For \( n \) larger than a dozen or so, \texttt{betai} is a much better way to evaluate the sum in (6.4.12) than would be the straightforward sum with concurrent computation of the binomial coefficients. (For \( n \) smaller than a dozen, either method is acceptable.)

**CITED REFERENCES AND FURTHER READING:**


## 6.5 Bessel Functions of Integer Order

This section and the next one present practical algorithms for computing various kinds of Bessel functions of integer order. In §6.7 we deal with fractional order. In fact, the more complicated routines for fractional order work fine for integer order too. For integer order, however, the routines in this section (and §6.6) are simpler and faster. Their only drawback is that they are limited by the precision of the underlying rational approximations. For full double precision, it is best to work with the routines for fractional order in §6.7.

For any real \( \nu \), the Bessel function \( J_\nu(x) \) can be defined by the series representation

\[
J_\nu(x) = \left( \frac{1}{2} \right)^\nu \sum_{k=0}^{\infty} \frac{(-1/4x^2)^k}{k!\Gamma(\nu+k+1)}
\]  

The series converges for all \( x \), but it is not computationally very useful for \( x \gg 1 \).

For \( \nu \) not an integer the Bessel function \( Y_\nu(x) \) is given by

\[
Y_\nu(x) = J_\nu(x) \cos(\nu \pi) - J_{-\nu}(x) \sin(\nu \pi)
\]

The right-hand side goes to the correct limiting value \( Y_n(x) \) as \( \nu \) goes to some integer \( n \), but this is also not computationally useful.

For arguments \( x < \nu \), both Bessel functions look qualitatively like simple power laws, with the asymptotic forms for \( 0 < x \ll \nu \)

\[
J_\nu(x) \sim \frac{1}{\Gamma(\nu+1)} \left( \frac{1}{2} x \right)^\nu \quad \nu \geq 0
\]

\[
Y_0(x) \sim \frac{2}{\pi} \ln(x)
\]

\[
Y_\nu(x) \sim -\frac{\Gamma(\nu)}{\pi} \left( \frac{1}{2} x \right)^{-\nu} \quad \nu > 0
\]
Bessel functions $J_0(x)$ through $J_3(x)$ and $Y_0(x)$ through $Y_2(x)$. For $x > \nu$, both Bessel functions look qualitatively like sine or cosine waves whose amplitude decays as $x^{-1/2}$. The asymptotic forms for $x \gg \nu$ are
\begin{align*}
J_\nu(x) &\sim \sqrt{2/\pi x} \cos \left(x - \frac{1}{2} \nu \pi - \frac{1}{4} \pi\right) \\
Y_\nu(x) &\sim \sqrt{2/\pi x} \sin \left(x - \frac{1}{2} \nu \pi - \frac{1}{4} \pi\right)
\end{align*}
(6.5.4)
In the transition region where $x \sim \nu$, the typical amplitudes of the Bessel functions are on the order
\begin{align*}
J_\nu(\nu) &\sim \frac{2^{1/3}}{3^{2/3} \Gamma(\frac{2}{3}) \nu^{1/3}} \sim \frac{0.4473}{\nu^{1/3}} \\
Y_\nu(\nu) &\sim -\frac{2^{1/3}}{3^{1/6} \Gamma(\frac{2}{3}) \nu^{1/3}} \sim -\frac{0.7748}{\nu^{1/3}}
\end{align*}
(6.5.5)
which holds asymptotically for large $\nu$. Figure 6.5.1 plots the first few Bessel functions of each kind.

The Bessel functions satisfy the recurrence relations
\begin{align*}
J_{n+1}(x) &= \frac{2n}{x} J_n(x) - J_{n-1}(x) \\
Y_{n+1}(x) &= \frac{2n}{x} Y_n(x) - Y_{n-1}(x)
\end{align*}
(6.5.6)
and
(6.5.7)
As already mentioned in §5.5, only the second of these (6.5.7) is stable in the direction of increasing $n$ for $x < n$. The reason that (6.5.6) is unstable in the
direction of increasing \( n \) is simply that it is the same recurrence as (6.5.7): A small amount of "polluting" \( Y_n \) introduced by roundoff error will quickly come to swamp the desired \( J_n \), according to equation (6.5.3).

A practical strategy for computing the Bessel functions of integer order divides into two tasks: first, how to compute \( J_0, J_1, Y_0, \) and \( Y_1 \), and second, how to use the recurrence relations stably to find other \( J \)'s and \( Y \)'s. We treat the first task first:

For \( x \) between zero and some arbitrary value (we will use the value 8), approximate \( J_0(x) \) and \( J_1(x) \) by rational functions in \( x \). Likewise approximate by rational functions the "regular part" of \( Y_0(x) \) and \( Y_1(x) \), defined as

\[
Y_0(x) - \frac{2}{\pi} J_0(x) \ln(x) \quad \text{and} \quad Y_1(x) - \frac{2}{\pi} \left[ J_1(x) \ln(x) - \frac{1}{x} \right] \tag{6.5.8}
\]

For \( 8 < x < \infty \), use the approximating forms \((n = 0, 1)\)

\[
J_n(x) = \sqrt{\frac{2}{\pi x}} \left[ P_n \left( \frac{8}{x} \right) \cos(X_n) - Q_n \left( \frac{8}{x} \right) \sin(X_n) \right] \tag{6.5.9}
\]

\[
Y_n(x) = \sqrt{\frac{2}{\pi x}} \left[ P_n \left( \frac{8}{x} \right) \sin(X_n) + Q_n \left( \frac{8}{x} \right) \cos(X_n) \right] \tag{6.5.10}
\]

where

\[
X_n \equiv x - \frac{2n + 1}{4} \pi \tag{6.5.11}
\]

and where \( P_0, P_1, Q_0, \) and \( Q_1 \) are each polynomials in their arguments, for \( 0 < 8/x < 1 \). The \( P \)'s are even polynomials, the \( Q \)'s odd.

Coefficients of the various rational functions and polynomials are given by Hart [1], for various levels of desired accuracy. A straightforward implementation is

FUNCTION bessj0(x)
REAL bessj0, x
REAL ax, xx, z
DOUBLE PRECISION p1, p2, p3, p4, p5, q1, q2, q3, q4, q5, r1, r2, r3, r4,
* r5, r6, s1, s2, s3, s4, s5, s6
We'll accumulate polynomials in double precision.
SAVE p1, p2, p3, p4, p5, r1, r2, r3, r4, r5, r6,
* s1, s2, s3, s4, s5, s6
** DATA p1, p2, p3, p4, p5, r1, r2, r3, r4, r5, r6,
* ** r6, s1, s2, s3, s4, s5, s6
\[ \frac{\text{data}}{1.0} \]
** DATA x1, x2, x3, x4, x5, x6
\[ \text{data} \]
** if(abs(x).lt.8.)then
Direct rational function fit.
** y=x**2
bessj0=(r1+y*(r2+y*(r3+y*(r4+y*(r5+y*r6)))))
** /((a1+y*(a2+y*(a3+y*(a4+y*(a5+y*r6)))))
** \]
\]

else
Fitting function (6.5.9).
ax=abs(x)
z=8./ax
y=x**2
xx=ax-.785398164
bessj0=sqrt(.636619772/ax)*(cos(xx)*(p1+y*(p2+y*(p3+y*(p4+y

\]

\]
FUNCTION bessy0(x)
REAL bessy0, x
C USES bessj0

Returns the Bessel function $Y_0(x)$ for positive $x$.

DOUBLE PRECISION p1, p2, p3, p4, p5, q1, q2, q3, q4, q5, r1, r2, r3, r4,
  r5, r6, s1, s2, s3, s4, s5, s6
SAVE p1, p2, p3, p4, p5, q1, q2, q3, q4, q5, r1, r2, r3, r4,
  r5, r6, s1, s2, s3, s4, s5, s6
DATA p1, p2, p3, p4, p5/1.d0, -0.1098628627e-2, 0.2734510407d-4,
  -0.2073370639d-5, 0.2093887211d-6/, q1, q2, q3, q4, q5/0.1562499995d-1,
  -0.1430488765d-3, -0.691147651d-5, -0.7621999995d-1, -0.934945152d-7/,
  r1, r2, r3, r4, r5, r6/2957821389.0d0, 7062834065.0d0, -512359803.6d0,
  10879881.29d0, -86327.92757d0, 40076544269.0d0, 745294964.8d0,
  7109466.438d0, 47447.26470d0, 1.030246d0, 1.0d0/
if(x.lt.8.)then
  Rational function approximation of (6.5.8).
  y=x**2
  bessy0=(r1+y*(r2+y*(r3+y*(r4+y*(r5+y*r6)))))/(s1+y*(s2+y*(s3+y*(s4+y*(s5+y*s6))))+0.636619772*bessj0(x)*log(x))
else
  Fitting function (6.5.10).
  z=8./x
  y=x**2
  xx=x-.785398164
  bessy0=4.7447.26470d0, 1.030246d0, 1.0d0/}
if(x.abs().lt.8.)then
  Direct rational approximation.
  y=x**2
  bessy0=(r1+y*(r2+y*(r3+y*(r4+y*(r5+y*r6)))))/(s1+y*(s2+y*(s3+y*(s4+y*(s5+y*s6))))+0.636619772*bessj0(x)*log(x))
else
  Fitting function (6.5.10).
  z=8./x
  y=x**2
  xx=x-.785398164
  bessy0=4.7447.26470d0, 1.030246d0, 1.0d0/}
endif
return
END

FUNCTION bessj1(x)
REAL bessj1, x
C USES bessj0

Returns the Bessel function $J_1(x)$ for any real $x$.

DOUBLE PRECISION p1, p2, p3, p4, p5, q1, q2, q3, q4, q5, r1, r2, r3, r4,
  r5, r6, s1, s2, s3, s4, s5, s6
SAVE p1, p2, p3, p4, p5, q1, q2, q3, q4, q5, r1, r2, r3, r4, r5, r6,
  s1, s2, s3, s4, s5, s6
DATA p1, p2, p3, p4, p5/1.d0, 0.183105d-2, -0.3516396496d-4, 0.2457520174d-5,
  -0.240337019d-6/, q1, q2, q3, q4, q5/0.04687499995d0, -0.2002690873d-3,
  0.844919909d-5, -0.82228987d-6, 0.105787412d-6/, r1, r2, r3, r4, r5, r6/72362614232.0d0, -7895059235.0d0, 242396853.1d0,
  -2972611.439d0, 15704.48260d0, -30.16036606d0/, s1, s2, s3, s4, s5, s6/144725228442.0d0, 2300535178.0d0, 99447.43394d0,
  -2972611.439d0, 15704.48260d0, -30.16036606d0/,
DATA p1, p2, p3, p4, p5/1.d0, 0.183105d-2, -0.3516396496d-4, 0.2457520174d-5,
  -0.240337019d-6/, q1, q2, q3, q4, q5/0.04687499995d0, -0.2002690873d-3,
  0.844919909d-5, -0.82228987d-6, 0.105787412d-6/, r1, r2, r3, r4, r5, r6/72362614232.0d0, -7895059235.0d0, 242396853.1d0,
  -2972611.439d0, 15704.48260d0, -30.16036606d0/, s1, s2, s3, s4, s5, s6/144725228442.0d0, 2300535178.0d0, 99447.43394d0,
  -2972611.439d0, 15704.48260d0, -30.16036606d0/,
if(abs(x).lt.8.)then
  Direct rational approximation.
  y=x**2
  bessj1=x*(r1+y*(r2+y*(r3+y*(r4+y*(r5+y*r6)))))/(s1+y*(s2+y*(s3+y*(s4+y*(s5+y*s6)))))
else
  Fitting function (6.5.9).
  ax=abs(x)
  z=8./ax
  y=x**2
  xx=ax-2.356194491
6.5 Bessel Functions of Integer Order

bessj1 = sqrt(.636619772/ax) * (cos(xx) * *(p1+y*(p2+y*(p3+y*(p4+y*(p5)))))) - z * sin(xx) * *(q1+y*(q2+y*(q3+y*(q4+y*(q5))))) * sign(1.,x)
endif
return
END

FUNCTION bessy1(x)
REAL bessy1, x
C USES bessj1
DOUBLE PRECISION p1, p2, p3, p4, p5, q1, q2, q3, q4, q5, r1, r2, r3, r4,
r5, r6, a1, s1, s2, s3, s4, s5, s6, s7, y
We'll accumulate polynomials in double precision.
SAVE p1, p2, p3, p4, p5, q1, q2, q3, q4, q5, r1, r2, r3, r4,
r5, r6, a1, s1, s2, s3, s4, s5, s6, s7
DATA p1, p2, p3, p4, p5 / 1.d0, .183105d-2, -.3516396496d-4, .2457520174d-5,
*.240337019d-6/, q1, q2, q3, q4, q5 / .04687499995d0, -.2002690873d-3,
*.8449199096d-5/, s1, s2, s3, s4, s5, s6, s7 / .2499580570d14, .42441966412d12,
*.37365367410d10, .224690002d8, .3849632885d4, 1.d0/
if(x.lt.8.)then
  Rational function approximation of (6.5.8).
y = x**2
  bessy1 = x*(r1+y*(r2+y*(r3+y*(r4+y*(r5+y*r6)))))/(s1+y*(s2+y*(s3+y*(s4+y*(s5+y*s7)))))+.636619772
else
  Fitting function (6.5.10).
z = 8./x
  y = x**2
  xx = 2.356194491
  bessy1 = sqrt(.636619772/x) * (sin(xx) * *(p1+y*(p2+y*(p3+y*(p4+y*(p5)))))) + z * cos(xx) * *(q1+y*(q2+y*(q3+y*(q4+y*(q5)))))
endif
return
END

We now turn to the second task, namely how to use the recurrence formulas (6.5.6) and (6.5.7) to get the Bessel functions \( J_n(x) \) and \( Y_n(x) \) for \( n \geq 2 \). The latter of these is straightforward, since its upward recurrence is always stable:

FUNCTION bessy(n,x)
INTEGER n
REAL bessy, x
C USES bessy0, bessy1
Returns the Bessel function \( Y_n(x) \) for positive \( x \) and \( n \geq 2 \).
INTEGER j
REAL by, bym, bym0, tox, bessy0, bessy1
if(n.lt.2)pause 'bad argument n in bessy'
tox = 2./x
by = bessy1(x) Starting values for the recurrence.
bym = bessy0(x)
do j=1,n-1
  byp = j*tox*by - bym
  bym = by
  by = byp
endo
bessy = by
return
END
The cost of this algorithm is the call to \texttt{bessy1} and \texttt{bessy0} (which generate a call to each of \texttt{bessj1} and \texttt{bessj0}), plus $O(n)$ operations in the recurrence.

As for $J_n(x)$, things are a bit more complicated. We can start the recurrence upward on $n$ from $J_0$ and $J_1$, but it will remain stable only while $n$ does not exceed $x$. This is, however, just fine for calls with large $x$ and small $n$, a case which occurs frequently in practice.

The harder case to provide for is that with $x < n$. The best thing to do here is to use Miller's algorithm (see discussion preceding equation 5.5.16), applying the recurrence downward from some arbitrary starting value and making use of the upward-unstable nature of the recurrence to put us on the correct solution. When we finally arrive at $J_0$ or $J_1$ we are able to normalize the solution with the sum (5.5.16) accumulated along the way.

The only subtlety is in deciding at how large an $n$ we need start the downward recurrence so as to obtain a desired accuracy by the time we reach the $n$ that we really want. If you play with the asymptotic forms (6.5.3) and (6.5.5), you should be able to convince yourself that the answer is to start larger than the desired $n$ by an additive amount of order \( [\text{constant} \times n]^{1/2} \), where the square root of the constant is, very roughly, the number of significant figures of accuracy.

The above considerations lead to the following function.

```fortran
FUNCTION bessj(n,x)
INTEGER n,IACC
REAL bessj,x,BIGNO,BIGNI
PARAMETER (IACC=40,BIGNO=1.e10,BIGNI=1.e-10)
C USES bessj0,bessj1

Returns the Bessel function $J_n(x)$ for any real $x$ and $n \geq 2$.

INTEGER j,jsum,m
REAL ax,bj,bjm,bjp,sum, tox,bessj0,bessj1
if(n.lt.2)pause 'bad argument n in bessj'
ax=abs(x)
if(ax.eq.0.)then
  bessj=0.
else if(ax.gt.float(n))then
  Upwards recurrence from $J_0$ and $J_1$.
  tox=2./ax
  bjm=bessj0(ax)
  bj=bessj1(ax)
  do 11 j=1,n-1
    bjpm=j*tox*bj-bjm
    bj=bjpm
  enddo 11
  bessj=bj
else
  Downwards recurrence from an even $m$ here computed. Make IACC larger to increase accuracy.
  m=2*((n+int(sqrt(float(IACC*n))))/2)
  bessj=0.
  jsum=0
  sum=0.
  bjp=0.
  bj=1.
  do 12 j=m,1,-1
    bjpm=j*tox*bj-bjp
    bj=bjpm
  enddo 12
  if(abs(bj).gt.BIGNO)then
    Renormalize to prevent overflows.
    bj=bj*BIGNI
  else
    js will alternate between 0 and 1; when it is 1, we accumulate in sum the even terms in (5.5.16).
    sum=0.
    bjp=0.
    bj=1.
    do 12 j=m,1,-1
      bjpm=j*tox*bj-bjp
      bj=bjpm
    enddo 12
    The downward recurrence.
    bessj=bj
end if
end if
end FUNCTION bessj
```
6.6 Modified Bessel Functions of Integer Order

The modified Bessel functions $I_n(x)$ and $K_n(x)$ are equivalent to the usual Bessel functions $J_n$ and $Y_n$ evaluated for purely imaginary arguments. In detail, the relationship is

\[
I_n(x) = (-i)^n J_n(ix) \\
K_n(x) = \frac{\pi}{2} i^{n+1} [J_n(ix) + iY_n(ix)]
\]  

(6.6.1)

The particular choice of prefactor and of the linear combination of $J_n$ and $Y_n$ to form $K_n$ are simply choices that make the functions real-valued for real arguments $x$.

For small arguments $x \ll n$, both $I_n(x)$ and $K_n(x)$ become, asymptotically, simple powers of their argument

\[
I_n(x) \approx \frac{1}{n!} \left(\frac{x}{2}\right)^n \quad n \geq 0 \\
K_0(x) \approx -\ln(x) \\
K_n(x) \approx \frac{(n-1)!}{2} \left(\frac{x}{2}\right)^{-n} \quad n > 0
\]  

(6.6.2)

These expressions are virtually identical to those for $J_n(x)$ and $Y_n(x)$ in this region, except for the factor of $-2/\pi$ difference between $Y_n(x)$ and $K_n(x)$. In the region