Efficient Algorithms for Computing the Parameter Derivatives of k-hypergeometric Functions and Their Extensions to Other Special Functions

Huizeng Qin, Nina Shang, Youmin Lu

***Abstract*—The k-hypergeometric functions are defined as**

$$ {}_pF_q(a,k,b,s;z)=\sum_{n=0}^{\infty}\frac{(a_1)_{n,k_1}(a_2)_{n,k_2}\cdots(a_p)_{n,k_p}z^n}{(b_1)_{n,s_1}(b_2)_{n,s_2}\cdots(b_q)_{n,s_q}n!}, $$

where $(x)_{n,k} = x(x+k)(x+2k)\cdots(x+(n-1)k)$ is the Pochhammer k-symbol. In this paper, efficient recursive algorithms for computing the parameter derivatives of the k-hypergeometric functions are developed. As the generalized hypergeometric functions are special cases of this function and many special functions can be expressed in terms of the generalized hypergeometric functions, the algorithms can also be extended to computing the parameter derivatives of the hypergeometric functions and many other special functions. The Bessel functions and modified Bessel functions are presented as examples of such an application. Theoretical analysis is worked out, some computation using Mathematica is performed, and data is provided to show the advantages of our algorithms.

***Index Terms*—Pochhammer k-symbol; k-hypergeometric function; Hypergeometric function; Parameter derivatives .**

I. Introduction

THE generalized hypergeometric functions ${}_pF_q$ have been studied extensively from the mathematical point of view[24,25]. They occur in a wide variety of problems in theoretical physics, applied mathematics, statistics, and engineering sciences. For example, the confluent hypergeometric Kummer function ${}_1F_1$ is closely related to the two-body Coulomb problem [26,27]; the Gauss hypergeometric function ${}_2F_1$ is the solution of Schrödinger equation when solving the Pöschl–Teller,Wood–Saxon or Hulthén potentials[28]. The derivatives of the hypergeometric functions with respect to all parameters have also been the subject of substantial research in recent years[15-21,23-25,27] as they are widely employed in mathematics, physics, and other related science and engineering fields. It should be noted that methods for finding the parameter/order derivatives of the hypergeometric functions can also be used to find the parameter/order derivatives of many other related special functions, and many authors[1-14] have strived to find the parameter/order derivatives of various special functions in recent years. Unfortunately, the existing methods based on analytical properties are not suitable for efficient computation.

This paper is mainly concerned about first developing efficient algorithms for computing the parameter/order derivatives of the k-hypergeometric functions, and then extending these methods to compute the parameter/order derivatives of the general hypergeometric function and other related special functions. As many Mathematica users know, the parameter/order derivatives of the hypergeometric functions can be calculated using both native Mathematica functions or by directly using their series or integral expressions. However, these methods cannot achieve the required precision or symbolic expression when the derivatives are of high order due to the limits of their computation efficiency. We can prove that the algorithms discussed in this paper are much more efficient analytically and also provide empirical evidence to show that our algorithms are efficient enough to qualitatively overcome existing computational hurdles. It is well-known that many special functions can be expressed in terms of the hypergeometric functions. Therefore, the parameter/order derivatives of these special functions can be reduced to the parameter derivatives of the hypergeometric functions. As a result, high-precision fast calculation of parameter/order derivatives of many other special functions can also be realized by using the algorithms in this paper. For example, high-precision fast calculation of the parameter/order derivatives of the Bessel, Struve, and Legendre functions can be realized with the application of our algorithms as they are directly related to the hypergeometric functions. We emphasize again that the order derivatives of the Bessel functions are used in the study of the monotonicity, which in turn has applications in quantum physics [29]. We use a large portion of section IV to show how our algorithms can be extended to efficiently compute the parameter/order derivatives of the Bessel function and the modified Bessel functions. Table 4 shows that the extension of our algorithms is able to achieve the required precision when algorithms using series and integral expressions directly are not feasible.

Manuscript received Dec. 24, 2020; revised May 08, 2021. This work was supported in part by the National Natural Science Foundation of China (61771010, 11771255, 61702307) and Young Innovation Teams of Shandong Province (2019KJI013).

Huizeng Qin is a Professor of the School of Mathematics and Statistics, Shandong University of Technology, Zibo, Shandong, 255000, P. R. China (e-mail: qinhz_000@163.com).

Nina Shang is a Professor of the School of Mathematics and Statistics, Shandong University of Technology, Zibo, Shandong, 255000 P. R. China (e-mail: shangnina@163.com).

Youmin Lu is a Professor of the Department of Mathematical and Digital Sciences, Bloomsburg University, Bloomsburg, PA 17815 USA (corresponding author; phone: 570-389-4625; fax: 570-389-3199; e-mail:ylu@bloomu.edu).

The classical hypergeometric function ${}_pF_q(a;b;z)$ is defined as

$$ {}_pF_q(a;b;z) = \sum_{n=0}^{\infty} \frac{(a_1)_n (a_2)_n \cdots (a_p)_n z^n}{(b_1)_n (b_2)_n \cdots (b_q)_n n!}, \tag{1} $$

where $a = (a_1, a_2, \cdots, a_p)$, $b = (b_1, b_2, \cdots, b_q)$ and $(x)_n$ is the standard Pochhammer symbol. In [19], Ancarania et. al study the parameter derivatives of the hypergeometric functions and give the following results:

$$ \frac{\partial^n}{\partial a_1^n}\left[{}_pF_q\left(a;b;z\right)\right] = \frac{A_n z^n}{(a_1)_n}\, {}_p\Theta_q^{(n)} \left(\begin{matrix} 1,\ldots,1|a_1,a_1+1,\ldots,a_1+\mathrm{n},a_2+\mathrm{n},\ldots,a_p+\mathrm{n} \\ a_1+1,\ldots,a_1+\mathrm{n}|\mathrm{n}+1,b_1+\mathrm{n},\ldots,b_q+\mathrm{n} \end{matrix} |;\mathrm{z},\ldots,\mathrm{z}\right) \tag{2} $$

and

$$ \frac{\partial^n}{\partial b_1^n}\left[{}_pF_q\left(a;b;z\right)\right] = \frac{n!(-1)^n A_1 z}{b_1^n}\, {}_p\Theta_q^{(n)} \left(\begin{matrix} 1,\ldots,1|b_1,b_1,\ldots,b_1,a_1+1,\ldots,a_p+1 \\ b_1+1,\ldots,b_1+1|2,b_1+1,\ldots,b_q+1 \end{matrix} |\,;\mathrm{z},\ldots,\mathrm{z}\right), \tag{3} $$

where $A_n = \frac{(a_1)_n (a_2)_n \cdots (a_p)_n}{(b_1)_n (b_2)_n \cdots (b_q)_n}$ and

$$ \begin{aligned} &{}_p\Theta_q^{(n)} \left(\begin{matrix} \alpha_1,\ldots,\alpha_{n+1}|\beta_1,\ldots,\beta_{n+p} \\ \gamma_1,\ldots,\gamma_n|\delta_1,\delta_2,\ldots,\delta_{q+1} \end{matrix} |;x_1,x_2,\ldots,x_{n+1}\right) \\ &= \sum_{m_1=0}^{\infty} \cdots \sum_{m_{n+1}=0}^{\infty} (\alpha_1)_{m_1} (\alpha_2)_{m_2} \cdots (\alpha_{n+1})_{m_{n+1}} \\ &\times \frac{(\beta_1)_{m_1} (\beta_2)_{m_1+m_2} \cdots (\beta_n)_{m_1+m_2+\ldots+m_n}}{(\gamma_1)_{m_1} (\gamma_2)_{m_1+m_2} \cdots (\gamma_n)_{m_1+m_2+\ldots+m_n}} \\ &\times \frac{(\beta_{n+1})_{m_1+m_2+\ldots+m_{n+1}} \cdots (\beta_{n+p})_{m_1+m_2+\ldots+m_{n+1}}}{(\delta_1)_{m_1+m_2+\ldots+m_{n+1}} \cdots (\delta_{q+1})_{m_1+m_2+\ldots+m_{n+1}}} \\ &\times \frac{x_1^{m_1} x_2^{m_2} \ldots x_{n+1}^{m_{n+1}}}{m_1! m_2! \ldots m_{n+1}!}. \end{aligned} \tag{4} $$

It is clear that (4) is a hypergeometric function with $n+1$ variables, and one can directly use formulas (2) and (3) to compute $\frac{\partial^n}{\partial a_1^n}\left[{}_pF_q\left(a;b;z\right)\right]$ and $\frac{\partial^n}{\partial b_1^n}\left[{}_pF_q\left(a;b;z\right)\right]$ respectively, but its time complexity is $O(N^{n+1})$ with N as the number of terms used, which clearly needs to be improved. The hypergeometric functions are solutions of hypergeometric equations. Therefore, the parameter derivatives also play an important role in solving non-homogeneous differential equations corresponding to special functions. For example, the confluent hypergeometric function $F = {}_1F_1(a,b,z)$ satisfies the differential equation

$$ \left[z\frac{d^2}{dz^2} + (b-z)\frac{d}{dz} - a\right] F = 0. \tag{5} $$

and its derivatives satisfy the following non-homogeneous differential equation [16]

$$ \left[z\frac{d^2}{dz^2} + (b-z)\frac{d}{dz} - a\right] G^{(n)} = nG^{(n-1)}, \tag{6} $$

where $G^{(n)} = \frac{\partial^n}{\partial a^n}\left[{}_1F_1(a,b,z)\right]$, and

$$ \begin{aligned} G^{(2)} = &\sum_{m_1=0}^{\infty} \sum_{m_2=0}^{\infty} \sum_{m_3=0}^{\infty} \frac{(1)_{m_1} (1)_{m_2} (a)_{m_1+m_2}}{(3)_{m_1+m_2} (a+1)_{m_1} (b)_{m_1+m_2+2}} \\ &\times \frac{(1)_{m_3} (a+m_1+m_2+2)_{m_3} z^{m_1+m_2+m_3+2}}{(m_1+m_2+3)_{m_3} (b+m_1+m_2+2)_{m_3} m_1! m_2! m_3!}. \end{aligned} \tag{7} $$

As a sequence, expression (7) can certainly be used to compute its value directly, but the complexity is $O(N^3)$, which is very inefficient again.

As an extension of the classical hypergeometric functions, the $k-$hypergeometric function [21] ${}_pF_q(a,k;b,s;z)$ is defined as the formal power series

$$ {}_pF_q(a,k;b,s;z) = \sum_{n=0}^{\infty} \frac{(a_1)_{n,k_1} (a_2)_{n,k_2} \cdots (a_p)_{n,k_p} z^n}{(b_1)_{n,s_1} (b_2)_{n,s_2} \cdots (b_q)_{n,s_q} n!}, \tag{8} $$

where $a = (a_1, a_2, \cdots, a_p), k = (k_1, k_2, \cdots, k_p), b = (b_1, b_2, \cdots, b_q)$, $s = (s_1, s_2, \cdots, s_q)$ and $(x)_{n,k}$ is the Pochhammer k-symbol [22] defined as

$$ (x)_{n,k} = x(x+k)(x+2k)\cdots(x+(n-1)k). \tag{9} $$

We can express the Pochhammer symbol $(x)_n$ in terms of the Gamma function

$$ (x)_n = \frac{\Gamma(x+n)}{\Gamma(x)} \tag{10} $$

or the Stirling number of the first kind

$$ (x)_n = \sum_{s=0}^{n} s(n,s)(-1)^{n-s} x^s. \tag{11} $$

Similarly, we can use the expression of the Pochhammer k-symbol $(x)_{n,k}$ [21]:

$$ (x)_{n,k} = \frac{\Gamma_k(x+nk)}{\Gamma_k(x)}, \tag{12} $$

where $\Gamma_k(x)$ is the $k-$Gamma function defined as

$$ \Gamma_k(x) = \lim_{n\to\infty} \frac{n! k^n (nk)^{\frac{x}{k}-1}}{(x)_{n,k}}, x \in C \backslash kZ^- \tag{13} $$

or

$$ \Gamma_k(x) = \int_0^{\infty} t^{x-1} e^{-\frac{t^k}{k}} dt = k^{\frac{x}{k}-1} \Gamma(\frac{x}{k}), Re(x) > 0. \tag{14} $$

The classical hypergeometric functions are clearly special cases of the k-hypergeometric functions as $k=1$ and s=1, therefore any algorithm for computing the parameter derivatives with respect to a or b of ${}_pF_q(a,k;b,s;x)$ can be applied to computing the parameter derivatives of ${}_pF_q(a;b;z)$. The k-hypergeometric function is a solution of the equation

$$ \left(xP_p(D,a,k) - Q_q(D,b,s)\right) y = 0, \tag{15} $$

where $P_p(\lambda,a,k) = (k_1\lambda + a_1)\cdots(k_p\lambda + a_p), Q_q(\lambda,b,s) = \lambda(s_1\lambda + b_1 - s_1)\cdots(s_q\lambda + b_q - s_q)$ and $D = x\frac{\partial}{\partial x}$. Letting $p=2$ and $q=1$ in (15), we have

$$ \begin{aligned} &(k_1k_2x - s_1)\, xy'' \\ &+ \left((k_1k_2 + a_2k_1 + a_1k_2)\, x - b_1\right) y' + a_1a_2y = 0. \end{aligned} \tag{16} $$

Letting k_1, k_2 and $s_1 = 1$, (16) becomes the standard hypergeometric differential equation. Although the k-hypergeometric functions can be expressed in terms of the classical hypergeometric functions in the following form

$$ \begin{aligned} {}_pF_q(a,k;b,s;x) &= {}_pF_q(\tfrac{a}{k};\tfrac{b}{s};\tfrac{kx}{s}) \\ &= \sum_{n=0}^{\infty} \frac{(\frac{a_1}{k_1})_n (\frac{a_2}{k_2})_n \cdots (\frac{a_p}{k_p})_n}{(\frac{b_1}{s_1})_n (\frac{b_2}{s_2})_n \cdots (\frac{b_q}{sq})_n n!} \left(\frac{kx}{s}\right)^n, \end{aligned} \tag{17} $$

where $\frac{a}{k} = \left(\frac{a_1}{k_1}, \frac{a_2}{k_2}, \cdots, \frac{a_p}{k_p}\right), \frac{b}{s} = \left(\frac{b_1}{s_1}, \frac{b_2}{s_2}, \cdots, \frac{b_q}{sq}\right)$, and $\frac{kx}{s} = \frac{k_1k_2\cdots k_px}{s_1s_2\cdots s_p}$, and one can directly use the right side of this expression to perform computations of the parameter derivative with respect to k or s of the k-hypergeometric functions logically, it is indeed very difficult, especially when the order of differentiation is high, due to the fact that the parameters k and s appear in the denominator of the generalized hypergeometric function. That being said, we pay our attention to finding efficient algorithms for computing the parameter derivatives of the k-hypergeometric functions first. Our algorithms are especially efficient when the order of the derivatives is high.

It should be pointed out that, although Mathematica has native functions that can be used to compute the lower order symbolic parameter derivatives of the generalized hypergeometric functions, they are time-consuming and sometimes cannot achieve the specified accuracy. The following example gives comparison by providing the Mathematica code and time spent for computing $\frac{\partial^3}{\partial a_1^2\partial b_1}\left[{}_2F_1(a,k;b,s;z)\right]$ and $\frac{\partial^4}{\partial a_1^2\partial b_1^2}\left[{}_2F_1(a,k;b,s;z)\right]$, where $z=\frac{2}{3}, a=(\frac{1}{2},\frac{2}{3}), k=(\frac{1}{3},\frac{3}{5}), b=\frac{4}{3}, s=\frac{9}{7}$, by using some Mathematica internal functions and our algorithms in this paper, respectively.

```
Clear[aa];Clear[cc];k={1/3, 3/5} ;a=1/2; b=2/3; c=4/3; s=9/7;
z=2/3 ;Prec=32;ms={2,0};ns={1};Clear[aa];Clear[cc];
a1={a,b};b1={c};s1={s};
Timing[D[D[Hypergeometric2F1[aa/k[[1]], b/k[[2]], cc/s, k[[1]]k[[2]]z/s],
{aa,ms[[1]]}]/.aa→ a,{cc,ns[[1]]}]/.cc→ c,Prec]]
Timing[Hypergeometricab[x,a1,k,b1,s1,2,1,ms,ns]
ns={2};Clear[aa];Clear[cc];
Timing[D[D[Hypergeometric2F1[aa/k[[1]], b/k[[2]], cc/s, k[[1]]k[[2]]z/s],
{aa,ms[[1]]}]/.aa→ a,{cc,ns[[1]]}]/.cc→ c,Prec]]
Timing[Hypergeometricab[x,a1,k,b1,s1,2,1,ms,ns]
{0.156250,-0.17105873358670190610911461781888}
{0.,-0.1710587335867019061091146178188814303285}
{0.703125,0.311619134384}
{0.,0.311619134383516381576850914082862448071}
```

where

$$D[D[Hypergeometric2F1[a,b,c,z]]]$$

is a Mathematica internal function and

$$Hypergeometricab[x,a,k,b,s,p,q,ms,ns]$$

is our function for computing

$$\frac{\partial^{m_1+m_2+\cdots+m_p+n_1+n_2+\cdots+n_q}}{\partial a_1^{m_1}\partial a_2^{m_2}\cdots\partial a_p^{m_p}\partial b_1^{n_1}\partial b_2^{n_2}\cdots\partial b_q^{n_q}}\left[{}_pF_q(a,k;b,s;z)\right]$$

in this paper. This example shows that our function is much more efficient and can achieve better accuracy. When the order of the derivative is getting higher, the Mathematica internal functions do not work well and our algorithm still works very efficiently. Here is another example:

```
Prec=16;Clear[a];Clear[b];Clear[a2];Clear[b2];a={2,2/3};
b={1/2,3};k={1,1};s={1,1};x=1/3;
Timing[N[D[D[HypergeometricPFQ[{2,a2},{1/2,b2},x],
{a2,1}]/.a2→ 2/3,{b2,1}]/.b2→ 3,Prec]]
Hypergeometricab[x,a,k,b,s,2,2,{0,1},{0,1}]
Timing[N[D[D[HypergeometricPFQ[{2,a2},{1/2,b2},x],
{a2,3}]/.a2→ 2/3,{b2,3}]/.b2→ 3,Prec]]
Hypergeometricab[x,a,k,b,s,2,2,{0,3},{0,3}]
{0.015625,-0.2069231935716211}
{0.,-0.2069231935716211044}
{0.062500,HypergeometricPFQ^{{0,3},{0,3},0)} [...]}
{0.,-0.011331765835823066706}
```

where "..." means that the contents of some square brackets are omitted. Using the Mathematica internal function,

$$\frac{\partial^2}{\partial a_2\partial b_2}\left[{}_2F_2(\{2,\tfrac{2}{3}\},\{\tfrac{1}{2},3\},\tfrac{1}{2})\right]$$

is calculated, whereas

$$\frac{\partial^6}{\partial a_2^3\partial b_2^3}\left[{}_2F_2(\{2,\tfrac{2}{3}\},\{\tfrac{1}{2},3\},\tfrac{1}{2})\right]$$

is not. Again, our function is much more efficient and can achieve much better accuracy.

II. Explicit formulas for the parameter derivatives of k-hypergeometric function

Using expression (11), one can get an expression for the derivatives of the Pochhammer symbol

$$\frac{d^n}{dx^n}\left[(x)_m\right]=\begin{cases} n!\sum_{k=n}^{m}\begin{pmatrix}k\\n\end{pmatrix}(-1)^{\mathrm{m-k}}s(m,k)x^{\mathrm{k-n}}, \\ \quad n=1,2,\cdots,m, \\ 0, n=m+1,m+2,\cdots \end{cases} \tag{18}$$

and the following expression for the parameter derivatives of the confluent hypergeometric function

$$\begin{aligned} G^{(n)}&=\frac{\partial^n}{\partial a^n}\left[{}_1F_1\left(a,b,z\right)\right]=\frac{\partial^n}{\partial a^n}\sum_{m=0}^{\infty}\frac{(a)_n z^m}{(b)_m m!} \\ &=n!\sum_{m=n}^{\infty}\sum_{k=n}^{m}C_k^n\frac{(-1)^{m-k}s(m,k)a^{k-n}z^m}{(b)_m m!}. \end{aligned} \tag{19}$$

The time complexity of using (19) to compute $G^{(n)}$ directly would be $O(N^2)$, which is clearly a great improvement of the result in reference [16].

In [21], the authors give the following proposition about the Pochhammer k-symbol.

Proposition 2.1 The following identities hold.

1).

$$(x)_{n,k}=\sum_{l=0}^{n-1}e_l^{n-1}(1,2,\cdots,n-1)k^l x^{n-l}. \tag{20}$$

2).

$$\frac{\partial}{\partial k}(x)_{n,k}=\sum_{l=0}^{n-1}l\,(x)_{l,k}\,(x+(l+1)k)_{n-1-l,k}, \tag{21}$$

where $e_s^n(x_1,x_2,\cdots,x_n)=\sum_{1\le i_1<\cdots<i_s\le n}x_{i_1}\cdots x_{i_s}$.

Following the idea of (18), we can give the following proposition that can be proved by using (12) and the expression $(x)_{n,k}=k^n\left(\frac{x}{k}\right)_n$.

Proposition 2.2 The following identities hold.

1).

$$\begin{aligned}(x)_{n,k}&=\sum_{l=1}^{n}(-1)^{n-l}s(n,l)k^{n-l}x^l \\ &=\sum_{l=0}^{n-1}(-1)^l s(n,n-l)k^l x^{n-l}.\end{aligned} \tag{22}$$

2).

$$A_{k,n,m}(x)=\begin{cases} m!\sum_{l=m}^{n}\begin{pmatrix}l\\m\end{pmatrix}s(n,l)(-k)^{n-l}x^{l-m}, \\ m=1,2,\cdots,n, \\ 0, m>n. \end{cases} \tag{23}$$

3).

$$B_{k,n,m}(x)=\begin{cases} m!\sum_{l=m}^{n-1}\begin{pmatrix}l\\m\end{pmatrix}\frac{(-1)^l s(n,n-l)x^{n-l}}{k^{(m-l)}}, \\ m=1,2,\cdots,n, \\ 0, m>n. \end{cases} \tag{24}$$

Remark Comparing the above propositions, one can find $e_l^{n-1}(1,2,\cdots,n-1)=(-1)^l s(n,n-l)$.

Using (23) and (24), we can get the following theorem.

Theorem 2.1 For $m=1,2,\cdots$,

$$\begin{aligned}&\frac{\partial^m}{\partial a_1^m}\left[{}_pF_q(a,k;b,s;z)\right] \\ &=\sum_{n=m}^{\infty}\frac{A_{k_1,n,m}(a_1)(a_2)_{n,k_2}\cdots(a_p)_{n,k_p}z^n}{(b_1)_{n,s_1}(b_2)_{n,s_2}\cdots(b_q)_{n,s_q}n!}\end{aligned} \tag{25}$$

and

$$\frac{\partial^m}{\partial k_1^m}\left[{}_pF_q(a,k;b,s;z)\right] = \sum_{n=m}^{\infty}\frac{B_{k_1,n,m}(a_1)(a_2)_{n,k_2}\cdots(a_p)_{n,k_p}z^n}{(b_1)_{n,s_1}(b_2)_{n,s_2}\cdots(b_q)_{n,s_q}n!}, \tag{26}$$

Similarly, we give the following proposition for $C_{k,n,m}(x)=\frac{d^m}{dx^m}\left[\frac{1}{(x)_{n,k}}\right]$ and $D_{k,n,m}(x)=\frac{\partial^m}{\partial k^m}\left[\frac{1}{(x)_{n,k}}\right]$.

Proposition 2.3 The following identities hold.

1).

$$C_{k,n,m}(x)=\frac{m!}{(n-1)!k^{n-1}}\sum_{i=0}^{n-1}C_{n-1}^{i}\frac{(-1)^{i+m}}{(x+ik)^{m+1}}. \tag{27}$$

2).

$$D_{k,n,m}(x)=\frac{(-1)^m(n-1)_m}{(n-1)!k^{n+m-1}} + \frac{(-1)^m m!}{(n-1)!k^{n-1}}\sum_{j=0}^{m}\begin{pmatrix} n+j-2 \\ j\end{pmatrix}\frac{1}{k^j} \sum_{i=1}^{n-1}\begin{pmatrix} n-1 \\ i\end{pmatrix}\frac{(-1)^i i^{m-j}}{(x+ik)^{m-j+1}}. \tag{28}$$

Proof Using

$$\frac{1}{(x)_n}=\sum_{i=0}^{n-1}\frac{(-1)^i}{(n-1-i)!i!(x+i)} = \frac{1}{(n-1)!}\sum_{i=0}^{n-1}\begin{pmatrix} n-2 \\ m\end{pmatrix}C_{n-1}^{i}\frac{(-1)^i}{x+i},$$

we have

$$\begin{aligned}\frac{1}{(x)_{n,k}} &= \frac{1}{k^n\left(\frac{x}{k}\right)_n} = \frac{1}{k^n}\sum_{i=0}^{n-1}\frac{(-1)^i}{(n-1-i)!i!\left(\frac{x}{k}+i\right)} \\ &= \frac{1}{(n-1)!k^{n-1}}\sum_{i=0}^{n-1}\begin{pmatrix} n-1 \\ i\end{pmatrix}\frac{(-1)^i}{x+ik} \\ &= \frac{1}{(n-1)!}\sum_{i=1}^{n-1}\begin{pmatrix} n-1 \\ i\end{pmatrix}(-1)^i\sum_{j=1}^{n-2}\frac{(-i)^{n-j-1}}{x^{n-j}k^j} \\ &+\frac{1}{(n-1)!}\sum_{i=1}^{n-1}\begin{pmatrix} n-1 \\ i\end{pmatrix}(-1)^i\frac{(-i)^{n-1}}{x}{}^{n-1}(x+ik).\end{aligned}$$

Therefore,

$$C_{k,n,m}(x)=\frac{m!}{(n-1)!k^{n-1}}\sum_{i=0}^{n-1}\begin{pmatrix} n-1 \\ i\end{pmatrix}\frac{(-1)^{i+m}}{(x+ik)^{m+1}},$$

and

$$D_{k,n,m}(x)=\frac{(-1)^m(n-1)_m}{(n-1)!k^{n+m-1}} +\frac{(-1)^m m!}{(n-1)!k^{n-1}}\sum_{j=0}^{m}\begin{pmatrix} n+j-2 \\ j\end{pmatrix}\sum_{i=1}^{n-1}\begin{pmatrix} n-1 \\ i\end{pmatrix}\frac{(-1)^i i^{m-j}}{k^j(x+ik)^{m-j+1}}.$$

Using (23),(24),(27) and (28), one can have the following theorem about the parameter derivatives of the k-hypergeometric functions.

Theorem 2.2 For $m_i=0,1,2,\cdots,i=1,2,\cdots,p$, and $n_j=0,1,2,\cdots,j=1,2,\cdots,q$,

$$\frac{\partial^{m_1+m_2+\cdots+m_p+n_1+n_2+\cdots+n_q}}{\partial a_1^{m_1}\partial a_2^{m_2}\cdots\partial a_p^{m_p}\partial b_1^{n_1}\partial b_2^{n_2}\cdots\partial b_q^{n_q}}\left[{}_pF_q(a,k;b,s;z)\right] = \sum_{n=M_p}^{\infty}\prod_{i=1}^{p}\prod_{j=1}^{q}\frac{A_{k_i,n,m_i}(a_i)C_{k_j,n,m_j}(b_j)z^n}{n!} \tag{29}$$

where $M_p=\max\{m_1,\cdots,m_p\}$ and

$$\frac{\partial^{m_1+m_2+\cdots+m_p+n_1+n_2+\cdots+n_q}}{\partial k_1^{m_1}\partial k_2^{m_2}\cdots\partial k_p^{m_p}\partial s_1^{n_1}\partial s_2^{n_2}\cdots\partial s_q^{n_q}}\left[{}_pF_q(a,k;b,s;z)\right] = \sum_{n=M_p}^{\infty}\prod_{i=1}^{p}\prod_{j=1}^{q}\frac{B_{k_i,n,m_i}(a_i)D_{k_j,n,m_j}(b_j)z^n}{n!}. \tag{30}$$

where $A_{k_i,n,m}(a_i)$ and $B_{k_i,n,m}(a_i)(i=1,2,\cdots,p)$ are defined by (23) and (24) respectively, and $C_{k_i,n,m}(a_i)$ and $D_{k_i,n,m}(a_i)(i=1,2,\cdots,p)$ are defined by (27) and (28), respectively.

III. Recursive formulas for the parameter derivatives of the Pochhammer k-symbol, its reciprocal and their efficiency analysis

Some explicit formulas of $A_{k,n,m}(x), B_{k,n,m}(x)$, $C_{k,n,m}(x)$ and $D_{k,n,m}(x)$ are given in the previous section. In this section, we establish some recursive algorithms for them and apply the algorithms to the computation of the parameter derivatives of the k-hypergeometric functions. First, using the fact that $(x)_{n,k}=(x)_{n-1,k}(x+(n-1)k)$, we can get the following lemma and corresponding recursive algorithms.

Lemma 3.1 For integers $n,m\geq 0$ and real number $k>0$, the following recursive formulas hold:

$$A_{k,n,m}(x)=(x+nk-k)A_{k,n\text{-}1,m}(x) + mA_{k,n\text{-}1,m\text{-}1}(x) \tag{31}$$

and

$$B_{k,n,m}(x)=(x+nk-k)B_{k,n\text{-}1,m}(x) + (n-1)mB_{k,n\text{-}1,m\text{-}1}(x). \tag{32}$$

Algorithm of $A_{k,n,m}(x)$ and $B_{k,n,m}(x)$

$$\begin{aligned}&A_{k,0,0}(x)=1, A_{k,0,l}(x)=0, l=1,2,\cdots,m,\\ &A_{k,n,0}(x)=(x+nk-k)A_{k,n\text{-}1,0}(x),\\ &A_{k,n,l}(x)=(x+nk-k)A_{k,n\text{-}1,l}(x)\\ &\quad +lA_{k,n\text{-}1,l\text{-}1}(x),\\ &\quad l=1,2,\cdots,m, n=1,2,\cdots.\end{aligned} \tag{33}$$

and

$$\begin{aligned}&B_{k,0,0}(x)=1, B_{k,0,l}(x)=0, l=1,2,\cdots,m,\\ &B_{k,n,0}(x)=(x+nk-k)B_{k,n\text{-}1,0}(x),\\ &B_{k,n,l}(x)=(x+nk-k)B_{k,n\text{-}1,l}(x)\\ &\quad +(n\text{-}1)lB_{k,n\text{-}1,l\text{-}1}(x),\\ &\quad l=1,2,\cdots,m, n=1,2,\cdots.\end{aligned} \tag{34}$$

Similarly, using the fact that $\frac{1}{(x)_{n-1,k}}=\frac{x+(n-1)k}{(x)_{n,k}}$, one can also get the following lemma and the recursive algorithms for the reciprocal of the Pochhammer k-symbol.

Lemma 3.2 For integers $n,m\geq 0$ and real number $k>0$, the following recursive formulas are true.

$$C_{k,n,m}(x)=\frac{C_{k,n-1,m}(x)-mC_{k,n,m-1}(x)}{x+(n-1)k} \tag{35}$$

and

$$D_{k,n,m}(x)=\frac{D_{k,n-1,m}(x)-(n-1)mD_{k,n,m-1}(x)}{x+(n-1)k}. \tag{36}$$

Algorithm of $C_{k,n,m}(x)$ and $D_{k,n,m}(x)$

$$\begin{aligned}&C_{k,0,0}(x)=1, C_{k,0,l}(x)=0\,(l=1,2,\cdots,m),\\ &C_{k,n,0}(x)=\frac{C_{k,n-1,0}(x)}{x+(n-1)k},\\ &C_{k,n,l}(x)=\frac{C_{k,n-1,l}(x)-lC_{k,n,l-1}(x)}{x+(n-1)k},\\ &l=1,2,\cdots,m, n=1,2,\cdots.\end{aligned} \tag{37}$$

and

$$\begin{aligned}&D_{k,0,0}(x)=1, D_{k,0,l}(x)=0\,(l=1,2,\cdots,m),\\ &D_{k,n,0}(x)=\frac{D_{k,n-1,0}(x)}{x+(n-1)k},\\ &D_{k,n,l}(x)=\frac{D_{k,n-1,l}(x)-(n-1)lD_{k,n,l-1}(x)}{x+(n-1)k},\\ &l=1,2,\cdots,m, n=1,2,\cdots.\end{aligned} \tag{38}$$

Since $A_{k,n,m}(x), B_{k,n,m}(x), C_{k,n,m}(x)$ and $D_{k,n,m}(x)$ in the preceding section are in explicit form, we don't need to compute $A_{k,l,j}(x), B_{k,l,j}(x), C_{k,l,j}(x)$ and $D_{k,l,j}(x)$ for $l=0,1,\cdots,n-1, j=0,1,\cdots,m-1$

intuitively if we use them to perform computation of (29) and (30). If we perform the computation of

$$\frac{\partial^{m_1+m_2+\cdots+m_p}}{\partial a_1^{m_1}\partial a_2^{m_2}\cdots\partial a_p^{m_p}}\left[{}_pF_q(a,k;b,s;z)\right]$$

and

$$\frac{\partial^{m_1+m_2+\cdots+m_p}}{\partial k_1^{m_1}\partial k_2^{m_2}\cdots\partial k_p^{m_p}}\left[{}_pF_q(a,k;b,s;z)\right]$$

by using (33), (34), (37) and (38), we must compute all $A_{k,l,j}(x), B_{k,l,j}(x), C_{k,l,j}(x)$ and $D_{k,l,j}(x)$ for $l = 0,1,\cdots,n-1, j = 0,1,\cdots,m-1$. However, the recursive ones are more efficient because we need to take the sum of the first N terms of the series (depending on the accuracy of the calculation and its relevant properties) when the explicit formulas are used. The total amount of time spent on computing $A_{k_i,n,m_i}(a_i)(B_{k_i,n,m_i}(a_i))(i = 1,\cdots,p, n = \widetilde{m},\widetilde{m}+1,\cdots,N,\ \widetilde{m} = \min\{m_1,\cdots,m_p\})$ is $O\left(N\sum_{i=1}^{p} m_i\right)$ if recursive algorithm (33)((34)) is used, and the time spent on computing $C_{s_j,n,n_i}(a_i)(D_{s_j,n,n_j}(a_i))(i = 1,2,\cdots,p, n = 0,1,2,\cdots,N)$ is $O\left(N\sum_{i=1}^{q} n_i\right)$ if recursive algorithm (37)((38)) is used. However, the time spent on computing $A_{k_i,n,m_i}(a_i)(B_{k_i,n,m}(a_i))(i = 1,2,\cdots,p, n = \widetilde{m},\widetilde{m}+1,\cdots,N)$ is $O\left(N^2\sum_{i=1}^{p} m_i\right)$ if formula (23)((24)) is used, and the time spent on $C_{k_i,n,m_i}(a_i)$ $(D_{k_i,n,m}(a_i))(i = 1,2,\cdots,p, n = \widetilde{m},\widetilde{m}+1,\cdots,N)$ is at least the order of $O\left(N^2\sum_{i=1}^{q} n_i\right)$ if formula (28) ((29)) is used. Therefore, the formulas in section II are much slower than the recursive algorithms in this section. We performed experiment by using Mathematica and the actual time spent by the two different sets of algorithms is recorded in the following table.

Table I. Comparison of efficiency between the explicit formula and the recursive algorithm

ALGO	$x,k,m,N,$	Time	$x,k,m,N,$	Time
(23)	$\frac{1}{3},\frac{2}{3},5,100,$	0.156250	$\frac{1}{3},\frac{2}{3},10,400,$	53.57812
(33)		0.000000		0.046875
(24)	$\frac{2}{3},\frac{5}{3},6,100$	0.156250	$\frac{1}{3},\frac{2}{3},6,400,$	53.32812
(34)		,0.015625		0.046875
(27)	$\frac{2}{3},\frac{2}{3},5,100,$	0.062500	$\frac{1}{3},\frac{2}{3},5,400,$	72.17187
(35)		0.015625		0.187500
(28)	$\frac{1}{3},\frac{4}{3},5,100,$	0.328125	$\frac{1}{3},\frac{2}{3},5,400,$	450.7850
(36)		0.031250		0.265625

The data in Table I shows that the advantage of the recursive algorithm over the explicit formulas becomes more significant as N increases. As N increases from 100 to 400, the recursive algorithm is from 2-10 to 310-1500 times as fast as the explicit formula.

IV. Algorithms and applications of the parameter derivatives of the hypergeometric functions to Bessel functions

In this section we develop our algorithms for computing the parameter derivatives of the hypergeometric functions and apply them to computing the parameter derivatives of the Bessel and modified Bessel functions. Using the recursive algorithm in section 3, we can also get recursive algorithms for (12) and (13). Similar to the usual approach [21], a standard algorithm for computing the k-hypergeometric functions is as follows:

$$\begin{array}{l} S_0 = 1, C_0 = 1, a = 1; b = 1, \\ \text{If } p > 0\ a = \prod_{j=1}^{p}(a_i + j * k_i) \\ \text{If } q > 0\ b = \prod_{l=1}^{q}(b_l + j * s_l) \\ C_{j+1} = \frac{azC_j}{b(j+1)}, S_{j+1} = S_j + C_{j+1}, j = 0,1,2,\cdots, \end{array} \tag{39}$$

where C_j represents the $j+1$st term of the power series (8), and S_j represents the sum of the first $j+1$ terms.

It follows (33) that only a two-dimensional array $A_{i,l}(l = 0,1,\cdots,m_i, i = 1,2,\cdots,p)$ is needed for storing and completing the calculation of the l-derivative of $(a_i)_{n,k}$ with respect to variable a_i. Similarly, two two-dimensional arrays $C1_{j,l}, C2_{j,l}(l = 0,1,\cdots,n_i, j = 1,2,\cdots,q)$ are needed for completing the calculation the l-derivative of $\frac{1}{(b_j)_{n,k}}$ with respect to variable b_j. Now we can improve Algorithms (33) and (37) and can give the following algorithm for computing the expression (29):

$$\begin{array}{l} \text{Initializing} \\ A_{i,0}= 1, i = 1,2,\cdots,p, n = 0, \\ C1_{j,0}= 1, j = 1,2,\cdots,q, C2 = C1, C_0 = 1, a = 1, b = 1, \\ ma = \max\{m_1,m_2,\cdots,m_p\}, nb = \max\{n_1,n_2,\cdots,n_q\}, \\ \text{If } ma + nb > 0\ S_0= 0 \text{ else } S_0= 1 \\ \text{loop body} \\ \text{If } p > 0 \\ \left\{\begin{array}{l} \quad A_{i,l}= (a_i+(n-1)*k_i)A_{i,l}+l*A_{i,l-1} \\ \quad (l = m_i, m_i-1,\cdots,1) \\ \quad A_{i,1}= (a_i+(n-1)k_i)A_{i,1}, \\ i = 1,2,\cdots,p, \\ a = \prod_{i=1}^{p} A_{i,m_i}, \end{array}\right. \\ \text{If } q > 0 \\ \left\{\begin{array}{l} \quad C2_{j,0}=\frac{C1_{j,0}}{b_j+(n-1)s_j} \\ \quad C2_{j,l}=\frac{C_{j,l}-l*C2_{j,l-1}}{b_j+(n-1)s_j}, (l = 1,2,\cdots,n_j) \\ j = 1,2,\cdots,q, \\ C2_{j,l}\longrightarrow C1_{j,l}(l = 1,2,\cdots,n_j, j = 1,2,\cdots,q), \\ b = \prod_{j=1}^{q} C_{j,n_j}, \end{array}\right. \\ C_n=\frac{C_{n-1}*x}{n}; S_n= S_{n-1}+abC_n, n = 1,2,\cdots. \end{array} \tag{40}$$

To improve Algorithms (34) and (38), all we have to do is to replace

$$\left\{\begin{array}{l} A_{i,l}= (a_i+(n-1)*k_i)A_{i,l}+l*A_{i,l-1}, \\ C2_{j,l}=\frac{C_{j,l}-l*C2_{j,l-1}}{b_j+(n-1)s_j} \end{array}\right. \tag{41}$$

by

$$\left\{\begin{array}{l} A_{i,l}= (a_i+(n-1)*k_i)A_{i,l}+(n-1)l*A_{i,l-1}, \\ C2_{j,l}=\frac{C_{j,l}-(n-1)l*C2_{j,l-1}}{b_j+(n-1)s_j}. \end{array}\right. \tag{42}$$

The computational time of (40) is clearly $\sum_{i=1}^{p} m_i + \sum_{i=1}^{q} n_i$ times of the computational time of (39). The efficiency of algorithms (39) and (40) is verified by the following numerical results using Mathematica. By using (17), we can compare them using the following internal functions in Mathematica:

$${}_pF_q(a,k;b,s;x) = \text{HypergeometricPFQ}[\tfrac{a}{k}, \tfrac{b}{s}; \tfrac{K_p z}{S_p}], \tag{43}$$

where $K_p = k_1k_2\cdots k_p, S_p = s_1s_2\cdots s_p$. The numerical results are as follows:

Table II. The time of (39),(40) and (29)

ALGO	$\{m_1,m_2,m_3\}$, $\{n_1,n_2\}$,x	$Time_{32}$	$Time_{64}$	$Time_{128}$
(43)	$\{0,0,0\}$,	0.01562	0.01562	0.01562
(39)	$\{0,0\}, 2/3$	0.00000	0.01562	0.01562
(40)	$\{1,2,2\}$,	0.01562	0.01562	0.03125
(29)	$\{2,3\}, 2/3$	0.04687	0.12500	0.68750
(40)	$\{4,2,3\}$,	0.01562	0.03125	0.04687
(29)	$\{4,3\}, 2/3$	0.04687	0.14062	0.59375

where $a=\{\frac{1}{2},\frac{3}{5},\frac{2}{7}\}, k=\{\frac{3}{2},\frac{3}{4},\frac{1}{5}\}, b=\{\frac{2}{3},\frac{3}{4}\}, s=\{\frac{1}{3},\frac{1}{4}\}$, and (29) is computed by using explicit formulas (23) and (27). As one can see from Table II, explicit formula (29) is much slower than (40), especially when the number of terms in a series is large.

By (17) and

$$ {}_2F_1(a,b;c;z)=\frac{\Gamma(c)}{\Gamma(b)\Gamma(c-b)}\int_0^1\frac{t^{b-1}(1-t)^{c-b-1}}{(1-zt)^a}dt, \tag{44}$$

we have

$$\begin{aligned} &{}_2F_1(a,k;b,s;z)=\frac{\Gamma(\frac{b}{s})}{\Gamma(\frac{a_2}{k_2})\Gamma(\frac{b}{s}-\frac{a_2}{k_2})}\\ &\times\int_0^1 t^{\frac{a_2}{k_2}-1}(1-t)^{\frac{b}{s}-\frac{a_2}{k_2}-1}(1-\frac{k_1k_2z}{s}t)^{-\frac{a_1}{k_1}}dt \end{aligned}\tag{45}$$

where $a=\{a_1,a_2\}$, $k=\{k_1,k_2\}$, $\frac{b}{s}>\frac{a_2}{k_2}$, $\frac{a_2}{k_2}>0$. $\frac{\partial^{m+n+r}}{\partial a^m\partial b^n\partial c^r}\left[{}_2F_1(\text{a,b;c,z}\right]$, $\frac{\partial^{m+n+r}}{\partial a^m\partial b^n\partial c^r}\left[{}_2F_1(\text{a,b,}k_1,k_2\text{;c,s;z}\right]$ can certainly be calculated by using the derivative law of parametric variables with integrals (44) and (45) directly, but it is very complicated and time-consuming. It is even more difficult to calculate $\frac{\partial^{m+n+r}}{\partial k_1^m\partial k_2^n\partial s^r}\left[{}_2F_1(a,b,k_1,k_2;c,s;z\right]$ by using (45). For example,

$$\begin{aligned} \frac{\partial}{\partial s}{}_2F_1(a,k;b,s;z)=&-\frac{b}{s^2B(\frac{a_2}{k_2},\frac{b}{s})}\int_0^1R_1(t,s)dt\\ &-\frac{a_1k_2z\Gamma(\frac{b}{s})}{s^2\Gamma(\frac{a_2}{k_2})\Gamma(\frac{b}{s}-\frac{a_2}{k_2})}\int_0^1R_2(t,s)dt\\ &-\frac{b\Gamma(\frac{b}{s})\left(\psi\left(\frac{b}{s}\right)-\psi\left(\frac{b}{s}-\frac{a_2}{k_2}\right)\right)}{s^2\Gamma(\frac{a_2}{k_2})\Gamma(\frac{b}{s}-\frac{a_2}{k_2})}\int_0^1R_3(t,s)dt, \end{aligned}\tag{46}$$

where

$$\begin{aligned} R_1(t,s)&=t^{\frac{a_2}{k_2}-1}(\text{1-t})^{\frac{b}{s}-\frac{a_2}{k_2}-1}(1-\frac{k_1k_2z}{s}t)^{-\frac{a_1}{k_1}}\ln(\text{1-t}),\\ R_2(t,s)&=t^{\frac{a_2}{k_2}}(1-t)^{\frac{b}{s}-\frac{a_2}{k_2}-1}(1-\frac{k_1k_2z}{s}t)^{-\frac{a_1}{k_1}-1},\\ R_3(t,s)&=t^{\frac{a_2}{k_2}}(1-t)^{\frac{b}{s}-\frac{a_2}{k_2}-1}(1-\frac{k_1k_2z}{s}t)^{-\frac{a_1}{k_1}}. \end{aligned}$$

However, we can rewrite (45) using our algorithm:

$$\int_0^1R_2(t,s)dt=B(\frac{a_2}{k_2},\frac{b}{s}-\frac{a_2}{k_2}){}_2F_1(a,k;b,s;z) \tag{47}$$

and

$$\begin{aligned} &\int_0^1R_2(t,s)\ln^n(1-t)\ln^m(1\text{-}\frac{k_1k_2z}{s_1}t)dt\\ &=(-k_1)^m\sum_{i=0}^{n}\binom{n}{i}s^iB_{0,i}(\frac{a_2}{k_2},y)|_{y=\frac{b}{s}-\frac{a_2}{k_2}}\\ &\frac{\partial^{m+i}}{\partial a_1^m\partial b^i}\left[{}_2F_1(a,k;b,s;z)\right], \end{aligned}\tag{48}$$

where $B_{0,i}(x,y)=\frac{\partial^i}{\partial y^i}B_{0,n}(x,y)$ and $B(x,y)$ is a Beta function. We can also use the following recursive algorithm [11] for $B_{0,i}(x,y)$:

$$B_{0,q}(x,y)=\sum_{j=0}^{q-1}\binom{\text{q-1}}{j}\psi^{(\text{q-1-j})}(x,y)B_{0,j}(x,y). \tag{49}$$

where $\psi^{(k)}(x,y)=\psi^{(k)}(y)-\psi^{(k)}(x+y)$. Let $(48)_R$ be the numerical result on the right-side of (48) and $(48)_L$ be the integral on the left-side of (48) in Mathematica. Some numerical results are as follows.

Table III. Some numerical results and time spent of (48)

ALGO	m,n	Time_{32}, Err	Time_{64}, Err
$(48)_L$	3, 0	$0.0781, 10^{-27}$	$0.2343, 10^{-21}$
$(48)_R$		$0.0156, 10^{-31}$	$0.0312, 10^{-63}$
Integral value: $-$ 5.2563366618411730489256044093 7$\cdots$			
ALGO	m,n	Time_{32}, Err	Time_{64}, Err
$(48)_L$	1, 2	$0.0781, 10^{-24}$	$0.2656, 10^{-18}$
$(48)_R$		$0.0312, 10^{-31}$	$0.0781, 10^{-63}$
Integral value: $-$ 99.008045662554634933190777651 9$\cdots$			
ALGO	m,n	Time_{32}, Err	Time_{64}, Err
$(48)_L$	2, 2	$0.0781, 10^{-24}$	$0.2500, 10^{-18}$
$(48)_R$		$0.0468, 10^{-31}$	$0.0937, 10^{-63}$
Integral value: 134.30455302556773469630042838747$\cdots$			
ALGO	m,n	Time_{32}, Err	Time_{64}, Err
$(48)_L$	3, 2	$0.0781, 10^{-24}$	$0.2500, 10^{-18}$
$(48)_R$		$0.0468, 10^{-31}$	$0.1093, 10^{-63}$
Integral value: $-$ 182.93522664801800304795997218 2$\cdots$			

where $\{a_1,a_2\}$=$\{\frac{1}{3},\frac{3}{5}\}$,$\{k_1,k_2\}$=$\{\frac{3}{2},\frac{6}{5}\}$,b=$\frac{4}{3}$,s=$\frac{8}{5}$,z=$\frac{2}{3}$;

Table IV. Some numerical results and time spent of (48)

ALGO	m,n	Time_{32}, Err	Time_{64}, Err
$(48)_L$	3, 0	$0.0781, 10^{-11}$	$0.1406, 10^{-14}$
$(48)_R$		$0.0156, 10^{-31}$	$0.0312, 10^{-63}$
Integral value: $-$ 7.1888336144179851500597011593579$\cdots$			
ALGO	m,n	Time_{32}, Err	Time_{64}, Err
$(48)_L$	1, 2	$0.0937, 10^{-11}$	$0.1562, 10^{-11}$
$(48)_R$		$0.0156, 10^{-31}$	$0.0468, 10^{-64}$
Integral value: $-$ 1760.5868975396014633077075809639$\cdots$			
ALGO	m,n	Time_{32}, Err	Time_{64}, Err
$(48)_L$	2, 2	$0.0781, 10^{-9}$	$0.1562, 10^{-11}$
$(48)_R$		$0.0312, 10^{-32}$	$0.0468, 10^{-64}$
Integra valuel: 1453.1890750445119565605536532631 8$\cdots$			
ALGO	m,n	Time_{32}, Err	Time_{64}, Err
$(48)_L$	3, 2	$0.0781, 10^{-9}$	$0.1562, 10^{-11}$
$(48)_R$		$0.0312, 10^{-31}$	$0.0468, 10^{-63}$
Integral value: $-$ 1199.7182932950773803479765506986$\cdots$			

where $\{a_1,a_2\}$=$\{1,\frac{3}{5}\}$,$\{k_1,k_2\}$=$\{\frac{9}{8},\frac{6}{5}\}$,b=1,s=$\frac{8}{5}$,z=$\frac{2}{3}$.

Table III and IV show that $(48)_R$ can always achieve the specified precision, and the efficiency is also much better. However, the numerical integration $(48)_L$ cannot achieve the specified precision, and even the high precision (64-bit) results are worse than those under the specified precision of 32-bit. This is because the accumulation error of improper integral affects the true value in integral calculation.

Now we can apply our algorithms to the computation of the parameter derivatives of Bessel functions because the Bessel functions can be expressed in terms of the generalized hypergeometric series as

$$J_\alpha(x)=\widehat{\Gamma}(\alpha+1)\left(\frac{x}{2}\right)^\alpha\ {}_0F_1(\alpha+1,\frac{-x^2}{4}), \tag{50}$$

where $\widehat{\Gamma}(\alpha+1)=\frac{1}{\Gamma(\alpha+1)}$. In [1] the author discusses the higher order derivatives of $J_\alpha(x)$ with respect to the parameter. For example, the author gives

$$\begin{aligned} &\frac{\partial^m}{\partial\upsilon^m}\left[\left\{\begin{array}{c}J_\upsilon(z)\\ I_\upsilon(z)\end{array}\right\}\right]\\ &=(-1)^m\frac{m!i}{2\pi}\sum_{k=0}^{m}\frac{1}{(m-k)!}\sum_{p=0}^{m-k}(-1)^p\binom{m-k}{p}\\ &\times\left[\begin{array}{c}e^{i\upsilon\pi}\left(\ln\frac{z}{2}+\pi i\right)^p\\ -e^{-i\upsilon\pi}\left(\ln\frac{z}{2}-\pi i\right)^p\end{array}\right]\Gamma^{(\text{m-k-p})}(-\upsilon)\\ &\times\left\{\delta_{k,0}\Gamma(\upsilon+1)\left\{\begin{array}{c}J_\upsilon(z)\\ I_\upsilon(z)\end{array}\right\}\mp\frac{(1-\delta_{k,0})Q_k(z)}{(\frac{2}{z})^{\upsilon+2}(\upsilon+1)^{k+1}}\right\}, \end{aligned}\tag{51}$$

where $I_\upsilon(z)$ is the modified Bessel function, and $Q_k(z)=$

$$F^{0:k+1;0}_{1:k+1;0}\left(\begin{array}{c} \text{-}:\upsilon,\upsilon,\cdots,\cdots,\text{-}; \\ 2:\upsilon+1,\upsilon+1,\cdots,\text{v}+1;\text{-}; \end{array} \pm\frac{z^2}{4},\mp\frac{z^2}{4}\right),$$

$$\begin{array}{l} F^{p:q;r}_{s:t;u}\left(\begin{array}{c} (a_p):(b_q);(c_r); \\ (d_s):(e_t);(f_u); \end{array} w,z\right) \\ \quad = \sum\limits_{j=0}^{\infty}\sum\limits_{k=0}^{\infty}\frac{[a_p]_{j+k}[b_q]_j[c_r]_k}{[d_s]_{j+k}[e_t]_j[f_u]_k}\frac{w^j z^k}{j!k!} \end{array} \tag{52}$$

is the Kampé Fériet function, where $(a_p)=a_1,a_2,\cdots,a_p$ and $[a_p]_k=\prod_{i=1}^{p}(a_i)_k$.

It is very complicated to compute $\frac{\partial^m}{\partial\alpha^m}J_\alpha(x)$ using formula (51). To use our recursive algorithms, we express it in terms of the hypergeometric function:

$$\begin{array}{l} \frac{\partial^m}{\partial\alpha^m}J_\alpha(x) \\ = \sum\limits_{l=0}^{m}\left(\begin{array}{c} m \\ l \end{array}\right)P_{m-l,\alpha}(x)\frac{\partial^l}{\partial\alpha^l}\left[{}_0F_1(\alpha+1,\frac{x^2}{4})\right], \end{array} \tag{53}$$

where

$$P_{l,\alpha}(x)=\sum_{u=0}^{l}\left(\begin{array}{c} l \\ u \end{array}\right)\widehat{\Gamma}^{(u)}(\alpha+1)\left(\frac{x}{2}\right)^{\alpha}\ln^{l-u}\frac{x}{2}. \tag{54}$$

For $\widehat{\Gamma}^{(u)}(\alpha)$ we give the following recurrence formula [14]:

$$\begin{array}{l} H^{\psi}_0(\alpha)=1, H^{\psi}_{n,0}(\alpha)=\widehat{\psi}^{(n-1)}(\alpha), \\ H^{\widehat{\psi}}_{n,l}(\alpha)=\sum\limits_{i=1}^{n-1}\left(\begin{array}{c} n\text{-}1 \\ i \end{array}\right)\widehat{\psi}^{(n-1-i)}(\alpha)H^{\widehat{\psi}}_{i,l-1}(\alpha), \\ H^{\widehat{\psi}}_n(\alpha)=\sum\limits_{l=0}^{n-1}H^{\widehat{\psi}}_{n,l}(\alpha), \widehat{\Gamma}^{(n)}(\alpha)=H^{\widehat{\psi}}_n(\alpha)\widehat{\Gamma}(\alpha) \end{array} \tag{55}$$

for $n=1,2,\cdots$, where $\widehat{\psi}=-\psi(\alpha)$. The Bessel functions $J_\alpha(x)$ have many integral representations, for example

$$J_\alpha(x)=\frac{2\left(\frac{1}{2}x\right)^{\alpha}}{\pi^{\frac{1}{2}}\Gamma(\alpha+\frac{1}{2})}\int_0^1(1\text{-}t^2)^{a-\frac{1}{2}}\cos(xt)\,dt \tag{56}$$

and

$$\begin{array}{l} J_\alpha(x)=\frac{1}{\pi}\int_0^{\pi}\cos(x\sin\theta-\alpha\theta)d\theta \\ \quad -\frac{\sin\alpha\pi}{\pi}\int_0^{\infty}e^{-x\sinh t-\alpha t}dt. \end{array} \tag{57}$$

So $\frac{\partial^m}{\partial\alpha^m}J_\alpha(x)$ has the integral representation

$$\begin{array}{l} \frac{\partial^m}{\partial\alpha^m}J_\alpha(x)=\frac{1}{\pi}\int_0^{\pi}\theta^n\cos(\alpha\theta-x\sin\theta+\frac{m\pi}{2})d\theta \\ -\int_0^{\infty}\sum\limits_{l=0}^{m}\left(\begin{array}{c} m \\ l \end{array}\right)\frac{(-t)^l\pi^{m-l-1}}{e^{-x\sinh t-\alpha t}}\sin\frac{2\alpha\pi+(m-l)\pi}{2}dt. \end{array} \tag{58}$$

Expression (51) is not suitable for numerical calculations. Although numerical calculation for (58) is possible, it is not nearly as fast as (53).

Table V. Numerical results and time spent for computing $\frac{\partial^m}{\partial\alpha^m}J_\alpha(x)$

ALGO	x,α,m	Time_{32}, Err	Time_{64}, Err
(58)	$\frac{14}{3},\frac{7}{5},4$	$0.0625, 10^{-24}$	$0.2187, 10^{-63}$
(53)		$0.0, 10^{-33}$	$0.0156, 10^{-66}$
Integral value: $0.40812913683842034037286730580\cdots$			
ALGO	x,α,m	Time_{32}, Err	Time_{64}, Err
(58)	$\frac{17}{4},\frac{16}{3},3$	$0.0468, 10^{-26}$	$0.1250, 10^{-62}$
(53)		$0.0, 10^{-32}$	$0.0156, 10^{-65}$
Integral value: $-0.0053668024864463098372076 49\cdots$			
ALGO	x,α,m	Time_{32}, Err	Time_{64}, Err
(58)	$\frac{7}{4},\frac{4}{3},5$	$0.0781, 10^{-32}$	$0.1718, 10^{-64}$
(53)		$0.0, 10^{-33}$	$0.0156, 10^{-66}$
Integral value: $-0.38986414451230387929926 6271\cdots$			
ALGO	x,α,m	Time_{32}, Err	Time_{64}, Err
$(60)_L$	$\frac{14}{3},\frac{-1}{3},4$	$0.1093, 10^{-10}$	$0.1718, 10^{-14}$
$(60)_R$		$0.0, 10^{-33}$	$0.0156, 10^{-66}$
Integral value: $-4278.3869243573204008124641 27\cdots$			
ALGO	x,α,m	Time_{32}, Err	Time_{64}, Err
$(60)_L$	$\frac{25}{3},\frac{1}{3},4$	$0.0781, 10^{-32}$	$0.2031, 10^{-65}$
$(60)_R$		$0.0, 10^{-34}$	$0.0, 10^{-66}$
Integral value: $-11.467810359882213266842654 62\cdots$			
ALGO	x,α,m	Time_{32}, Err	Time_{64}, Err
$(60)_L$	$\frac{25}{7},\frac{3}{4},5$	$0.0625, 10^{-32}$	$0.1718, 10^{-65}$
$(60)_R$		$0.0156, 10^{-34}$	$0.0156, 10^{-66}$
Integral value: $14.8098121601852840672358350 3919\cdots$			

Using (56) and (50), we have

$$\begin{array}{l} \int_0^1(1-t^2)^{\alpha-\frac{1}{2}}\cos(xt)\,dt \\ \quad =\frac{1}{2}B(\frac{1}{2},\alpha+\frac{1}{2})_0F_1(\alpha+1,\frac{x^2}{4}) \end{array} \tag{59}$$

for $\alpha>-\frac{1}{2}$, so

$$\begin{array}{l} 2\int_0^1(1-t^2)^{\alpha-\frac{1}{2}}\cos(xt)\ln^m(1-t^2)dt \\ =\sum\limits_{l=0}^{m}\left(\begin{array}{c} m \\ l \end{array}\right)B_{m-l}(\frac{1}{2},\alpha+\frac{1}{2})\frac{\partial^l}{\partial\alpha^l}\left[{}_0F_1(\alpha+1,\frac{x^2}{4})\right]. \end{array} \tag{60}$$

Table V shows that $(60)_R$ can always achieve the specified precision, and the calculation speed is also fast. However, when α approaches $-\frac{1}{2}$ from the right, numerical calculation of $(60)_L$ can not achieve the specified precision.

V. Conclusion

In this article, we first establish recursive representations of the Stirling numbers of the first kind for Pochhammer k-symbol $(x)_{n,k}(\frac{1}{(x)_{n,k}})$ and its derivatives $A_{k,n,m}(x)(C_{k,n,m}(x))$ and $B_{k,n,m}(x)(D_{k,n,m}(x))$ with respect to x and k. Thus formulas for

$$\frac{\partial^{m_1+m_2+\cdots+m_p+n_1+n_2+\cdots+n_q}}{\partial a_1^{m_1}\partial a_2^{m_2}\cdots\partial a_p^{m_p}\partial b_1^{n_1}\partial b_2^{n_2}\cdots\partial b_q^{n_q}}\left[{}_pF_q(a,k;b,s;z)\right]$$

and

$$\frac{\partial^{m_1+m_2+\cdots+m_p+n_1+n_2+\cdots+n_q}}{\partial k_1^{m_1}\partial k_2^{m_2}\cdots\partial k_p^{m_p}\partial s_1^{n_1}\partial s_2^{n_2}\cdots\partial s_q^{n_q}}\left[{}_pF_q(a,k;b,s;z)\right]$$

for feasible computations are obtained. This is an essential progress over the multiple series mentioned in the introduction. These results, however, do not allow efficient computations. In order to improve the efficiency further, recursive algorithms for the derivatives of $(x)_{n,k}$ and $\frac{1}{(x)_{n,k}}$ with respect to x and k, and the parameter derivatives

$$\frac{\partial^{m_1+m_2+\cdots+m_p+n_1+n_2+\cdots+n_q}}{\partial a_1^{m_1}\partial a_2^{m_2}\cdots\partial a_p^{m_p}\partial b_1^{n_1}\partial b_2^{n_2}\cdots\partial b_q^{n_q}}\left[{}_pF_q(a,k;b,s;z)\right]$$

and

$$\frac{\partial^{m_1+m_2+\cdots+m_p+n_1+n_2+\cdots+n_q}}{\partial k_1^{m_1}\partial k_2^{m_2}\cdots\partial k_p^{m_p}\partial s_1^{n_1}\partial s_2^{n_2}\cdots\partial s_q^{n_q}}\left[{}_pF_q(a,k;b,s;z)\right]$$

are developed. As an example, the algorithms are also extended to the computation of the parameter derivatives of the Bessel functions and the modified Bessel functions. To illustrate the advantages and rationality of our results, numerical calculations in Mathematica are performed and data are provided. Numerical results show that the advantages are obvious in both of computational accuracy and efficiency. Some special integrals are also calculated by using the relationship between the integrals and their related special functions, and the advantages of accuracy and calculation efficiency are also obvious.

References

[1] Yury Brychkov,“Higher derivatives of the Bessel functions with respect to the order”, Int. Trans. Spec. Funct., vol. 27, no. 7, pp. 566-577, 2016.

[2] Apelblat A., Kravitsky N., “Integral representations of derivatives and integrals with respect to the order of the Bessel functions $J_v(t), I_v(t)$, the Anger function $\mathbf{J}_v(t)$ and the integral Bessel function $\mathbf{Ji}_v(t)$”, IMA J. Appl. Math. vol. 34, pp. 187-210, 1985.

[3] J. L. González-Santander, “Closed form expressions for derivatives of Bessel functions with respect to the order”, Journal of Mathematical Analysis and Applications, vol. 466, no. 1, November 2017.

[4] F. Oberhettinger, “On the derivative of Bessel functions with respect to the order”, J. Math. Phys., vol. 37, pp. 75-78, 1958.

[5] T.M. Dunster, “On the order derivatives of Bessel functions”, Constr Approx, vol. 46, pp. 47-68, 2017.

[6] J. L. Gonzalez-santander, “Reflection Formulas for Order Derivatives of Bessel Functions”, Proceedings of the American Mathematical Society, https://arxiv.org/pdf/1809.08124.pdf.

[7] JR. Airey, “The Bessel function derivatives $\partial\partial v \cdot J_v(x)$ and $\partial 2\partial v2 \cdot J_v(x)$”, Lond Edinburgh Dublin Philos Mag J Sci Ser 7.1935; vol. 19, no. 125, pp. 236-243, 1935.

[8] Yu. A. Brychkov & K. O. Geddes, “On the derivatives of the Bessel and Struve functions with respect to the order”, Integral Transforms and Special Functions, vol. 16, no. 3, pp. 187-198, 2005.

[9] J. Sesma, “Derivatives with respect to the order of the Bessel function of the first kind”,Proceedings of Theamerican Mathematical Society, https://arxiv.org/pdf/1401.4850.pdf.

[10] Cohl, H.S., “Derivatives with respect to the degree and order of associated Legendre functions for $|z| > 1$ using modified Bessel functions”, Integral Transforms Spec. Funct., vol. 21, no. 7-8, pp. 581-588, 2010.

[11] N. Shang, A. Li, Z. Sun, and H. Qin, “A Note on the Beta Function and Some Properties of Its Partial Derivatives”, IAENG International Journal of Applied Mathematics, vol. 44, no. 4, pp. 200-205, 2014.

[12] Brychkov Yu. A., “On the derivatives of the Legendre functions $P_\nu^\mu(z)$ and $Q_\nu^\mu(z)$ with respect to μ and ν”, Integral Transforms Spec. Funct., vol. 21, pp. 175-181, 2010.

[13] A. Li, and H. Qin, “The Representations on the Partial Derivatives of the Extended, Generalized Gamma and Incomplete Gamma Functions and Their Applications”, IAENG International Journal of Applied Mathematics, vol. 47, no. 3, pp. 312-318, 2017.

[14] R. Szmytkowski, “A note on parameter derivatives of classical orthogonal polynomials”, https:// arxiv.org/pdf/0901. 2639.pdf.

[15] L. U. Ancarani and, G. Gasaneo, “Derivatives of any order of the Gaussian hypergeometric function ${}_2F_1(a, b, c; z)$ with respect to the parameters a, b and c”, J. Phys. A: Math. Theor., vol. 42, 395208, 2009, https: //www. researchgate.net/publication/231045961

[16] L. U. Ancarani1 and G. Gasaneo, “Derivatives of any order of the confluent hypergeometric function ${}_1F_1(a, b, z)$ with respect to the parametera a or b”, Journal of Mathematical Physics, vol. 49, 063508, 2008.

[17] H. Kang, C. An, “Differentiation formulas of some hypergeometric functions with respect to all parameters”, Appl. Math. Comput. vol. 258, pp. 454-464, 2015.

[18] J. Froehlich, “Parameter derivatives of the jacobi polynomials and the gaussian hypergeometric function”, Int. Trans. Spec. Funct. vol. 2, pp. 253-266, 1994.

[19] L. U. Ancarani, G. Gasaneo, “Derivatives of any order of the Gaussian hypergeometric function ${}_pF_q(a_1, ..., a_p; b_1, ..., b_q; z)$ with respect to the parameters a_i and b_i” , J. Phys. A: Math. Theor., vol. 43, 085210, 2010.

[20] B. X. Fejzullahu, “Parameter derivatives of the generalized hypergeometric function”, Int. Trans. Spec. Funct., vol. 28, no. 11, pp. 781-788, 2017.

[21] Rafael Diaz, “Eddy Pariguan, On hypergeometric functions and Pochhammer k-symbol”, Divulgaciones Matematicas, vol. 15, no. 2, pp. 179-192, 2007.

[22] Emin O., Inci Ege, Hasmet G., “Biljana Jolevska-Tuneska, On partial derivatives of the incomplete beta function”, Applied Mathematics Letter, vol. 21, no. 7, pp. 675-681, 2008.

[23] S. Mubeen, G. M. Habibullah, “An Integral Representation of Some k-Hypergeometric Functions”, International Mathematical Forum, vol. 7, no. 4, pp. 203-207, 2012.

[24] Erdelyi A, Magnus W, Oberhettinger F and Tricomi F G, “Higher Transcendental Functions”, New York: McGraw-Hill, vol I, II and III, 1953.

[25] Exton H., “Multiple Hypergeometric Functions and Applications”, New York: Wiley, 1976.

[26] Landau L D and Lifshitz E M, “Quantum Mechanics: Non-Relativistic Theory”, Oxford: Pergamon, 2003, https://www.researchgate.net/publication/247317275.

[27] Flügge S., “Practical Quantum Mechanics I”, Berlin: Springer, 1971.

[28] Landau, L.J., “Bessel functions: monotonicity and bounds”, J. Lond. Math. Soc. vol. 2, no. 61, pp. 197-215, 2000.

[29] L. Landau,“Monotonicity and bounds on Bessel functions”, Electronic Journal of Differential Equations, Conf. 04, 2000, pp. 147-154,2000.

[30] A. Li and H. Qin, ”Some Transformation Properties of the Incomplete Beta Function and Its Partial Derivatives”, IAENG International Journal of Applied Mathematics, vol.49, no. 1, pp. 109-113, 2019.

[31] A. Li, Z. Sun and H. Qin, “The Algorithm and Application of the Beta Function and Its Partial Derivatives”, Engineering Letters, vol. 23, no. 3, pp. 140-144, 2015.

[32] A. Li, F. Qin and H. Qin, “The Calculation and Application of the Partial Derivatives of the Generalized Hypergeometric Function”, IAENG International Journal of Applied Mathematics, vol. 50, no. 3, pp. 713-719, 2020.

[33] Z. Sun and H. Qin, “Some Results on the Derivatives of the Gamma and Incomplete Gamma Function for Non-positive Integers”, IAENG International Journal of Applied Mathematics, vol. 47, no. 3, pp. 265-270, 2017.