should be changed to double precision real types.

(3) Calculation method

1. ε-transformation

ε-transformation starts with 1 and determines 3 through the recurrent equation 2. This routine calculates subsequence 4 until the convergence conditions are met.

1 $\varepsilon_{-1}^{(n)}=0, \varepsilon_0^{(n)}=S_n(=\sum_{i=1}^{n} a_i), n=1,2,\cdots$

2 $\varepsilon_k^{(n)}=\varepsilon_{k-1}^{(n+1)}+1/\Delta\varepsilon_{k-1}^{(n)}$

3 $\varepsilon_k^{(n)}, n=1,2,\cdots;k=1,2,\cdots,n-1$

4 $\varepsilon_0^{(1)}, \varepsilon_1^{(2)}, \cdots \varepsilon_{n-1}^{(n)}\cdots$

2. ρ-transformation

ρ-transformation starts with 1 and determines 3 through the recurrent equation 2. This routine calculates subsequence 4 until the convergence conditions are met.

1 $\rho_{-1}^{(n)}=0, \rho_0^{(n)}=S_n=(\sum_{i=1}^{n} a_i), n=1,2,\cdots$

2 $\rho_k^{(n)}=\rho_{k-1}^{(n+1)}+k/\Delta\rho_{k-1}^{(n)}$

3 $\rho_k^{(n)}, n=1,2,\cdots;k=1,2,\cdots n-1$

4 $\rho_0^{(1)}, \rho_1^{(2)}, \cdots, \rho_{n-1}^{(n)}\cdots$

3. t-transformation and u-transformation

t-transformation and u-transformation each start with 1 and determine 2. This routine calculates subsequence 3 until the convergence conditions are met.

164

1 $t_0^{(n)}=S_n(=\sum_{i=1}^{n}a_n),n=1,2,\cdots$

2 $t_k^{(n)}=\Delta^k(\frac{n^{k-1}S_n}{a_n})/\Delta^k(\frac{n^{k-1}}{a_n})n=1,2,\cdots;k=1,2,\cdots n-1$

3 $t_0^{(1)},t_1^{(2)},\cdots t_{n-1}^{(n)}\cdots$

u-transformation is a transformation where t of t-transformation is changed to u and n^{k-1} of 2 is changed to n^{k-2}.

Refer to the paper in the bibliography for details of the calculation method.

(4) Example

Series

The following sample program calculates

$1-1/3+1/5-\cdots+^{(-1)^{n-1}}/(2n-1)+\cdots=\frac{\pi}{4}$

with double precision with required absolute accuracy 10^{-10}.

```
      IMPLICIT REAL*8 (A-H,O-Z)
      EXTERNAL FUNTRM
      ICAT=1
      ISER=1
      EPSA=1.D-10
      EPSR=0.DO
      NMAX=50
      CALL ACCELD(ICAL,ISER,FUNTRM,EPSA,EPSR,NMAX,V,ERR,NTERM,
     ILL)
      D=DABS(V-DATAN(1.DO))
      WRITE(6,600) V,D,ERR,NTERM,ILL
600   FORMAT(1H ,D18.10,2D11.3,2I6)
      STOP
      END
      FUNCTION FUNTRM(N)
      IMPLICIT REAL*8 (A-H,O-Z)
      IF(N.EQ.1) SGN=1.DO
      FUNTRM=SGN/DFLOAT(N+N-1)
      SGN=-SGN
      RETURN
      END
```

(5) Notes

1. The series other than the alternating series may suffer loss of significant digits. So, do not make the tolerance too small; otherwise, convergence will not occur.

2. To perform t-, u-, ε-, or ρ-transformation, each of these routines calls subroutine LEVNTS/D, LEVNUS/D, WYNNES/D, or WYNNRS/D correspondingly. The specifications of these subroutines are the same as those for the present routines except that the former does not have input argument ICAT. Taking account of these conditions, one can use these special subroutines directly. Euler transformation routine EULERS/D having the same specifications and Brezinsky θ-transformation routine BRZSKS/D are also available.

3. A function subprogram that calculates a general term must be defined as a function of only the number N (integer beginning from 1) of the term. This function routine is always called in natural sequence like N=1, 2, 3 ... from the beginning. Use this to save the quantity of calculation. (See the example above.)

Bibliography

(1) T. Fessler; ACM Trans. Math. Softw. Vol. 9, pp. 371-381 (1983).

(2) D. A. Smith; SIAM J. Numer. Anal. Vol. 16, pp. 223-240 (1979).

(1987. 07. 25) (1988. 06. 20)

16. Linear programming

LIPS/D (Linear programming by CRISS-CROSS method)

Linear Programming by CRISS-CROSS Method

Programmed by	Yasuo Akatsuka; November 1980
Format	Subroutine Language; FORTRAN Size; 2684 (2770) byte

(1) Outline

The minimization of optimum control and the maximization problem are solved by the CRISS-CROSS method by which the simplex method is applied. LIPS is a single precision routine, and LIPD is a double precision routine.

(Minimize)

(Objective function) $z=a_{1,1}+a_{1,2}x_1+a_{1,3}x_2+\cdots\cdots+a_{1,n}x_{n-1}$

(Conditional expression) $a_{2,1}+a_{2,2}x_1+a_{2,3}x_2+\cdots\cdots+a_{2,n}x_{n-1}\geqq 0$

$\cdots\cdots$

$a_{m,1}+a_{m,2}x_1+a_{m,3}x_2+\cdots\cdots+a_{m,n}x_{n-1}\geqq 0$

(Prerequisite condition) $x_i(i=1,\cdots,n-1)\geqq 0$

(2) Directions

CALL LIPS/D(A, L, M, N, IW, Y, EPS, ICON)

Argument	Type and kind (*1)	Attribute	Content
A	Real type Two-dimensional array	Input/output	A coefficient matrix of the objective function and conditional expression is input.
L	Integer type	Input	Value of the first subscript in array declaration of A. L≧M
M	Integer type	Input	Number of rows of A (number of conditional expressions + 1 (objective function)).
N	Integer type	Input	Number of columns of A (number of unknowns of equation + 1).

168

Argument	Type and kind (*1)	Attribute	Content
IW	Integer type One-dimensional array	Input/output	Name of one-dimensional array with M+N number of elements. This is used as a work area.
Y	Real type One-dimensional array	Output	One-dimensional array with M+N-1 number of elements. Y_1: Minimum (maximum) value of objective function. $Y_{2\sim n}$: Optimal solution. $Y_{n+1\sim n+m-1}$: Slack value.
EPS	Real type	Input	Values smaller than EPS are considered 0 during calculation.
ICON	Integer type	Input/output	Input. ICON=0: Calculation is performed until an optimal solution is obtained. ICON≠0: The program returns each time a solution is output. Output. ICON=30000: No solution is obtained. ICON=1000: Called with ICON≠0. ICON=0: The solution was normally obtained.

*1 For double precision subroutines, all real types should be changed to double precision real types.

(3) Notes

1. To solve a maximization problem, objective function Z can be given with the signs of its coefficients reversed. However, the value of Y is the absolute value.

2. To solve an equalities problem, the equalities can be changed to inequalities by adding conditional expressions with signs reversed, or by adding variables and conditional expressions.

Bibliography

1) N. K. Kwak; "Mathematical Programming with Business Applications," McGraw-Hill Book Company (1973).

2) T. Hu; "Integer Programming And Network Flows," ADDISON-WESLEY PUBLISHING COMPANY, (1970).

(1987.07.07)

SIMPLX/SIMPLD (Linear programming by simplex method)

Linear Programming by Simplex Method

Programmed by	Michiyo Kato; January 1975
Format	Subroutine language; FORTRAN Size; 138 and 139 lines respectively

(1) Outline

SIMPLX/SIMPLD solves maximization and minimization problems for optimum control, using the simplex method which is a kind of linear programming.

Under restrictive conditions;

$$a_{i1}x_1+\cdots\cdots+a_{in0}x_{n0}\leqq s_i \quad (i=1,\cdots,m_1) \tag{1}$$

$$a_{j1}x_1+\cdots\cdots+a_{jn0}x_{n0}\geqq s_j \quad (j=1,\cdots,m_2) \tag{2}$$

$$a_{k1}x_1+\cdots\cdots+a_{kn0}x_{n0}=s_k \quad (k=1,\cdots,m_3) \tag{3}$$

and

$$x_l\geqq 0 \quad (l=1,\cdots,n_0) \tag{4}$$

$x_1,\cdots,x_{n0}$ which maximizes

$$z=c_1x_1+\cdots\cdots+c_nx_n \tag{5}$$

is determined. SIMPLX is a subroutine for single precision and SIMPLD is for double precision.

(2) Directions

CALL SIMPLX(A, KA, M1, M2, M3, NO, S, C, INDEX, OF, X, EPS, ILL)

Argument	Type and kind (*1)	Attribute	Content
A	Real type Two-dimensional array	Input/output	The coefficient matrix of the conditional expression is set in the first NO columns. The size with M=M1+M2+M3 rows and NO+M+M2 columns is needed.
KA	Integer type	Input	First subscript of AKA≧M1+M2+M3

170

Argument	Type and kind (*1)	Attribute	Content
M1	Integer type	Input	M1 is the number of standard conditional expressions in expression (1). M1≧0
M2	Integer type	Input	M2 is the number of inequalities in the opposite direction in expression (2). M2≧0
M3	Integer type	Input	M3 is the number of equations in expression (3). M3≧0
NO	Integer type	Input	NO is the number of variables. NO≧2
S	Real type One-dimensional array	Input	S(M), one-dimensional array name. The right-hand sides of conditional expressions (1), (2), and (3) are set. Note M=M1+M2+M3.
C	Real type One-dimensional array	Input	C(N), one-dimensional array name. The coefficients of evaluation function (5) are set. Note N=NO+M+M2.
INDEX	Integer type	Input	1 is set for maximization and -1 is set for minimization.
X	Real type One-dimensional array	Output	X(N), one-dimensional array name. An optimum solution is put in X(1), ..., X(NO), and the value of the slack variable *2 is put in X(NO+1), ..., X(N).
OF	Real type	Output	The maximum (minimum) value of the objective function is put in this argument.
EPS	Real type	Input	A value smaller than EPS detected during calculation is regarded as 0. EPS>0

Argument	Type and kind (*1)	Attribute	Content
ILL	Integer type	Output	ILL = 30000: The input argument does not satisfy the specified conditions. ILL = 1001: When INDEX = 1, $z_j-c_j<0$ is established and all elements of the corresponding base vector are negative and θ^* are negative. ILL = 1002: When INDEX = −1, $z_j-c_j>0$ is established and all elements of the corresponding base vector are negative and θ^* are negative.

*1 For double precision subroutines, real types are all assumed to be double precision real types.

*2 See the calculation method in (3).

(3) Calculation method

Here, we take up an example under standard conditional expressions only ($m=m_1$). If slack variable $\lambda_1 \geqq 0, \cdots, \lambda_m \geqq 0$ is introduced, expression (1) becomes

$$a_{i1}x_1+\cdots+a_{in0}x_{n0}+\lambda i=s_i \quad (i=1,\cdots,m) \tag{6}$$

. If $\lambda_1, \cdots, \lambda_m$ is assumed to be $x_{n0+1}, \cdots, x_{n0+m}$, expressions (6), (4), and (5) are represented by:

$$\sum_{j=0}^{n0} a_{ij}x_j+\sum_{j=1}^{m} x_{n0+j}=s_i \quad (i=1,\cdots,m) \tag{7}$$

$$x_j \geqq 0 \quad (j=1,\cdots,n0+m) \tag{8}$$

$$Z=\sum_{j=1}^{n0+m} c_j x_j \quad (j=1,\cdots,n0+m) \tag{9}$$

If (7) is assumed to be $AX=S$, A is the matrix of $m\times(n0+m)$, X is the column vector of $(n0+m)$ and S is the column vector of m.

172

Step 1

m base vectors $a_{n0+1},\cdots,a_{n0+m}$ are selected from matrix A. The rank of $(a_{n0+1},\cdots,a_{n0+m})$ is m.

X which satisfies

$$x_{n0+1}a_{n0+1}+\cdots+x_{n0+m}a_{n0+m}=S \tag{10}$$

is a feasible basic solution.

Step 2

Initial end point is supposed to be;

$$X=(0,\cdots,0,x_{n0+1},\cdots,x_{n0+m}) \tag{11}$$

$a_{n0+1},\cdots,a_{n0+m}$ is linearly independent, and $a_1,\cdots,a_m$ is represented by

$$a_j=x_{n0+1,j}a_{n0+1}+\cdots+x_{n0+m,j}a_{n0+m} \quad (j=1,\cdots,m) \tag{12}$$

. From expressions (9) and (11), the following is obtained:

$$z_0=c_{n0+1}x_{n0+1}+\cdots+c_{n0+m}x_{n0+m} \tag{13}$$

z_j is supposed to be;

$$z_j=x_{n0+1,j}c_{n0+1}+\cdots+x_{n0+m,j}c_{n0+m} \tag{14}$$

Then, for adequate positive number θ, (10)$-\theta\times$(12) and (13)$-\theta\times$(14) become;

$$(x_{n0+1}-\theta x_{n0+1,j})a_{n+1}+\cdots+(x_{n0+m}-\theta x_{n0+m,j})a_{n0+m}+\theta a_j=S \tag{15}$$

$$(x_{n0+1}-\theta x_{n0+1,j})c_{n0+1}+\cdots+(x_{n0+m}-\theta x_{n0+m,j})c_{n0+m}+\theta c_j=z_0-\theta(z_j-c_j) \tag{16}$$

If θ which makes one of $(x_{n0+i}=\theta x_{n0+i,j})(i=1,\cdots,m)$ 0 and the rest non-negative in the left-hand side of expression (16) is found, the left-hand side beccmes new evaluation function z. That is;

$$z=z_0-\theta(z_j-c_j) \tag{17}$$

is established. If $(z_j-c_j)\leqq 0(j=1,\cdots,m)$ is satisfied, $z\geqq z_0$ is established. This improves the maximization problem. The solution obtained when $(z_j-c_j)\geqq 0$ is established is the optimum solution. The minimization problem is the reverse of the above.

Step 3

$$\theta = \min \frac{x_{n0+i}}{x_{n0+i,j}}$$

is selected and assigned to expression (15);

$$x_{n0+1}a_{n0+1}+\cdots+x_{n0+i-1}a_{n0+i-1}+x_{n0+i+1}a_{n0+i+1}+\cdots+x_{n0+m}a_{n0+m}+x_ja_j=S \tag{18}$$

$$\begin{cases} x_{n0+k} = x_{n0+k} - \theta x_{n0+k,j} = x_{n0+k} - \dfrac{x_{n0+i} \cdot x_{n0+k,j}}{x_{n0+i,j}} \quad (k=1,2,\cdots,i-1,i+1,m) \\ x_j = \theta = \dfrac{x_{n0+i}}{x_{n0+i,j}} \end{cases} \tag{19}$$

x_{n0+k}, x_j becomes positive and the end point moves to

$X' = (0, \cdots, x_j, \cdots, x_{n0+1}, \cdots, x_{n0+i-1}, x_{n0+i+1}, x_{n0+m})$.

(4) Example

The following program handles a minimization problem involving six variables.

```
      DIMENSION A(100,200),B(100),C(200),X(200),W(1600)
   10 READ(5,1000,END=30) NS
      NW=NS*80
      READ(5,1010) (W(I),I=1,NW)
1000 FORMAT(I10)
1010 FORMAT(80A1)
      WRITE(6,1020) (W(I),=I=1,NW)
1020 FORMAT(1H ,80A1)
      READ(5,1040) M1,M2,M3,NO,INDEX
      M=M1+M2+M3
      N=NO+M+M2
      DO 20 I=1,M
      READ(5,1030) (A(I,I1),I1=1,NO)
   20 CONTINUE
      READ(5,1030)(B(I2),I2=1,M)
      READ(5,1030) (C(I2),I2=1,N)
1030 FORMAT(8F10.0)
1040 FORMAT(5I10)
      EPS=1.E-5
      CALL SIMPLX(A,100,M1,M2,M3.NO,B,C,INDEX,OF,X,EPS,ILL)
      WRITE(6,1050) ILL,OF
      WRITE(6,1060) (X(I),I=1,NO)
      WRITE(6,1070)(C(I),I=1,N)
      GO TO 10
   30 STOP
1050 FORMAT(1HO,'ILL=',I5,' OBJECT FUNCTION VALUE IS',E15.
     *6)
1060 FORMAT(1H ,5E12.5)
1070 FORMAT(1H ,' ZJ-CJ' /5E12.5)
      END
```

Example of input

```
      6
***   SUBJECT TO
  X1+3X2-X3+2X5>=7.0
   -2X2+4X3+X4       >=12
     -4X2+3X3+8X5+X6>=10
       AND     X(1<=I<=6)>=0
     MINIMIZE    F(X)=X2-3X3+2X5
  3                      6     -1
1.0     3.0   -1.0   0.0    2.0    0.0
0.0    -2.0    4.0   1.0    0.0    0.0
0.0    -4.0    3.0   0.0    8.0    1.0
7.0    12.0   10.0
```

174

```
    0.0      1.0    -3.0    0.0     2.0     0.0
```

Example of output

```
     ***    SUBJECT TO
       X1+3X2-X3+2X5>=7.0
         -2X2+4X3+X4        >=12
            -4X2+3X3+8X5+X6>=10
              AND      X(1<=I<=6)>=0
            MINIMIZE     F(X)=X2-3X3+2X5
ILL=     0  OBJECT FUNCTION VALUE IS  -0.110000E+02
 0.0           0.40000E+01 0.50000E+01 0.0       0.0
 0.0
ZJ-CJ
-20.20000E+00 0.0          0.0         -0.80000E+00-0.24000E+0
ZJ-CJ
0.0          -0.20000E+00-0.80000E+00 0.0
```

(5) Note

If a problem contains many conditional expressions for the number of variables, it is recommended to transform the problem into a dual problem.

Example: SUBJECT to

$$3u_2 \geqq 1$$

$$2u_1+4u_2 \geqq 4$$

$$2u_1+ u_2 \geqq 3$$

$$3u_1 \geqq 3$$

$$4u_1+2u_2 \geqq 4$$

$$u_1+2u_2 \geqq 1$$

and, under $u_i(i=1,2) \geqq 0$, minimize $g=2u_1+2u_2$ is transformed to:

$$2x_2+2x_3+3x_4+4x_5+x_6 \leqq 2$$

$$3x_1+4x_2+x_3+2x_5+2x_6 \leqq 2$$

and, it is substituted by maximize $f=x_1+4x_2+3x_3+3x_4+4x_5+x_6$ under $x_i(i=1,2,\cdots,6) \geqq 0$.

$$u_1=4/3,\ u_2=1/3$$

$$x_2=1/3,\ x_3=2/3$$

$$z=g(u)=f(x)=10/3$$

(1989. 01. 25)

17. Special data processing

176

MAXCOS/I/D (Maximum Condensed Sort of a Vector)

Maximum Condensed Sort of a Vector

Programmed by	Michiyo Kato, September 1982
Format	Subroutine Language: FORTRAN; Size: 19, 19, and 19 lines respectively.

(1) Outline

If a real vector A and its element count K1 are given, MAXCOS/I/D rearranges them based on the Quicksort method without any overlap, and gives to KG the order in the vector A.

(2) Directions

CALL MAXCOS/I/D(A, K1, K2, KG, ILL)

Argument	Type and kind (*1)	Attribute	Content
A	Real type One-dimensional array	Input/output	Vector A to be processed. This routine rearranges the elements in decreasing order without any overlap.
K1	Integer type	Input	Number of elements of vector A to be processed.
K2	Integer type	Output	Number of elements of vector A that do not overlap with another.
KG	Integer type	Output	Original number of element A that does not overlap with another.
ILL	Integer type	Output	ILL=0: Normal termination. ILL=30000: Abnormal input.

*1 For MAXCOI (MAXCOD), all real types should be integer types (double precision real types).

(1987.08.11)

MAXS/I/D (Maximum element of vector and its number)

Maximum Element of Vector and Its Number

Programmed by	Michiyo Kato; September 1982
Format	Subroutine Language; FORTRAN Size; 17 lines each

(1) Outline

When real vector A is given, MAXS, MAXI, and MAXD each return the maximum element value and its number.

(2) Directions

CALL MAXS/I/D(A, K1, A1, K2, ILL)

Argument	Type and kind (*1)	Attribute	Content
A	Real type One-dimensional array	Input	Target vector for processing.
K1	Integer type	Input	Number of elements of target vector A.
A1	Real type	Output	Maximum element of vector A.
K2	Integer type	Output	Number of maximum element A1.
ILL	Integer type	Output	ILL=0: Normal termination. ILL=30000: No vector is specified for A.

*1 For MAXI (MAXD), any real type should be changed to an integer type (double precision real type).

(1987.08.11)

178

MINCOS/I/D (Condensed sorting in ascending order and element numbers)

Minimum Condensed Sort of a Vector

Programmed by	Michiyo Kato; September 1982
Format	Subroutine Language; FORTRAN Size; 22 lines each

(1) Outline

When real vector A is given, MINCOS, MINCOI, and MINCOD each sort the vector elements in ascending order without overlap and assign them sequential numbers within vector A.

(2) Directions

CALL MINCOS/I/D(A, K, K1, KG, ILL)

Argument	Type and kind (*1)	Attribute	Content
A	Real type One-dimensional array	Input/output	Target vector for processing. The vector elements are sorted by this routine in ascending order without overlap.
K	Integer type	Input	Number of elements of target vector A.
K1	Integer type	Output	Number of elements of A which are not overlapped.
KG	Integer type one-dimensional array	Output	Ascending sequential number assigned to a non-overlapping element.

Argument	Type and kind (*1)	Attribute	Content
ILL	Integer type	Output	ILL=0: Normal termination. ILL=30000: Input error.

*1 For MINCOI (MINCOD), any real type should be changed to an integer type (double precision real type).

(1987.08.11)

180

MINS/I/D (Minimum element of vector and its number)

Minimum Element of Vector and Its Number

Programmed by	Michiyo Kato; September 1982
Format	Subroutine Language; FORTRAN Size; 17 lines each

(1) Outline

When real vector A is given, MINS, MINI, and MIND each return the minimum element value of the vector and its number.

(2) Directions

CALL MINS/I/D(A, K1, A1, K2, ILL)

Argument	Type and kind (*1)	Attribute	Content
A	Real type One-dimensional array	Input	Target vector for processing.
K1	Integer type	Input	Number of elements of target vector A.
A1	Real type	Output	Minimum element of vector A.
K2	Integer type	Output	Number of minimum element A1.
ILL	Integer type	Output	ILL=0: Normal termination. ILL=30000: No vector is specified for A.

*1 For MINI (MIND), any real type should be changed to an integer type (double precision real type).

(1987.08.11)

181

MINSOS/I/D (Sort of vector in ascending order and element number)

Minimum Sort of a Vector

Programmed by	Michiyo Kato; September 1982
Format	Subroutine Language; FORTRAN Size; 13 lines each

(1) Outline

When real vector A and the number of its elements K are given, MINSOS, MINSOI, and MINSOD each sort them in ascending order based on the Hoare's Quicksort method and returns sequential numbers in original vector A to KG.

(2) Directions

CALL MINSOS/I/D(A, K, KG, ILL)

Argument	Type and kind (*1)	Attribute	Content
A	Real type One-dimensional array	Input/output	Target vector for processing. The vector elements are sorted by this routine in ascending order.
K	Integer type	Input	Number of elements of target vector A.
KG	Integer type one-dimensional array	Output	Number in ascending order of original element of vector A.
ILL	Integer type	Output	ILL=0: Normal termination. ILL=30000: Input error.

*1 For MINSOI (MINSOD), any real type should be changed to an integer type (double precision

182

real type).

(1987.08.11)

PRODUS/I/D (The Product Set of Two Vectors)

The Product Set of Two Vectors

Programmed by	Michiyo Kato, September 1982
Format	Subroutine Language: FORTRAN; Size: 38, 38, and 38 lines respectively.

(1) Outline

If real vectors A and B are given, PRODUS/I/D rearranges them in increasing order of the product set A∩B, and outputs their order that was in the original vector.

(2) Directions

CALL PRODUS/I/D(A, B, K1, K2, K3, KG, KS, ILL)

Argument	Type and kind (*1)	Attribute	Content
A	Real type One-dimensional array	Input/output	Vector A to be processed. This routine outputs A∩B to A in increasing order. Number of arrays≧K1
B	Real type One-dimensional array	Input	Vector to be processed.
K1	Integer type	Input	K1 = number of elements of A to be processed + K2.
K2	Integer type	Input	Number of element of vector B.

184

Argument	Type and kind (*1)	Attribute	Content
K3	Integer type	Output	Number of elements of A∩B.
KG	One-dimensional array of integer type.	Output	Gives the original element numbers that are rearranged in increasing order of the elements of a product set A∩B. However, the element number of B follows that of A. Size: K1.
KS	One-dimensional array of integer type.	WORK	Vector of size K2.
ILL	Integer type	Output	ILL=0: Normal termination. ILL=30000: Abnormal input.

*1 For PRODUI (PRODUD), all real types should be integer types (double precision real types).

(1987.08.11)

SETPACK(MINS/I/D,MAXS/I/D,MINCOS/I/D,MINSOS/I/D,MAXCOS/I/D,SUMSOS/I/D,SUBSOS/I/D,PRODUS/I/D) (Set operation program package)

Set Operation Package

Programmed by	Michiyo Kato; September 1982
Format	Subroutine Language; FORTRAN Size; 531 lines in total

(1) Outline

These subroutines decide the minimum and maximum elements of vector data and their numbers, sort vector data in ascending or descending order, condense vector data by deleting redundant elements, or generate a set sum, difference, and product vector of two sets of vector data.

(2) Directions

These subroutines are explained individually.

(3) Calculation method

All these subroutines, other than those which determine the minimum and maximum elements of vector data, first arrange vector data by the Hoare's quick sort method[1] and then perform their individual processing.

(4) Performance

The following table shows the result of the speed test conducted using FORTRAN77 OPT(3) on the M-200.

No.	Subroutine name	Calculation time (ms)	Size of vector
I	MINS/I/D	6	10000
II	MAXS/I/D	6	10000
III	MINCOS/I/D	115	10000
IV	MINSOS/I/D	102	10000
V	MAXCOS/I/D	120	10000

186

No.	Subroutine name	Calculation time (ms)	Size of vector
VI	SUMSOS/I/D	182	10000+5000
VII	SUBSOS/I/D	192	10000+5000
VIII	PRODUS/I/D	189	10000+5000

(5) Note

1. A subroutine of the SORTxy type[2] for vector sorting is called.

Bibliography

1) D. E. Knuth, "The Art of Computer Programming, "Vol. 3, Sorting and Searching, Chapter 5:Sorting, Addison-Wesley (1972)

2) Refer to SORTxK, SORTxy, and SRTVxz.

(1987. 08. 05) (1987. 08. 21)

SORTPACK(SORTxK,SORTxy,SRTVxz) (Internal sorting of scalar or vector data)

(x:C,D,I,Q,S), (y:C,D,I,Q,S), (z:C,D,I,Q,S)

Internal Sorting of Scalar or Vector Data

(SORTPACK)

Programmed by	Ichizo Ninomiya: July 1982
Format	Subroutine Language; FORTRAN77 Size; SORTxk: About 118 lines SORTxy: About 136 lines SRTVxz: About 30 lines

(1) Outline

Each of these subroutines sorts scalar data $\{a_i, i=1,\cdots,N\}$ or vector data $\{v_i, i=1,\cdots,N\}$ in computer main storage based on key data $\{k_i, i=1,\cdots,N\}$.

SORTxK, SORTxy, and SORTVxz are used to sort key data, scalar data, and vector data respectively. Their names depend on key data type x, scalar data type y, and vector data type z respectively.

x is C, D, I, or S. y and z are C, D, I, Q, or s. Where

- C: Character type (arbitrary number of bytes $\leq$256)
- D: Double precision (8 bytes) real type
- I: 4-byte integer type
- Q: Quadruple precision (16 bytes) real type
- S: Single precision (4 bytes) real type

(2) Directions

```
CALL SORTCK(N, AK, IND)
```

188

CALL SORTIS(N, AK, A, IND)

CALL SRTVSD(N, AK, V, KV, LV, IW, W, IND)

Argument	Type and kind (*)	Attribute	Content
N	Integer type	Input	Number of data items. N≧1
AK	One-dimensional array *	Input/output	Key data. One-dimensional array of size N. This is sorted.
A	One-dimensional array **	Input/output	Scalar data. One-dimensional array of size N. This is sorted.
V	Two-dimensional array **	Input/output	Vector data. Two-dimensional array of size L×N. This is sorted.
KV	Integer type	Input	Adjustable dimension of V. KV≧L
LV	Integer type	Input	Number of columns of V. Length of vector data. LV≧1
IW	Integer type One-dimensional array	Work area	One-dimensional array of size N.
W	One-dimensional array **	Work area	One-dimensional array of size LV. Same type as V.
IND	Integer type	Input/output	Input IND=0: Data is sorted in ascending order of the key. IND≠0: Data is sorted in descending order of the key. However, character type data is always sorted in ascending order regardless of the IND value. Output IND=0: Normal termination. IND=30000: Argument error.

* The type is determined by the fifth character of the subroutine name. C, D, I, Q, S

** The type is determined by the sixth character of the subroutine name. C, D, I, Q, S

C: Character type Q: Quadruple precision real type

D: Double precision real type S: Real type

I: Integer type

(3) Calculation method

Each of these subroutines uses the Hoare's Quick Sort method that is the highest in speeds for In-place sorting requiring no work area.

1. Sorting key data only

The following discusses sorting only key data in ascending order. The entire data is regarded as a file and the consecutive parts are regarded as subfiles. The outline of the algorithm is as follows:

(1) Empty the stack.

(2) When the size of the subfile being processed becomes small than 16, go to (6).

(3) Check three data items at the top, middle, and end of a subfile, then exchange them so that the largest comes last, the smallest comes to the center, and the intermediate comes first. Use the first data as the standard.

(4) Check data items one by one alternately from the beginning and end of a subfile, then move those smaller than the standard data to the first half and those larger than the standard data to the latter half. After completing this operation, the standard data is transfered to the correct position and the subfile is divided into two subfiles (first and latter halves).

(5) Of the two subfiles, put the first and last locations of the larger subfile into the stack. Make the smaller subfile current and go to (2).

(6) Sort the subfile by straight insertion sorting.

(7) If the stack is empty, terminates sorting. Otherwise, fetch the first and last locations of the subfile that has been saved last, then go to (2).

To sort key data in descending order, change the signs of the entire key data, sort it in ascending order, then change the signs back again.

2. Sorting key data and scalar data

Add a procedure, which moves dependent scalar data along with key data, to the procedure that sorts key data only.

3. Sorting key data and vector data

190

Assign key data natural sequence numbers and sort it as dependent scalar data along with key data. The final location of vector data is known from number data. So, by moving vector data cyclically, almost all vector data is moved to the correct location by one movement.

(4) Performance

We performed a speed test for sorting real scalar data in ascending order with the real key. We used FORTRAN77 (OPT3) on the M-200 for this test. The table shows test results. For comparison, the table also shows the results of the subroutine operations based on seven kinds of in-place sorting methods appearing in the book of Knuth[1]. The key data used was generated by uniform random numbers.

METHOD	NAME	100	1000	10000
BUBBLE SORT	BBSRTS	4MS	436MS	43318MS
STRAIGHT SELCTION SORT	SSSRTS	4MS	367MS	36562MS
STRAIGHT INSERTION SORT	SISRTS	2MS	191MS	18783MS
MERGE EXCHANGE SORT	MESRTS	3MS	48MS	870MS
DIMINISHING INCREMENT SORT	DISRTS	2MS	21MS	363MS
HEAP SORT	HPSRTS	1MS	18MS	246MS
QUICK SORT (FORT77)	SORTSS	1MS	11MS	130MS
QUICK SORT (ASSEMBLER)	SORTSS	1MS	8MS	102MS

(5) Example

1. Only character type key data C consisting of four characters is sorted.

```
CHARACTER*4 C(1000)
.....
.....
CALL SORTCK(N,C,IND)
.....
.....
END
```

2. Real number key data AK and number data NO are sorted in ascending order.

```
      REAL*4 AK(10000)
      INTEGER*4 NO(10000)
      ..........
      ..........
      DO 10 I=1,N
   10 NO(I)=I
      IND=0
      CALL SORTSI(N,AK,NO,IND)
      ..........
      ..........
      END
```

3. Double precision eigenvalues E are sorted together with eigenvectors V in descending order of eigenvalues E.

```
      REAL*8 E(50),V(100,50),W(100)
      INTEGER*4 IW(50)
      ..........
      N=50
      KV=100
      L=100
      IND=1
      CALL SRTVDD(N,E,V,KV,L,IW,W,IND)
      ..........
      ..........
      END
```

(6) Notes

1. Order of the key data depends on the type of data. So, it is important to select a subroutine appropriate to the type of target key data.

2. Scalar and vector data other than key data is not compared but is just moved based on the key data. Therefore, different types of data can be mixed if they match in the number of bytes. For instance, a program for double precision real data can be used to sort 8-byte complex data. To sort vector data consisting of four 4-byte numbers, it is better to use a scalar sort subroutine by regarding it as quadruple precision (16 bytes) scalar data rather than vector data.

Bibliography

1) D. E. Knuth; "The Art of Computer Programming", Vol. 3, Sorting and Searching, Chapter 5:Sorting, Addison-Wesley (1972).

(1989. 01. 25)

192

SUBSOS/I/D (The Difference Set of Two Vectors)

The Difference Set of Two Vectors

Programmed by	Michiyo Kato, September 1982
Format	Subroutine Language: FORTRAN; Size: 36, 36, and 36 lines respectively.

(1) Outline

If real vectors A and B are given, SUBSOS/I/D obtains the difference set A-B in increasing order of elements, and outputs the number in the original vector.

(2) Directions

CALL SUBSOS/I/D(A, B, K1, K2, K3, KG, KS, ILL)

Argument	Type and kind (*1)	Attribute	Content
A	Real type One-dimensional array	Input/output	Vector A to be processed. This routine outputs A-B to A in increasing order.
B	Real type One-dimensional array	Input	Vector B to be processed.
K1	Integer type	Input	K1 = number of elements of A to be processed + K2.
K2	Integer type	Input	Number of element of vector B.

Argument	Type and kind (*1)	Attribute	Content
K3	Integer type	Output	Number of elements of A-B.
KG	One-dimensional array of integer type	Output	Gives the original element numbers that are rearranged in increasing order of the elements of a difference set A-B. However, the element number of B follows that of A. Size: K1.
KS	One-dimensional array of integer type	WORK	Vector of size K2.
ILL	Integer type	Output	ILL=0: Normal termination. ILL=30000: Abnormal input.

*1 For SUBSOI (SUBSOD), all real types should be integer types (double precision real types).

(1987.08.11)

194

SUMSOS/I/D (The Sum Set of a Two Vectors)

The Sum Set of a Two Vectors

Programmed by	Michiyo Kato, September 1982
Format	Subroutine Language: FORTRAN; Size: 15, 15, and 15 lines respectively.

(1) Outline

If real vectors A and B are given, SUMSOS/I/D obtains the sum set A∪B in increasing order, and gives the original vector numbers.

(2) Directions

CALL SUMSOS/I/D(A, B, K1, K2, K3, KG, ILL)

Argument	Type and kind (*1)	Attribute	Content
A	Real type One-dimensional array	Input/output	Vector A to be processed. This routine outputs A∪B to A in increasing order.
B	Real type One-dimensional array	Input	Vector B to be processed.
K1	Integer type	Input	K1: Number of elements of A to be processed + K2.
K2	Integer type	Input	Number of elements of vector B.

Argument	Type and kind (*1)	Attribute	Content
K3	Integer type	Output	Number of elements of A∪B.
KG	Integer type	Output	The elements of a sum set is rearranged in increasing order, and the original element number is given. However, the element number of B follows that of A.
ILL	Integer type	Output	ILL=0: Normal termination. ILL=30000: Abnormal input.

*1 For SUMCOI (SUMCOD), all real types should be integer types (double precision real types).

(1987.08.11) (1987.08.27)

196

18. Figure display application program

CONRM (Contour Line)

Contour Line

Programmed by	Kazuo Hatano
Format	Subroutine Language: FORTRAN; Size: undefined

For details, refer to p. 170-172 of "Guide to Figurative Output (Revised)," page 242, August 1985."

(1987. 07. 27)

198

CONTOR (Contour Line)

Contour Line

Programmed by	Kazuo Hatano
Format	Subroutine Language: FORTRAN; Size: Undefined

For details, refer to pages 165 and 166 of "Guide to Figurative Output (Revised), page 242, August 1985."

(1987.07.27)

CONT1M (Contour Line)

Contour Line

Programmed by	Kazuo Hatano
Format	Subroutine Language: FORTRAN; Size: undefined

For details, refer to p. 173-174 of "Guide to Figurative Output (Revised), page 242, August 1985."

(1987.07.27)

200

CONT1S (Contour Line)

Contour Line

Programmed by	Kazuo Hatano
Format	Subroutine Language: FORTRAN; Size: Undefined

For details, refer to pages 168 and 169 of "Guide to Figurative Output (Revised), page 242, August 1985."

(1987.07.27)

CTL2 (Contour line display program)

Contour Line

Programmed by	Tomikazu Kamiya and Akihiko Yamamoto, October 1984
Format	Subroutine Language: FORTRAN; Size: 327 lines

(1) Outline

(a) Draws the contour line using dashed lines.

(b) Capable of specifying the height of contour lines to be drawn and the type of the corresponding pen and line (solid or dashed) by simple repetition method.

(c) Capable of indicating the function values (numerical strings) in an optional size, digits, and steps at the same time for each contour line level.

(2) Directions

CALL CTL2(A, IA, NX, NY, LX, LY, ZLH, DZ, BEXY, KINO, DH, HN, JN, N)

Argument	Type and kind (*1)	Attribute	Content
A			Two-dimensional array. It contains the values of mesh points.
IA			Two-dimensional work array. The size should match that of A.
NX NY			Adjustable dimensions of the first subscript of arrays A and IA. Adjustable dimensions of the second subscript of arrays A and IA.

202

Argument	Type and kind (*1)	Attribute	Content
LX LY			Number of mesh points (LX≧NK) in the X direction (the first subscript). Number of mesh points (LY≧NY) in the Y direction (the second subscript).
ZLH			One dimensional array of ZLH(2). The lower limit of the contour line level is put in ZLH(1), and the upper limit is put in ZLH(2).
DZ			Spacing between contour lines to be drawn (DZ≠0.0).
BEXY			One-dimensional array declared as BEXY(4). The following values should be input. The X coordinates at mesh point A(1,1) are put in BEXY(1), and the X coordinates at mesh point A(LX,LY) are put in BEXY(3). The Y coordinates at mesh point A(1,1) are put in BEXY(2), and the Y coordinates at mesh point A(LX,LY) are put in BEXY(4).

Argument	Type and kind (*1)	Attribute	Content
KINO	Integer type One-dimensional array	Input	This argument is declared as KINO(4), and the drawing conditions should be set as follows: KINO(1): The relationship between the level of contour lines and the kind of lines should be specified. =0: All lines are solid lines. =1: All lines are broken lines. =2: Positive or 0 parts are solid lines. Negative parts are broken lines. =3: Negative parts are solid lines. Positive or 0 parts are broken lines. = others: The kinds of pens, that is, "|JN(1)|, |JN(2)|, |JN(3)|, and ···" are repeated for each DZ from ZLH(1). If JN(K)>0, solid lines are drawn. If JN(K)<0, broken lines are drawn. KINO(2): Whether to draw the value at each level of contour lines should be specified. =0: Values are not drawn (contour lines only). =0: Values are drawn at the same time. They are plotted at each step specified by KINO (3). KINO(3): When the values at each level of contour lines are to be drawn at the same time, the number of steps should be specified. About one-third of LX and LY should be specified. When the pen shifts to the contour line of the next level before the current number of steps reaches that of KINO(3), the values at the current level are drawn. KINO(4): The number of digits of numbers to be drawn at KINO(2)≠0 should be specified. This argument is the same as the argument N of the NUMBER routine.
DH	Real type	Input	When broken contour lines are to be drawn, the length of a broken line segment should be specified. This argument uses the DASHP routine internally, and has the same meaning as the argument DLEN. The length of broken lines varies according to how much contour lines are congested or BEXY, LX, or LEY.
HN	Real type One-dimensional array	Input	The height of numeric strings to be drawn at KINO(2)≠0 should be specified according to each level. If the level of contour lines corresponds to the kind of the pen|JN(K)|, the numeric strings of height HN(K) is drawn. HN should be specified at the same time as JN. If N=1, a simple variable can be specified. (Size: N)

Argument	Type and kind (*1)	Attribute	Content
JN	Integer type One-dimensional array	Input	When a contour line is to be plotted, the kind of the pen and line should be specified by the iterative method. Assume that the level of contour lines is ZLH(1)+DZ·(K-1). If the pen is |JN(MOD(K, N))|, and the segment JN(MOD(K, N))>0, a solid line is drawn. If the pen is |JN(MOD(K, N))|, and the segment JN(MOD(K, N))<0, a broken line is drawn. If MOD(K, N)=0, N is assumed. However, if KINO(1)=0 to 3, it is preceded as the kind of the segment. Thus, it is meaningless to specify a negative JN(K). For example, if (1) KINO(1)=1, JN(1)=1, JN(2)=2, JN(3)=3, and N=3 and (2) KINO(1)=4, JN(1)=-1, JN(2)=-2, JN(3)=-3, and N=3 are specified as arguments, they have the same meaning. (Size: N)
N	Integer type	Input	Size of HN and JN. If N=1, HN and JN can be single variables.

Notes: 1. For the relationship between the level of contour lines and the kind of lines, the two specification methods, KINO(1) and JN, are available. However, KINO (1) should be preferentially specified.

2. If KINO(1)=0 and KINO(2)=0, the function is quite the same as CTLKTL except the specification of the pen.

3. If a value other than zero is given to only a specific HN(K), and a zero is given to others,

only a specific level of contour lines can stand out.

(1987. 08. 11) (1988. 06. 02)

206

SOLMOR (Solid Figure)

Solid Figure

Programmed by	Kazuo Hatano
Format	Subroutine Language: FORTRAN; Size: undefined

For details, refer p. 179-181 of "Guide to Figurative Output (Revised), page 242, August 1985."

(1987.07.27)

SOLRM (Solid Figure)

Solid Figure

Programmed by	Kazuo Hatano
Format	Subroutine Language: FORTRAN; Size: undefined

For details, refer p.181-186 of "Guide to Figurative Output (Revised), page 242, August 1985."

(1987.07.27)

208

TRIMAP (Contour Line)

Contour Line

Programmed by	Yoshio Sato
Format	Subroutine Language: FORTRAN; Size: undefined

For details, refer to p. 175-179 of "Guide to Figurative Output (Revised), page 242, August 1985."

(1987. 07. 27)

19. Others

210

BITLOGIC(IAND,IOR,IEOR,IEQV,INEQV,INAND,INOR,IMPLY,IDIF,ICOMPL,INOT)

(Bitwise logical operations for 4-byte data)

Bitwise Logical Operations for 4-Byte Data

Programmed by	Ichizo Ninomiya; August 1980
Format	Function Language; Assembler Size; 71 lines

(1) Outline

BITLOGIC is a set of functions that performs bit-by-bit logical operations for 4-byte data.

IAND: Logical product INAND: Negation of logical product

IOR: Logical add INOR: Negation of logical add

IEOR: Exclusive OR IMPLY: Implication

IEQV: Equivalence IDIF: Logical difference

INEQV: Negation of equivalence (same as IEOR) ICOMPL, INOT: Negation

(2) Directions

4-byte integer type function. Whatever argument can be used if it is a 4-byte number. ICOMPL and INOT each use one variable, and the other functions each use two variables.

K=IAND (X, Y) K=INOR (X, Y)

K=IOR (X, Y) K=IMPLY (X, Y)

K=IEOR (X, Y) K=IDIF (X, Y)

K=IEQV (X, Y) K=ICOMPL(X)

K=INEQV(X, Y) K=INOT (X)

K=INAND(X, Y)

The truth table of these functions is shown for reference.

X	Y	IAND	IOR	IEOR (INEQV)	IEQV	INAND	INOR	IMPLY	IDIF	ICOMPL(X) (INOT)
0	0	0	0	0	1	1	1	1	0	1
0	1	0	1	1	0	1	0	1	0	1
1	0	0	1	1	0	1	0	0	1	0
1	1	1	1	0	1	0	0	1	0	0

(3) Note

The logical elements for logical operators .NOT., .AND., and .OR. must be logical variables (constants, array elements, and relational expressions). BITLOGIC function performs bit-by-bit logical operations for 4-byte data.

(1987.06.24) (1987.08.21)

212

IBITCT (Count of Bits "1" of a 4-Byte Data)

Count of Bits "1" of a 4-Byte Data

Programmed by	Ichizo Ninomiya, November 1982
Format	Function Language: Assembler; Size: 59 lines

(1) Outline

IBITCT counts the number of "1" bits in the binary representation of four byte data.

(2) Directions

1. IBITCT(N)

   N is the four-byte data of an arbitrary type. The value of the function is a four-byte integer type.

2. Limit of argument

   All data is acceptable if it is a four-byte data.

(3) Calculation method

The number of bits per byte is counted and added up from the table.

(4) Performance

The M-200 requires about 2 micro seconds per data.

(1987.08.11)

IBITRV (Bit Reversal of a 4-Byte Data)

Bit Reversal of a 4-Byte Data

Programmed by	Ichizo Ninomiya, November 1982
Format	Function Language: Assembler; Size: 66 lines

(1) Outline

IBITRV puts a 4-byte data bit pattern in the reverse order.

(2) Directions

1. IBITRV(N)

   N is the 4-byte data of an arbitrary type. The value of the function is a 4-byte integer type.

2. Limit of argument

   All data is acceptable if it is a four byte data.

(3) Calculation method

The data is put in the reverse order using the table of byte data reversed in advance.

(4) Performance

The M-200 requires about two micro seconds per data.

(1987.08.11)

214

IGCD (Greatest common divisor of two integers)

GCD of Two Integers

Programmed by	Ichizo Ninomiya; March 1987
Format	Function Language; FORTRAN77 Size; 14 lines

(1) Outline

IGCD calculates the greatest common divisor of two integers.

(2) Directions

1. IGCD(M, N)

M and N each are an arbitrary integer-type expression.

2. Range of argument

Any value can be specified for the argument.

(3) Calculation method

The Euclid's Algorithm is used.

IGCD determines the greatest common divisor of the absolute values of two numbers regardless of whether they are positive or negative. If one of two numbers is 0, 0 is the greatest common divisor.

(1987.08.11)

PRIME (Generation of prime number table)

Generation of Prime Number Table

Programmed by	Ichizo Ninomiya; March 1987
Format	Subroutine Language; FORTRAN77 Size; 51 lines

(1) Outline

PRIME calculates all prime numbers below a given positive integer.

(2) Directions

CALL PRIME(N, NSIZE, NPRIME, IPRIME, ILL)

Argument	Type and kind	Attribute	Content
N	Integer type	Input	Given integer. N≧2.
NSIZE	Integer type	Input	Size of array IPRIME for prime number table. NSIZE≧5.
NPRIME	Integer type	Output	Number of prime numbers obtained. Number of prime numbers below N or NSIZE, whichever is smaller.
IPRIME	Integer type one-dimensional array	Output	Prime number table. The prime numbers obtained are stored in ascending order starting with 2.
ILL	Integer type	Output	Error code. ILL=0: Normal. ILL=30000: Argument error.

(3) Calculation method

PRIME repeats dividing integers K, other than multiples of 2, 3, and 5, by prime numbers

greater than 7 and smaller than $\sqrt{K}$. If it cannot be exactly divided by any prime number, it is a prime number.

(4) Note

If NSIZE is too small, only the first NSIZE number of prime numbers out of those equal to or less than N is obtained.

From the viewpoint that the number of prime numbers equal to or less than N is about N/logN, it is recommended to specify rather a large value for NSIZE. If N is small, make NSIZE=N for safety.

Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., pp.821-873.

(1987.08.11)

PRMFAC (Prime factor decomposition of an integer)

Prime Factor Decomposition of an Integer

Programmed by	Ichizo Ninomiya; March 1987
Format	Subroutine Language; FORTRAN77 Size; 68 lines

(1) Outline

PRMFAC decomposes a given positive integer N into prime factors and uses them to calculate Euler's function $\varphi(N)$ and Mobius' function $\mu(N)$.

(2) Directions

CALL RMFAC(N, NFAC, IFAC, IEXP, IEULER, IMOEBS, ILL)

Argument	Type and kind	Attribute	Content
N	Integer type	Input	Given integer. N≧1.
NFAC	Integer type	Output	Kind of prime factors of N.
IFAC	Integer type one-dimensional array	Output	The Jth prime factor in ascending order of J that lies in 1≦J≦NFAC enters IFAC(J).
IEXP	Integer type one-dimensional array	Output	The exponent of prime factor IFAC(J) of J that lies in 1≦J≦NFAC enters IEXP(J).

218

Argument	Type and kind	Attribute	Content
IEULER	Integer type	Output	Euler's function $\varphi(N)$.
IMOBBS	Integer type	Output	Mobius' function $\mu(N)$.
ILL	Integer type	Output	Error code. ILL=0: Normal. ILL=30000: N<1.

(3) Calculation method

1. When the subroutine is called for the first time, the routine PRIME is called to generate the prime number table consisting of 46340 or less prime numbers.

2. Whether K is a factor of N is examined, for all prime numbers K in the range of $2 \leq K \leq \sqrt{N}$.

3. When N is decomposed to $N=\prod_{i=1}^{m} P_i^{e_i}$:

$$\varphi(N)=N \cdot \prod_{i=1}^{m} (p_i-1)/p_i,$$

$\mu(N)=1$ ····· When N=1.

$\mu(N)=(-1)^m$ ··· When $e_i=1$ is met for any i.

$\mu(N)=0$ ····· When $e_i \geqq 2$ is met for certain i.

(4) Note

NFAC is 10 or less for all integers up to $2^{31}-1$.

Bibliography

1) Handbook of Mathematical Functions, Dover, N.Y., pp. 821-873.

(1987. 08. 11)

220

RANDOM/DRANDM (Generation of uniform random number)

Generation of Uniform Random Number

Programmed by	Ichizo Ninomiya; 1980
Format	Function Language; Assembler Size; 25 lines

(1) Outline

RANDOM and DRANDM each generate uniform random numbers in interval (0, 1) using the congruence method. RANDOM is a single precision routine, and DRANDM is a double precision routine. DRANDM needs to be declared as double precision.

(2) Directions

X=RANDOM(IX)

Argument	Type and kind	Attribute	Content
IX	Integer type	Input	Initial value. Non-negative integer. To generate one random number sequence, this argument must be set to 0 at the second and subsequent call.

(3) Example

```
      DIMENSION X(100)
      X(1)=RANDOM(1)
      DO   1 I=2,100
    1 X(I)=RANDOM(0)
```

(4) Note

When a large amount of random numbers are generated, RANU2 of Fujitsu SSL II is more efficient than these functions.

221

(1987.06.26)

222

ROUND/DROUND (Round-off of real numbers)

Round-off of Real Numbers

Programmed by	Ichizo Ninomiya; April 1981
Format	Function Language; Assembler Size; 18 lines each

(1) Outline

ROUND rounds off a double precision real number into a single precision real number. DROUND rounds off a quadruple precision real number into a double precision real number.

(2) Directions

ROUND(D), DROUND(Q)

D and Q are arbitrary double and quadruple expressions respectively. DROUND needs to be declared as double precision.

(1987.06.24)