A

IIPBF: a MATLAB toolbox for infinite integrals of product of Bessel functions

J. TILAK RATNANATHER, Johns Hopkins University
JUNG H. KIM, Johns Hopkins University
SIRONG ZHANG, Beihang University
ANTHONY M. J. DAVIS, University of California San Diego
STEPHEN K. LUCAS, James Madison University

A MATLAB toolbox, IIPBF, for calculating infinite integrals involving a product of two Bessel functions $J_a(\rho x)J_b(\tau x)$, $J_a(\rho x)Y_b(\tau x)$ and $Y_a(\rho x)Y_b(\tau x)$, for non-negative integers a, b, and a well behaved function $f(x)$, is described. Based on the Lucas algorithm previously developed for $J_a(\rho x)J_b(\tau x)$ only, IIPBF recasts each product as the sum of two functions whose oscillatory behavior is exploited in the three step procedure of adaptive integration, summation and extrapolation. The toolbox uses customised QUADPACK and IMSL functions from a MATLAB conversion of the SLATEC library. In addition, MATLAB's own quadgk function for adaptive Gauss-Kronrod quadrature results in a significant speed up compared with the original algorithm. Usage of IIPBF is described and thirteen test cases illustrate the robustness of the toolbox. The first five of these and five additional cases are used to compare IIPBF with the BESSELINT code for rational and exponential forms of $f(x)$ with $J_a(\rho x)J_b(\tau x)$. An appendix shows a novel derivation of formulae for three cases.

Categories and Subject Descriptors: G.1.4 [**Numerical Analysis**]: Quadrature and Numerical Differentiation; G.4 [**Mathematics of Computing**]: Mathematical Software

General Terms: Bessel Function, Quadrature

Additional Key Words and Phrases: extrapolation, adaptive quadrature, summation

ACM Reference Format:
Ratnanather, J. T., Kim, J. H., Zhang, S., Davis, A. M. J., and Lucas, S. K. 2011. IIPBF: a MATLAB toolbox for infinite integrals of product of Bessel functions. ACM Trans. Math. Softw. V, N, Article A (January YYYY), 11 pages.
DOI = 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

1. INTRODUCTION

Lucas [1995] developed an algorithm for computing infinite integrals of products of Bessel functions of the first kind. Specifically, for given a well-behaved function $f(x)$, two non-negative integers a, b and two positive real constants ρ, τ, the algorithm computes $\int_0^\infty f(x)B_{a,b,\rho,\tau}(x)dx$ where $B_{a,b,\rho,\tau}(x) = J_a(\rho x)J_b(\tau x)$. Available as a stand alone FORTRAN77 package containing functions from IMSL and QUADPACK libraries, the algorithm has been used in elasticity, electrodynamics, fluid dynamics, biophysics and geo-

Authors' addresses: J. T. Ratnanather, Center for Imaging Science and Institute for Computational Medicine, Whitaker Biomedical Engineering Institute, The Johns Hopkins University, Clark 301, 3400 N. Charles St, Baltimore, MD 21218-2826; J. H. Kim, Dept. of Applied Mathematics and Statistics, The Johns Hopkins University, Baltimore, MD 21218; S. Zhang, School of Mathematical and System Sciences, Beihang University, Beijing, 100191, China; A. M. J. Davis, Dept. of Mechanical and Aerospace Engineering, University of California San Diego, La Jolla, CA 92093-0411; S. K. Lucas, Dept. of Mathematics and Statistics, James Madison University, Harrisonburg, VA 22807.

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

physics to name but a few applications [Craster 1998; Davis and Stone 1998; Pan et al. 2007; Petrov and Schwille 2008; Robinson 2002; Salo et al. 2006; Sherwood 2005; Tartakovsky et al. 2000; Xu et al. 2003]. Briefly, the Lucas algorithm rewrites the product, $B_{a,b,\rho,\tau}$, as the sum of two functions i.e. $h_1 + h_2$ whose oscillatory behaviour is exploited in a three step procedure of (i) Integration, (ii) Summation and (iii) Extrapolation (ISE). ISE was first adopted for integrands involving $J_n(x)$ whose zeros form the subdivision of the range giving rise to a summation of integrals of alternating signs, convergence of which is accelerated via an extrapolation technique [Lucas and Stone 1995]. The decomposition of the product as the sum of two functions results in applying the ISE method twice. The algorithm used modifications of adaptive `IMSL` routines `dqdage` and `dqdagie` for finite and infinite adaptive integration with Gaussian quadrature respectively; further it used `dqelg` in `QUADPACK` [Piessens et al. 1983] as the ε-algorithm and `mWtrans` as the mW transform [Sidi 1988] to deal with the oscillatory components and acceleration of the summation of the integrals, and methods for finding or approximating the zeros of h_1 and h_2. Lucas [1995] obtained results for $J_a(x)J_b(x)$ approaching machine precision with less then 1000 function evaluations compared with a few digits of accuracy for up to 15000 evaluations with the `IMSL` routines.

The increasing usage of `MATLAB` suggests that a version of the Lucas algorithm would be a valuable computational science tool. This can be achieved by taking advantage of the recent availability of `MATLAB` versions of `IMSL` and `QUADPACK` functions via a conversion of the `SLATEC` library [Barrowes 2009]. Further, the algorithm has not previously been adapted for combinations of Bessel functions of the first and second kind.

In what follows, first the development of a `MATLAB` toolbox, `IIPBF`, is described for three types of $B_{a,b,\rho,\tau}$ i.e. $J_a(\rho x)J_b(\tau x)$, $J_a(\rho x)Y_b(\tau x)$ or $Y_a(\rho x)Y_b(\tau x)$. Next a modification of the general Lucas algorithm for dealing with zeros of h_2 for the ε-algorithm and the use of `MATLAB`'s own adaptive Gauss-Kronrod quadrature algorithm (`quadqk`) [Shampine 2008] are described. Results for several test cases are presented. The discussion concludes the paper.

2. METHOD

This section describes the decomposition of $B_{a,b,\rho,\tau}$ as the sum of high and low frequency components respectively given by h_1 and h_2. This is followed by a summary of a modification of the Lucas algorithm in which the zeros of h_1 are used as integral endpoints with the mW transform for the infinite integral involving h_1, and the zeros for h_2 estimated stepwise are used for the ε-algorithm.

2.1. Product decomposition

First $J_a(\rho x)J_b(\tau x)$ is expressed as $h_1(x;a,b,\rho,\tau) + h_2(x;a,b,\rho,\tau)$ where

$$\begin{aligned} h_1(x;a,b,\rho,\tau) &= \frac{1}{2}(J_a(\rho x)J_b(\tau x) - Y_a(\rho x)Y_b(\tau x)), \\ h_2(x;a,b,\rho,\tau) &= \frac{1}{2}(J_a(\rho x)J_b(\tau x) + Y_a(\rho x)Y_b(\tau x)) \end{aligned}$$

with the asymptotic behaviour for large x [Lucas 1995]

$$\left.\begin{aligned} h_1(x;a,b,\rho,\tau) &\sim \frac{1}{\pi\sqrt{\rho\tau}x}\cos\left((\rho+\tau)x - \frac{(a+b+1)\pi}{2}\right) \\ h_2(x;a,b,\rho,\tau) &\sim \frac{1}{\pi\sqrt{\rho\tau}x}\cos\left((\rho-\tau)x - \frac{(a-b)\pi}{2}\right) \end{aligned}\right\}. \quad (1)$$

Second $J_a(\rho x)Y_b(\tau x)$ is expressed as $h_1(x;a,b,\rho,\tau)+h_2(x;a,b,\rho,\tau)$ where

$$\begin{aligned} h_1(x;a,b,\rho,\tau) &= \frac{1}{2}(J_a(\rho x)Y_b(\tau x)+Y_a(\rho x)J_b(\tau x)) \\ h_2(x;a,b,\rho,\tau) &= \frac{1}{2}(J_a(\rho x)Y_b(\tau x)-Y_a(\rho x)J_b(\tau x)) \end{aligned}$$

whose asymptotic behavior is

$$\left.\begin{aligned} h_1(x;a,b,\rho,\tau) &\sim \frac{1}{\pi\sqrt{\rho\tau}x}\sin\left((\rho+\tau)x-\frac{(a+b+1)\pi}{2}\right) \\ h_2(x;a,b,\rho,\tau) &\sim -\frac{1}{\pi\sqrt{\rho\tau}x}\sin\left((\rho-\tau)x-\frac{(a-b)\pi}{2}\right) \end{aligned}\right\}. \tag{2}$$

Third $Y_a(\rho x)Y_b(\tau x)$ is expressed as $h_1(x;a,b,\rho,\tau)+h_2(x;a,b,\rho,\tau)$ where

$$\begin{aligned} h_1(x;a,b,\rho,\tau) &= -\frac{1}{2}(J_a(\rho x)J_b(\tau x)-Y_a(\rho x)Y_b(\tau x)), \\ h_2(x;a,b,\rho,\tau) &= \frac{1}{2}(J_a(\rho x)J_b(\tau x)+Y_a(\rho x)Y_b(\tau x)) \end{aligned}$$

whose asymptotic behaviour is

$$\left.\begin{aligned} h_1(x;a,b,\rho,\tau) &\sim -\frac{1}{\pi\sqrt{\rho\tau}x}\cos\left((\rho+\tau)x-\frac{(a+b+1)\pi}{2}\right) \\ h_2(x;a,b,\rho,\tau) &\sim \frac{1}{\pi\sqrt{\rho\tau}x}\cos\left((\rho-\tau)x-\frac{(a-b)\pi}{2}\right) \end{aligned}\right\}. \tag{3}$$

2.2. Calculating zeros of h_1 an h_2

The ISE method is applied to $f(x)h_1(x)$ and $f(x)h_2(x)$. While the zeros of h_1 are easy to find, those of h_2 can be difficult to compute. Let $h_{1,1}$ and $h_{2,1}$ denote the first zeros of $h_1(x)$ and $h_2(x)$ respectively.

When $\rho=\tau$, the integration domain is split with $[0,ymax]$ for $f(x)B_{a,b,\rho,\tau}(x)$ and $[ymax,\infty]$ for $f(x)h_1(x)+f(x)h_2(x)$. From Eqs. 1-3, $h_2(x)$ is a non-oscillating monotonically decreasing function, extrapolation is used for $h_1(x)$ and thus the zeros of $h_2(x)$ are not needed. $ymax$ is calculated as the first zero of $h_1(x)$ beyond which point $Y_a(x)$ no longer dominates $h_1(x)$ [Olver 1960]. Subsequent zeros for the mW transform [Sidi 1988] are computed using the asymptotic approximations from Equations 1-3.

When $\rho\neq\tau$, the estimate for the zeros of $Y_a(x)$ [Olver 1960] are used initially to establish $h_{1,1}$ and $h_{2,1}$. The domain $[0,\infty]$ is split into $[0,ymin]$, $[ymin,ymax]$ and $[ymax,\infty]$ with $ymin=min(h_{1,1},h_{2,1})$ and $ymax=max(h_{1,1},h_{2,1})$. Subsequent zeros of $h_1(x)$ for the mW transform are computed using the asymptotic approximations from Equations 1-3, while zeros of $h_2(x)$ after $h_{2,1}$ for the ε-algorithm are obtained via stepwise increments of $\pi/|\rho-\tau|$ [Lucas 1995].

2.3. The `IIPBF` Toolbox

By adding (a) to `QUADPACK` function names, `IIPBF` algorithm for $0<a,b<1000$, $\rho,\tau\in\mathbb{R}$ and `type` $=1,2,3$ for one of the three forms of $B_{a,b,\rho,\tau}$ is described in algorithm 1.

The functions `dqagiea`, `dqagea` and `dqelga` were adapted from the `MATLAB` version of the `SLATEC` library [Barrowes 2009]. In particular, `dqagea` and `dqagiea` use the recently developed adaptive Gauss-Kronrod quadrature algorithm (`quadgk`) [Shampine 2008] now available in `MATLAB` instead of `dqk15` and `dqk15i` respectively; `dqelga` is called via a wrapper `epsalg`. The converted `SLATEC` routines [Barrowes 2009] together with the original `mWtrans` algorithm were customised to pass a second function to deal with

ALGORITHM 1: `IIPBF` Algorithm

Input: `type`, a, b, ρ, τ, `abserr` (absolute error tolerance), `relerr` (relative error tolerance)
Output: $I_{a,b,\rho,\tau}$, `neval` (number of evaluations of h_1 and h_2), `esterr` (estimated relative error)
if $\rho = \tau$ **then**
  Calculate $h_{1,1}$ with `abserr`$/3$ and `relerr`$/3$;
  $I_1 = \int_0^{h_{1,1}} f(x)B_{a,b,\rho,\tau}(x)dx$ using `dqagea`;
  $I_2 = \int_{h_{1,1}}^{\infty} f(x)h_2(x;a,b,\rho,\tau)dx$ using `dqagiea`;
  $I_3 = \int_{h_{1,1}}^{\infty} f(x)h_1(x;a,b,\rho,\tau)dx$ using `mWtrans`;
  $I_4 = 0$
else
  Calculate $h_{1,1}$ and $h_{2,1}$ with `abserr`$/4$ and `relerr`$/4$;
  $I_2 = \int_{h_{1,1}}^{\infty} f(x)h_1(x;a,b,\rho,\tau)dx$ using `mWtrans`;
  $I_3 = \int_{h_{2,1}}^{\infty} f(x)h_2(x;a,b,\rho,\tau)dx$ using `epsalg`;
  if $h_{1,1} < h_{2,1}$ **then**
    $I_1 = \int_0^{h_1} f(x)B_{a,b,\rho,\tau}(x)dx$ using `dqagea`;
    $I_4 = \int_{h_{1,1}}^{h_{2,1}} f(x)h_2(x;a,b,\rho,\tau)dx$ using `dqagea`
  else
    $I_1 = \int_0^{h_{2,1}} f(x)B_{a,b,\rho,\tau}(x)dx$ using `dqagea`;
    $I_4 = \int_{h_{2,1}}^{h_{1,1}} f(x)h_1(x;a,b,\rho,\tau)dx$ using `dqagea`
  end
end
$I_{a,b,\rho,\tau} = I_1 + I_2 + I_3 + I_4$

the decomposition of $f(x)B_{a,b,\rho,\tau}(x)$. Available at `http://www.cis.jhu.edu/software/iipbf`, the toolbox is run as follows:

```
[result,est_error,nevals]=IIPBF(f,rho,tau,a,b,abserr,relerr,type);
```

with input parameters:

`f`	=	user-defined function: $f(x)$
`rho`	=	ρ
`tau`	=	τ
`a,b`	=	non-negative integers
`abserr`	=	absolute error tolerance
`relerr`	=	relative error tolerance
`type`	=	product type

and output:

`result`	=	computed integral
`esterr`	=	estimated relative error
`neval`	=	number of function evaluations for `esterr` to fall below tolerance

3. RESULTS

Table I lists the cases that were used to test `IIPBF`. The first three cases are from Lucas [1995] and the remaining ones are simplified forms of those being calculated with the toolbox in a separate study. While evidently faster algorithms exist for simpler integrals, it is important to be able to demonstrate feasibility of `IIPBF`. Fig. 1 and 2 plots actual error and estimated relative error with respect to the number of evaluations of h_1 and h_2 for requested relative error tolerances from 10^{-4} to 10^{-14} in 10 decrements of 0.1.

Table I. Test cases for $f(x)B_{a,b,\rho,\tau}(x)$ with parameters $u, \rho, \tau \in \mathbb{Z}^+$.

Case	$f(x)B_{a,b,\rho,\tau}(x)$	Value
1	$J_0(x)J_1(3x/2)$	$2/3$
2	$J_0(x)J_5(2x)/x^4$	$27/4096$
3	$xJ_0(x)J_{20}(1.1x)/(1+x^2)$	$\approx -6.050747903049 \times 10^{-3}$
4	$J_0(x)J_1(x)/x$	$2/\pi$
5	$J_1(x)J_1(x)/x^2$	$4/3\pi$
6	$xK_0(xu)J_0(\rho x)J_0(\tau x)$	$1/\left(\left(u^2+\rho^2+\tau^2\right)^2-4\rho^2\tau^2\right)^{1/2}$
7	$x^2K_1(xu)J_1(\rho x)J_1(\tau x)$	$4u\rho\tau/\left(\left(u^2+\rho^2+\tau^2\right)^2-4\rho^2\tau^2\right)^{3/2}$
8	$e^{-2ux}J_0(x)Y_0(x)$	$(1/\pi)\int_0^{\pi/2}(1+u^2\cos^2 z)^{-1/2}dz$
9	$xe^{-x^2/u}J_2(x)Y_2(x)$	$\frac{4}{s\pi}-\frac{2}{\pi}-\frac{uK_2(u/2)}{2\pi e^{u/2}}$
10	$x^3e^{-x^2/u}J_2(x)Y_2(x)$	$-\frac{4}{\pi}+\frac{u^2(2+u)K_0(u/2)}{4\pi e^{u/2}}+\frac{u(8+4u+u^2)K_1(u/2)}{4\pi e^{u/2}}$
11	$e^{-ux}Y_0(\rho x)Y_0(\tau x)$	$-\frac{2}{\pi}\int_\rho^\infty \frac{G(y)dy}{\sqrt{y^2-\rho^2}}$ where $\rho > \tau$ and $G(y) = -\frac{2}{\pi(\alpha_1^2+\alpha_2^2)}\left[\alpha_1 \ln \frac{\sqrt{(\alpha_1+u)^2+(\alpha_2+y)^2}}{\tau} + \alpha_2 \arctan\left(\frac{\alpha_2+y}{\alpha_1+u}\right)\right]$ $\alpha_{1,2} = \frac{1}{\sqrt{2}}\left[\sqrt{(u^2+\tau^2-y^2)^2+4u^2y^2} \pm (u^2+\tau^2-y^2)\right]^{1/2}$
12	$J_0(\rho x)J_0(\tau x)$	$(2/\pi\rho)\mathbf{K}(\tau/\rho)$ where $\tau < \rho$
13	$Y_0(\rho x)Y_0(\tau x)$	$(2/\pi\rho)\mathbf{K}(\tau/\rho)$ where $\tau < \rho$

Formulae 4-8 were obtained from Gradshteyn & Ryzhik [2007]; those for cases 9-10 were obtained from Adamchik [1995]; for cases 11-13 see Appendix A for derivation of formulae involving the elliptic function $\mathbf{K}$ *that are simpler than those by Glasser [1974].*

Table II. Additional test cases for comparison of `BESSELINT` and `IIPBF`.

Case	$f(x)B_{a,b,\rho,\tau}(x)$	Value
14	$e^{-2ux}xJ_0(x)J_1(x)$	$\frac{\mathbf{K}-\mathbf{E}}{2\pi\rho\sqrt{\rho^2+u^2}}$
15	$e^{-2ux}[J_0(x)]^2$	$\frac{\mathbf{K}}{\pi\sqrt{\rho^2+u^2}}$
16	$e^{-2ux}[J_1(x)]^2$	$\frac{(2u^2+\rho^2)\mathbf{K}-2(u^2+\rho^2)\mathbf{E}}{\pi\rho^2\sqrt{\rho^2+u^2}}$
17	$e^{-2ux}[xJ_1(x)]^2$	$\frac{3}{4\pi}\int_0^{\pi/2}\frac{\cos^2 z dz}{[u^2+\cos^2 z]^{5/2}}$
18	$\frac{x}{u^2+x^2}J_a(\rho x)J_a(\tau x)$	$\frac{1}{2}\pi i J_a(i\rho u)H_a^{(1)}(i\tau u) \quad a > -1$

`MATLAB` functions for $\mathbf{K}$ *and* $\mathbf{E}$ *which are the complete elliptic integrals of the first and second kind, respectively, of modulus* $\rho/\sqrt{\rho^2+u^2}$ *where* $u \in \mathbb{Z}^+$ *were used; formulae were taken from Gradshteyn & Ryzhik [2007].*

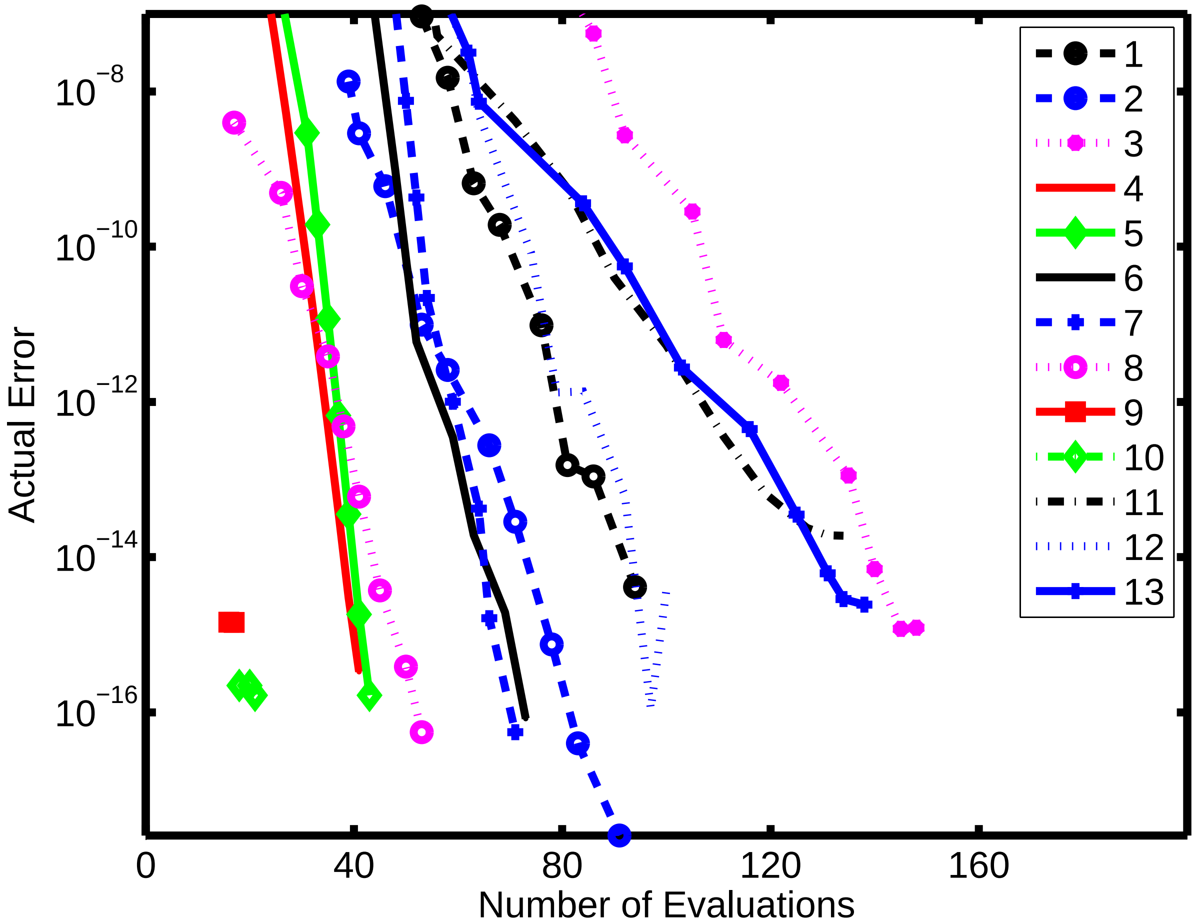

Fig. 1. Comparing actual error to number of function evaluations of h_1 and h_2 with requested relative error tolerances from 10^{-4} to 10^{-14} in 10 decrements of 0.1 for the test cases in table I. In cases 6 and 7, parameters $\{u, \rho, \tau\} = \{1, 2, 1\}$. For cases 8, 9 and 10 $u = 1.5$, 0.2 and 2 respectively. For case 11, $\{u, \rho, \tau\} = \{0.1, 2, 1\}$. For cases 12 and 13, $\{\rho, \tau\} = \{3, 1\}$.

An alternative to `IIBPF` is `BESSELINT` [Van Deun and Cools 2006a; 2008; 2006b] for the integrand $f(x)\Pi_{i=1}^{k} J_{a_i}(\rho_i x)$ where $f(x) = x^s e^{-ux}/(t^2 + x^2)$. Briefly, the algorithm uses asymptotic expansions for J_{a_i} and the incomplete Gamma function [Van Deun and Cools 2006c] to approximate the infinite part of the integral. Therefore it is appropriate to compare the performance and reliability of `IIPBF` with `BESSELINT` for cases 1-5 in table I and five additional cases in table II. The tests were run with `MATLAB` version 7.9.0.529 (R2009b) on Intel Core2 Duo CPU L6700 with 2.66GHz. Table III lists run-time execution, absolute difference between computed values, and estimated relative errors for a requested relative error tolerance of 50`eps`. Clearly while both `BESSELINT` and `IIPBF` give similar values with similar relative errors, the former is faster by a factor of 4-50. Further, `MATLAB`'s profiler tool was used to identify which parts of the code consumed most of the CPU processing. Other than calling `besselj` [Van Deun

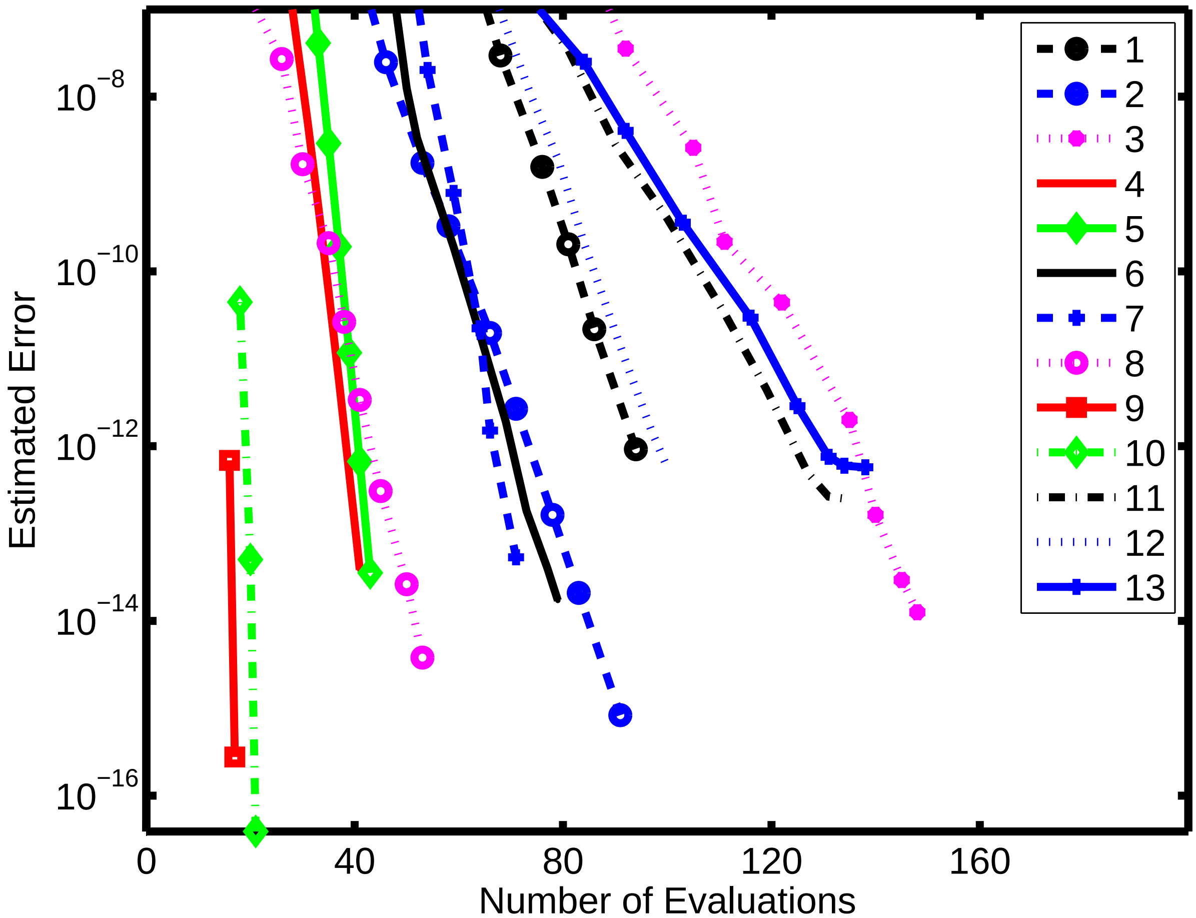

Fig. 2. Comparing estimated error to number of function evaluations of h_1 and h_2 with requested relative error tolerances from 10^{-4} to 10^{-14} in 10 decrements of 0.1 for the test cases in table I. In cases 6 and 7, parameters $\{u, \rho, \tau\} = \{1, 2, 1\}$. For cases 8, 9 and 10 $u = 1.5$, 0.2 and 2 respectively. For case 11, $\{u, \rho, \tau\} = \{0.1, 2, 1\}$. For cases 12 and 13, $\{\rho, \tau\} = \{3, 1\}$.

and Cools 2006b], the bottleneck in the respective codes were `ira` and `mWtrans` which both deal with the infinite range approximation. However the latter consumed more due to the nature of its implementation.

4. DISCUSSION

A MATLAB toolbox, IIPBF, for infinite integration of products of Bessel functions of the first and second kind has been developed and tested.

A key component of IIPBF is the use of adaptive quadrature algorithms. Specifically using `quadgk` in `dqagea` and `dqagiea` enabled error estimates to be derived robustly with significantly less function evaluations than in the original Lucas algorithm by nearly two orders of magnitude for cases 1-3; indeed, the number of function evaluations in an early version of IIPBF were about the same as those by Lucas [1995].

Table III. Comparison of BESSELINT and IIPBF

Case	Time (secs)		Estimated Error		Absolute
	BESSELINT	IIPBF	BESSELINT	IIPBF	Difference
1	0.1189	0.6542	1.67×10^{-16}	2.36×10^{-15}	1.11×10^{-16}
2	0.0351	0.6454	3.95×10^{-16}	8.38×10^{-16}	1.73×10^{-18}
3	0.1830	0.7743	8.87×10^{-15}	1.51×10^{-15}	5.47×10^{-17}
4	0.0163	0.5725	1.74×10^{-16}	2.95×10^{-15}	0
5	0.0140	0.5838	1.31×10^{-16}	1.80×10^{-15}	0
14	0.0992	0.5347	5.56×10^{-15}	7.51×10^{-16}	6.94×10^{-18}
15	0.0088	0.5283	1.29×10^{-14}	2.06×10^{-15}	2.78×10^{-16}
16	0.0176	0.5407	1.33×10^{-14}	8.32×10^{-16}	2.17×10^{-17}
17	0.0106	0.5447	5.56×10^{-15}	1.59×10^{-16}	8.67×10^{-19}
18	0.0385	0.6564	4.13×10^{-13}	2.40×10^{-15}	1.54×10^{-16}

The last column is the absolute difference between the computed values.

Generally, quadgk performs the adaptive subdivision locally and conservatively, is able to process subintervals simultaneously, samples many points within the subintervals and is therefore faster and more reliable than its predecessor [Shampine 2008; Gander and Gautschi 2001; Gonnet 2009]. In addition, dqagiea exploits the ability of quadgk to handle infinite integrals but if convergence is slow, the ε-algorithm is used instead.

Two codes for computing infinite integrals of products of Bessel functions of the first kind have emerged in recent years. One is a FORTRAN code for products of Bessel functions of order 0 or 1 and $f(x) = 1$ or x [Singh and Mogi 2005]. The other is BESSELINT [Van Deun and Cools 2006a; 2006b] with modifications for rational and exponential $f(x)$ [Van Deun and Cools 2008]. When comparing BESSELINT and IIPBF, the latter can be used for any well-behaved $f(x)$ while the rational exponential form of $f(x)$ in the former is used to facilitate stable quadrature over a converging series in the asymptotic part. Quadrature in both cases are adaptive however BESSELINT uses hard-coded Gauss-Legendre with 15 points followed 19 points over sub-intervals determined approximately by the zeros of the integrand. The singularity near $x = 0$ is dealt with internally in the quadgk function for weak logarithmic and algebraic forms via a simple transform whereas extrapolation is used in BESSELINT. The breakpoint beyond which asymptotic properties are used is determined by the first zero of h_1 and h_2 in IIPBF whereas in BESSELINT it is automatically determined via the incomplete gamma function.

Since IIBPF is as accurate as BESSELINT, it will have wider applicability as it would be more laborious to derive asymptotic forms of more general $f(x)$. Eventually, it is possible that IIPBF will be superseded by combining the best features of both codes in which case Hankel functions may be useful [Huybrechs and Vandewalle 2006] in developing the procedure for $Y_{a_i}(\rho_i x)$.

A. FORMULAE FOR CASES 11-13

This appendix describes the derivation of expressions for integrals in cases 11-13 in table I that are found to be simplificatons of those obtained by Glasser [1974].

The elliptic integral $\mathbf{K}(k)$ is defined by

$$\mathbf{K}(k) = \int_0^1 \frac{dx}{\sqrt{(1-x^2)(1-k^2x^2)}} = \int_1^\infty \frac{dX}{\sqrt{(X^2-1)(X^2-k^2)}} \qquad (k^2 < 1).$$

An alternative form, whose integrand has a strictly positive denominator, is

$$\mathbf{K}(k) = \int_0^\infty \frac{dt}{\sqrt{2(1-k^2)\cosh t + 2(1+k^2)}}.$$

Since

$$\int_0^\infty Y_0(\tau x)\cos xy\, dx = -\frac{H(y-\tau)}{\sqrt{y^2-\tau^2}} = -\int_0^\infty J_0(\tau x)\sin xy\, dx,$$

where $H(u)$ denotes the Heaviside unit function, we may use the Parseval equations for cosine and sine transforms to show that

$$\int_0^\infty Y_0(\tau x)Y_0(\rho x)\, dx = \int_0^\infty J_0(\tau x)J_0(\rho x)\, dx = \frac{2}{\pi\rho}\mathbf{K}(\tau/\rho) \qquad (\rho > \tau),$$

which is demonstrated below to be a simpler form of Eq. (10) in Glasser [1974].

The above use of cosine transforms can be extended to show that

$$\int_0^\infty e^{-ux}Y_0(\tau x)Y_0(\rho x)\, dx = -\frac{2}{\pi}\int_\rho^\infty \frac{G(y)dy}{\sqrt{y^2-\rho^2}} \qquad (\rho > \tau),$$

where

$$G(y) = \int_0^\infty e^{-ux}Y_0(\tau x)\cos xy\, dx. \qquad (u, \tau, y > 0).$$

The formula,

$$\int_0^\infty e^{-ux}Y_0(\tau x)\, dx = -\frac{2}{\pi\sqrt{u^2+\tau^2}}\ln\left[\frac{\sqrt{u^2+\tau^2}+u}{\tau}\right],$$

enables $G(y)$ to be evaluated by replacing u by $u - iy$ and taking the real part of the result. If $\sqrt{(u+iy)^2+\tau^2} = \alpha_1 + i\alpha_2$, then $u, y, \tau > 0$ implies $\alpha_1, \alpha_2 > 0$ where

$$\alpha_1, \alpha_2 = \frac{1}{\sqrt{2}}\left[\sqrt{(u^2+\tau^2-y^2)^2+4u^2y^2} \pm (u^2+\tau^2-y^2)\right]^{1/2}.$$

After some algebra, it is found that

$$G(y) = -\frac{2}{\pi(\alpha_1^2+\alpha_2^2)}\left[\alpha_1 \ln\frac{\sqrt{(\alpha_1+u)^2+(\alpha_2+y)^2}}{b} + \alpha_2 \arctan\left(\frac{\alpha_2+y}{\alpha_1+u}\right)\right].$$

Hence $G(y) = O(y^{-2}\ln y)$ as $y \to \infty$. The known expressions for G when $u = 0$ or $y = 0$ are recovered.

Consistency with most of Glasser's formulae is demonstrated as follows. Since

$$\int_0^\infty K_0(\rho x)\cos xy\, dx = \frac{\pi}{2\sqrt{\rho^2+y^2}},$$

the Parseval formula gives

$$\int_0^\infty Y_0(\tau x)K_0(\rho x)\,dx = -\int_\tau^\infty \frac{dy}{\sqrt{(y^2+\rho^2)(y^2-\tau^2)}} = -\frac{1}{\sqrt{\rho^2+\tau^2}}\mathbf{K}\left(\frac{\rho}{\sqrt{\rho^2+\tau^2}}\right),$$

after setting $y^2 = (\rho^2+\tau^2)X^2 - \rho^2$. This agrees with the first of Glasser's Eqs (8) and confirms that his K denotes the elliptic integral $\mathbf{K}$. Similarly, with $\rho > \tau$,

$$\int_0^\infty K_0(\tau x)K_0(\rho x)\,dx = \frac{\pi}{2}\int_0^\infty \frac{dy}{\sqrt{(y^2+\rho^2)(y^2+\tau^2)}} = \frac{\pi}{2\rho}\mathbf{K}\left(\sqrt{1-\frac{\tau^2}{\rho^2}}\right),$$

after setting $y^2 = \rho^2X^2 - \rho^2$. This appears to contradict the second of Glasser's Eq.(8) but the hypergeometric function identity,

$$F(\alpha, \alpha-\beta+1/2; \beta+1/2, z^2) = (1+z)^{-2\alpha}F\left[\alpha, \beta; 2\beta, \frac{4z}{(1+z)^2}\right],$$

yields, with $\alpha = \beta = 1/2$ and $k < 1$,

$$\mathbf{K}(k) = \frac{\pi}{2}F(1/2, 1/2; 1, k^2) = \frac{\pi}{2(1+k)}F\left(1/2, 1/2; 1, \frac{4k}{(1+k)^2}\right) = \frac{1}{1+k}\mathbf{K}\left(\frac{2\sqrt{k}}{1+k}\right).$$

In particular,

$$\mathbf{K}\left(\frac{\rho-\tau}{\rho+\tau}\right) = \frac{\rho+\tau}{2\rho}\mathbf{K}\left(\sqrt{1-\frac{\tau^2}{\rho^2}}\right) \qquad (\rho > \tau),$$

$$\mathbf{K}(\tau/\rho) = \frac{\rho}{\rho+\tau}\mathbf{K}\left(\frac{2\sqrt{\rho\tau}}{\rho+\tau}\right) \qquad (\rho > \tau),$$

which shows the equivalence of Glasser's second Eq. (8) and Eq. (10) and the above concise formulae. Since Glasser's Eq. (9) is used only to obtain Eq. (10), there is no need to be concerned with its derivation.

REFERENCES

ADAMCHIK, V. 1995. The evaluation of integrals of Bessel functions via G-function identities. *J. Comp. Appl. Math. 64*, 283–290.

BARROWES, B. 2009. *The SLATEC library converted into MATLAB functions*. http://www.mathworks.com/matlabcentral/fileexchange/14535.

CRASTER, R. V. 1998. Scattering by cracks beneath fluid-solid interfaces. *J. Sound Vib. 209,* 2, 343–372.

DAVIS, A. M. J. AND STONE, H. A. 1998. Slow translation, rotation or oscillation of a disk in a rotating fluid: Effect of a plane wall or another disk. *Q. J. Mech. App. Math. 51,* Part 4, 495–513.

GANDER, W. AND GAUTSCHI, W. 2001. Adaptive quadrature: revisited. *BIT 40*, 84–101.

GLASSER, M. L. 1974. Some definite integrals of the product of two Bessel functions of the second kind: (order zero). *Math. Comp. 28*, 613–615.

GONNET, P. 2009. Adaptive quadrature re-revisited. Ph.D. thesis, ETH Zurich.

GRADSHTEYN, I. S. AND RYZHIK, I. M. 2007. *Gradshteyn and Ryzhik's Table of Integrals, Series, and Products* 7th Ed. Academic Press. A. Jeffrey and D. Zwillinger (Eds.).

HUYBRECHS, D. AND VANDEWALLE, S. 2006. On the evaluation of highly oscillatory integrals by analytic continuation. *SIAM J. Num. Anal. 44*, 1026–1048.

LUCAS, S. K. 1995. Evaluating infinite integrals involving products of Bessel functions of arbitrary order. *J. Comp. App. Math. 64*, 269–282.

LUCAS, S. K. AND STONE, H. A. 1995. Evaluating infinite integrals involving Bessel functions of arbitrary order. *J. Comp. App. Math. 64*, 217 – 231.

OLVER, F. W. J., Ed. 1960. *Royal Society Mathematical Tables Vol. 7, Bessel Functions Part III, Zeros and Associated Values*. Cambridge University Press, Cambridge.

PAN, E., BEVIS, M., HAN, F., ZHOU, H., AND ZHU, R. 2007. Surface deformation due to loading of a layered elastic half-space: a rapid numerical kernel based on a circular loading element. *Geophys. J. Int. 171,* 1, 11–24.

PETROV, E. P. AND SCHWILLE, P. 2008. Translational diffusion in lipid membranes beyond the Saffman-Delbruck approximation. *Biophys. J. 94,* 5, L41–L43.

PIESSENS, R., DONCKER-KAPENGA, E. D., UBERHUBER, C. W., AND KAHAUER, D. K. 1983. *QUADPACK, a sub-routine package for automatic integration*. Springer-Verlag, Berlin.

ROBINSON, N. I. 2002. An isotropic elastic medium containing a cylindrical borehole with a rigid plug. *Int. J. Solids Struct. 39,* 19, 4889–4904.

SALO, J., WEI-SALLABI, H., AND VAINIKAINEN, P. 2006. Statistical analysis of the multiple scattering radio channel. *IEEE Trans. Ant. Prop. 54,* 11, Part 1, 3114–3124.

SHAMPINE, L. F. 2008. Vectorized adaptive quadrature in MATLAB. *J. Comp. App. Math. 211*, 131–140.

SHERWOOD, J. D. 2005. Optimal probes for withdrawal of uncontaminated fluid samples. *Phys. Fluids 17*, 083102.

SIDI, A. 1988. A user-friendly extrapolation method for oscillatory infinite integrals. *Math. Comp. 51*, 249–266.

SINGH, N. P. AND MOGI, T. 2005. Electromagnetic response of a large circular loop source on a layered earth: A new computation method. *Pure and Applied Geophysics 162*, 181–200.

TARTAKOVSKY, D. M., MOULTON, J. D., AND ZLOTNIK, V. A. 2000. Kinematic structure of minipermeameter flow. *Water Resources Research 36*, 2433–2442.

VAN DEUN, J. AND COOLS, R. 2006a. Algorithm 858: Computing infinite range integrals of an arbitrary product of Bessel functions. *ACM Trans. Math. Software 32*, 580–596.

VAN DEUN, J. AND COOLS, R. 2006b. A Matlab implementation of an algorithm for computing integrals of products of Bessel functions. *LNCS 4151*, 284–295.

VAN DEUN, J. AND COOLS, R. 2006c. A stable recurrence for the incomplete gamma function with imaginary second argument. *Numer. Math. 104*, 445–456.

VAN DEUN, J. AND COOLS, R. 2008. Integrating products of Bessel functions with an additional exponential or rational factor. *Comp. Phys. Comm. 178*, 578–590.

XU, Y. S., LIN, J., LI, H. M., AND WU, F. M. 2003. Analysis for the potential function of the digital microstructure image of porous media. *Comm. Theor. Phys. 40,* 4, 393–394.