2.6 Bessel Functions I_0, I_1, K_0 and K_1

A. Purpose

These subprograms compute values of the modified (hyperbolic) Bessel functions of the first kind, I_0 and I_1 and the modified (hyperbolic) Bessel functions of the second kind, K_0 and K_1. These functions are discussed in [1] and [2].

B. Usage

B.1 Program Prototype, Single Precision

REAL X, BI0, BI1, BK0, BK1

INTEGER INFO, IWANT

To compute I_0 *and/or* K_0:
Assign values to X and IWANT, and

CALL SBI0K0(X, BI0, BK0, IWANT, INFO)

To compute I_1 *and/or* K_1:
Assign values to X and IWANT, and

CALL SBI1K1(X, BI1, BK1, IWANT, INFO)

B.2 Argument Definitions

X [in] Argument of function. Require X > 0 for the K functions.

BI0, BI1, BK0, BK1 [out] Function values, depending on IWANT.

IWANT [in] Specification of the desired functions:

$= \ \ 1$: BIn $= I_n(x)$
$= \ \ 2$: BK$n = K_n(x)$
$= \ \ 3$: BIn $= I_n(x)$ and BK$n = K_n(x)$
$= -1$: BIn $= e^{-|x|}I_n(x)$
$= -2$: BK$n = e^x K_n(x)$
$= -3$: BIn $= e^{-|x|}I_n(x)$ and BK$n = e^x K_n(x)$.

INFO [out] indicates the status on termination. INFO = 0 means the computation was successful. See Section E for the meaning of nonzero values of INFO.

B.3 Modifications for Double Precision

For double precision usage, change the REAL type statement to DOUBLE PRECISION, and change the subroutine names to DBI0K0 and DBI1K1, respectively.

C. Examples and Remarks

The listing of DRSBESI0 and ODSBESI0 gives an example of using these subprograms to evaluate the Wronskian identity

$$\zeta(x) = x[I_0(x)K_1(x) + I_1(x)K_0(x)] - 1 = 0$$

D. Functional Description

The functions I_n and K_n are a pair of linearly independent solutions for the differential equation

$$x^2\frac{d^2w}{dx^2} + x\frac{dw}{dx} - \left(x^2 + n^2\right) w = 0$$

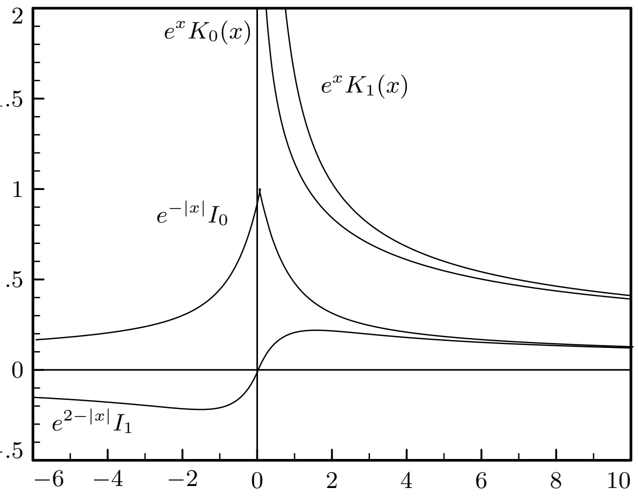
The functions I_0 and I_1 are defined for all real x. The function I_0 is even, and I_1 is odd. They are monotone increasing functions of $|x|$, approaching $e^{|x|}(2\pi|x|)^{-\frac{1}{2}}$ as $|x| \to \infty$. The functions $e^{-|x|}I_0(x)$ and $e^{-|x|}I_1(x)$ are monotone decreasing functions of $|x|$, approaching $(2\pi|x|)^{-\frac{1}{2}}$ as $|x| \to \infty$.

The functions K_0 and K_1 have real values for positive real x, approach $+\infty$ as $x \to 0^+$ and have complex values for negative real x. As $x \to 0^+$, $K_0(x) \to -\ln(x)$ and $K_1(x) \to 1/x$. $K_n(x)$ is a monotone decreasing function of x, approaching $e^{-x}(\pi/2x)^{\frac{1}{2}}$ as $x \to +\infty$.

The K subprograms treat $x \le 0$ as an error condition. If the complex values of K_0 and K_1 for negative x are desired they may be computed from the formulae

$$\begin{aligned} K_0(x) &= \ \ K_0(-x) - i\pi I_0(-x) \\ K_1(x) &= -K_1(-x) - i\pi I_1(-x), \end{aligned} \qquad x < 0$$

where i denotes the imaginary unit. See Equation 9.6.31 in [1].



The computer approximations for these functions, except for $K_0(x)$ when $x < 2$, were developed by L. W. Fullerton, [3] and [4], using functional forms involving sine, cosine, square root, logarithm, and Chebyshev polynomial approximations. The computer approximations for $K_0(x)$ when $x < 2$ were developed by W. J. Cody of Argonne National Laboratory using rational polynomial approximations. Where Chebyshev polynomial approximations are used, these subprograms select the polynomial degrees to adapt to machine accuracy of up to 30 decimal places. The coefficients given by Cody have only 21 digits. If 30 decimal place precision is needed, approximations due to Fullerton are available, but for precisions up to 16 decimal places, they are somewhat less accurate than the approximations due to Cody, because Fullerton computes $K_0(x)$ using $I_0(x)$, $\ln(x)$ and a Chebyshev polynomial (see formula 9.6.13 in [1]); the errors in $I_0(x)$ and $\ln(x)$, together with the error in the Chebyshev polynomial approximation, are greater than the errors in Cody's approximation.

The single precision subprograms for $I_n(x)$ and $K_n(x)$ were tested on an IBM PC/AT (using IEEE arithmetic) by comparison with the corresponding double precision subprograms over various argument ranges. Each interval was divided into 10,000 subintervals, and a point was randomly selected in each subinterval. The relative precision of IEEE single precision arithmetic is $\rho = 2^{-23} \approx 1.19 \times 10^{-7}$. The test results may be summarized as follows.

		Relative Error, in units of ρ		
Function	**Interval**	**Max.**	**Mean**	**Std. Dev.**
$I_0(x)$	[0, 2.5]	1.21	0.23	0.17
	[2.5, 5]	1.15	0.35	0.25
	[5, 15]	1.31	0.40	0.26
	[15, 80]	1.21	0.39	0.26
$I_1(x)$	[0, 2.5]	1.01	0.23	0.17
	[2.5, 5]	1.49	0.38	0.28
	[5, 15]	1.39	0.42	0.28
	[15, 80]	1.60	0.41	0.29
$K_0(x)$	[0, 1.8]	2.15	0.41	0.37
	[1.8, 2.1]	2.28	0.43	0.34
	[2.1, 5]	1.33	0.35	0.26
	[5, 15]	1.36	0.34	0.24
	[15, 80]	1.33	0.36	0.26
$K_1(x)$	[0, 1.8]	2.01	0.28	0.23
	[1.8, 2.1]	2.30	0.41	0.31
	[2.1, 5]	1.11	0.31	0.22
	[5, 15]	1.09	0.31	0.22
	[15, 80]	1.12	0.31	0.22

The double precision subprograms for $I_n(x)$ were tested on an IBM PC/AT (using IEEE arithmetic) by comparison with extended precision subprograms over various argument ranges. The double precision routines for $K_n(x)$ were tested on an IBM PC/AT (using IEEE arithmetic) by comparison with an independent double precision procedure, due to Cody, that creates a purified argument, that is, one for which the function can be evaluated without significant error. Each interval was divided into 2000 subintervals, and a point was randomly selected in each subinterval. The relative precision of IEEE double precision arithmetic is $\rho = 2^{-52} \approx 2.22 \times 10^{-16}$. The test results may be summarized as follows.

		Relative Error, in units of ρ		
Function	**Interval**	**Max.**	**Mean**	**Std. Dev.**
$I_0(x)$	[0, 2.5]	1.38	0.24	0.31
	[2.5, 5]	1.36	0.34	0.25
	[5, 9]	5.14	0.63	0.79
$I_1(x)$	[0, 2.5]	1.04	0.24	0.18
	[2.5, 5]	1.41	0.34	0.26
	[5, 9]	1.97	0.45	0.33
$K_0(x)$	[0, 1.8]	3.37	0.07	0.77
	[1.8, 2.1]	2.75	0.03	0.81
	[2.1, 5]	2.94	0.54	0.52
	[5, 15]	2.24	0.58	0.50
	[15, 80]	2.78	0.64	0.55
	[80, 225]	3.21	0.66	0.59
$K_1(x)$	[0, 1.8]	3.41	0.74	0.63
	[1.8, 2.1]	4.75	1.07	0.85
	[2.1, 5]	5.31	1.49	1.10
	[5, 15]	8.84	3.43	2.22
	[15, 80]	57.67	16.51	13.62
	[80, 225]	115.95	49.63	33.35

The poor accuracy reported for $I_0(x)$ in [5, 9] occurs largely in [8.5, 9], and may be due to a questionable reference value. Other reports of poor accuracy are probably justified.

Having no adequate reference function for $I_0(x)$ and $I_1(x)$ for $x > 9.0$, subprograms for these functions were tested by computing the Wronskian relation $I_0(x)K_1(x) + I_1(x)K_0(x)$ and comparing the result to $1/x$. The test results may be summarized as follows.

	Relative Error, in units of ρ		
Interval	**Max.**	**Mean**	**Std. Dev.**
$[9, 80]$	1.52	0.43	0.30
$[80, 700]$	1.66	0.43	0.30

The errors here are much smaller than the errors reported above for $K_0(x)$ and $K_1(x)$. The very accurate values of the Wronskian relation suggest a functional dependence between the algorithms implemented in the different hyperbolic Bessel function subprograms.

References

1. Milton Abramowitz and Irene A. Stegun, **Handbook of Mathematical Functions**, *Applied Mathematics Series 55*, National Bureau of Standards (1966).

2. J. F. Hart et al., **Computer Approximations**, J. Wiley and Sons, New York (1968) Section 6.8.

3. L. Wayne Fullerton, **Program Library Write–up**

Argument range testing in SBI0K0 or DBI0K0:

INFO	Argument Range	BI0	BK0	IWANT	Reason
-3	$x \le 0$	correct	Ω	$\lvert\text{IWANT}\rvert \ge 2$	$K_0(x)$ and $e^x K_0(x)$ are imaginary
-2	$x > \text{XI}_0\text{MAX} > \text{XK}_0\text{MAX}$	Ω	zero	IWANT > 0	$I_0(x)$ overflows, $K_0(x)$underflows
1	$\text{XI}_0\text{MAX} \ge x > \text{XK}_0\text{MAX}$	correct	zero	IWANT ≥ 2	$K_0(x)$ underflows

Argument range testing in SBI1K1 or DBI1K1:

INFO	Argument Range	BI1	BK1	IWANT	Reason
-4	$0 < x < \text{XK}_1\text{MIN}$	correct	Ω	$\lvert\text{IWANT}\rvert \ge 2$	$K_1(x)$ and $e^x K_1(x)$ overflow
-3	$x \le 0$	correct	Ω	$\lvert\text{IWANT}\rvert \ge 2$	$K_1(x)$ and $e^x K_1(x)$ are imaginary
-2	$x > \text{XI}_1\text{MAX} > \text{KX}_1\text{MAX}$	Ω	zero	IWANT > 0	$I_1(x)$ overflows, $K_1(x)$ underflows
1	$\text{XI}_1\text{MAX} \ge x > \text{XK}_1\text{MAX}$	correct	zero	IWANT ≥ 2	$K_1(x)$ underflows

B001. Technical report, Los Alamos Scientific Laboratory (1973).

4. L. Wayne Fullerton, *Portable special function routines*, in Wayne Cowell, editor, **Portability of Numerical Software**, *Lecture Notes in Computer Science 57*, 452–483, Springer Verlag, Berlin (1977).

E. Error Procedures and Restrictions

These subroutines set INFO to indicate the termination status. INFO = 0 means the computation was successful. INFO = -1 means IWANT had an improper value. Any other value means the argument was one for which one of the requested functions does not have a defined real value, or the function value would be outside the underflow or overflow limits of the host system.

The subroutines obtain host exponent limits using R1MACH() or D1MACH() of Chapter 19.1. These are used to compute argument limits, XI_0MAX, etc., using asymptotic formulas — Equations 9.7.1 and 9.2.2 of [1]. For example on a machine with IEEE arithmetic, these argument limits are as listed in the following table:

Bound	Single Precision	Double Precision
XI_0MAX	91.900	713.9869
XK_0MAX	85.337	705.3427
XI_1MIN	2.531×10^{-38}	4.45×10^{-308}
XI_1MAX	91.906	713.9876
XK_1MIN	1.187×10^{-38}	2.247×10^{-308}
XK_1MAX	85.34	705.3434

The different argument range tests and resulting settings of BI0, BK0, BI1, BK1, and INFO are given in the tables above, where Ω denotes the largest representable number.

When INFO $\neq 0$ the error message processor of Chapter 19.2 is called — with LEVEL = 0 if INFO < 0 and with LEVEL = -2 if INFO > 0. The user can alter the default action of the error processor by calling ERMSET of Chapter 19.2.

F. Supporting Information

The source language is ANSI Fortran 77.

Subprograms SBI0K0 and SBI1K1 designed and developed by W. V. Snyder, JPL, 1990, based on earlier subprograms by L. W. Fullerton and W. J. Cody.

Entry	Required Files
DBI0K0	AMACH, DBI0K0, DCSEVL, DERM1, DERV1, DINITS, ERFIN, ERMSG, IERM1, IERV1
DBI1K1	AMACH, DBI1K1, DCSEVL, DERM1, DERV1, DINITS, ERFIN, ERMSG, IERM1, IERV1
SBI0K0	AMACH, ERFIN, ERMSG, IERM1, IERV1, SBI0K0, SCSEVL, SERM1, SERVI, SINITS
SBI1K1	AMACH, ERFIN, ERMSG, IERM1, IERV1, SBI1K1, SCSEVL, SERM1, SERVI, SINITS

DRSBI0K0

```
c      program DRSBI0K0
c>> 1996-05-28 DRSBI0K0 Krogh Moved formats up.
c>> 1994-10-19 DRSBI0K0 Krogh  Changes to use M77CON
c>> 1992-03-18 DRSBI0K0 CLL Added "c" to "program" line above.
c>> 1990-11-21 WVS (edited by CLL)
c      Demonstration driver for single precision hyperbolic Bessel
c      function subprograms.
c
c      Compute the Wronskian relation
c
c      z = x * (I1(x)*K0(x) + I0(x)*K1(x)) - 1.0
c
c      z should be approximately zero.
c
c      ------------------------------------------------------------------
c--S replaces "?": DR?BI0K0, ?BI0K0, ?BI1K1
c      ------------------------------------------------------------------
      real              X, BI0, BK0, BI1, BK1, Z
      integer IX, INFO
c
100   format (3x,a1,7x,4(a5,8x),a1/3x,'-',7x,4('-----',8x),'-')
150   format (1x,f4.1,1x,2(f12.8,1x),4x,'INFINITY',5x,'INFINITY')
200   format (1x,f4.1,1x,4(f12.8,1x),1x,g9.2)
c
      print 100, 'X', 'I0(X)','I1(X)','K0(X)','K1(X)','Z'
c
      x = 0.0e0
      call sbi0k0 (x, bi0, bk0, 1, info)
      call sbi1k1 (x, bi1, bk1, 1, info)
      print 150, x, bi0, bi1
      do 300 ix = 5, 50, 5
         x = ix / 10.0e0
         call sbi0k0 (x, bi0, bk0, -3, info)
         call sbi1k1 (x, bi1, bk1, -3, info)
         z = x * (bi1*bk0 + bi0*bk1) - 1.0e0
         print 200, x, bi0, bi1, bk0, bk1, z
300   continue
      stop
      end
```

ODSBI0K0

```
   X       I0(X)        I1(X)        K0(X)        K1(X)        Z
   -       -----        -----        -----        -----        -
  0.0   1.00000000   0.00000000    INFINITY     INFINITY
  0.5   0.64503527   0.15642083   1.52410924   2.73100948   -0.12E-06
  1.0   0.46575961   0.20791042   1.14446294   1.63615346    0.0
  1.5   0.36743361   0.21903940   0.95820999   1.24316573   -0.60E-07
  2.0   0.30850834   0.21526928   0.84156823   1.03347695    0.0
  2.5   0.27004644   0.20658463   0.75954866   0.90017444   -0.60E-07
  3.0   0.24300034   0.19682670   0.69776160   0.80656350   -0.60E-07
  3.5   0.22280243   0.18739997   0.64902627   0.73646754   -0.60E-07
  4.0   0.20700192   0.17875084   0.60929769   0.68157595    0.0
  4.5   0.19419828   0.17095883   0.57609683   0.63714987    0.12E-06
  5.0   0.18354082   0.16397227   0.54780757   0.60027385    0.0
```