Computation of Hypergeometric Functions

by
John Pearson
Worcester College

Dissertation submitted in partial fulfilment of the requirements for the degree of Master of Science in

Mathematical Modelling and Scientific Computing

University of Oxford

4 September 2009

Abstract

We seek accurate, fast and reliable computations of the confluent and Gauss hypergeometric functions ${}_1F_1(a;b;z)$ and ${}_2F_1(a,b;c;z)$ for different parameter regimes within the complex plane for the parameters a and b for ${}_1F_1$ and a, b and c for ${}_2F_1$, as well as different regimes for the complex variable z in both cases. In order to achieve this, we implement a number of methods and algorithms using ideas such as Taylor and asymptotic series computations, quadrature, numerical solution of differential equations, recurrence relations, and others. These methods are used to evaluate ${}_1F_1$ for all $z \in \mathbb{C}$ and ${}_2F_1$ for $|z| < 1$. For ${}_2F_1$, we also apply transformation formulae to generate approximations for all $z \in \mathbb{C}$. We discuss the results of numerical experiments carried out on the most effective methods and use these results to determine the best methods for each parameter and variable regime investigated.

We find that, for both the confluent and Gauss hypergeometric functions, there is no simple answer to the problem of their computation, and different methods are optimal for different parameter regimes. Our conclusions regarding the best methods for computation of these functions involve techniques from a wide range of areas in scientific computing, which are discussed in this dissertation. We have also prepared MATLAB code that takes account of these conclusions.

Acknowledgements

I would like to offer special thanks to my project supervisors at the University of Oxford, Mason Porter and Sheehan Olver, for their invaluable help and support during the last few months.

I would also like to thank the Numerical Algorithms Group, in particular David Sayers and Mick Pont, for providing the topic for this project, for supplying me with a copy of the NAG Toolbox, and for their assistance throughout. Thanks go to the National Institute of Standards and Technology, and in particular to Dan Lozier and Frank Olver, for kindly providing me with an advance copy of the new book, 'NIST Digital Library of Mathematical Functions', a preliminary version of which is now hosted at `http://dlmf.nist.gov/`, which has given me many ideas about special functions and their computation. I am also grateful to Andy Wathen for giving me his thoughts on this dissertation.

Further, I would like to thank the Engineering and Physical Sciences Research Council and the Numerical Algorithms Group for the provision of funding which has enabled me to take this course.

Finally, I would like to express my gratitude to those who have taught and supervised me this year and throughout my time at Oxford, to my family for their continued support, and to my friends for making this year a very enjoyable one.

Contents

1 Introduction

The computation of the hypergeometric function ${}_pF_q$, a special function encountered in a variety of applications, is frequently sought. However, aside from the most basic hypergeometric functions, this is an extremely difficult task in practice. The reason for this is that the non-trivial structure of the series that defines the function creates many numerical issues such as cancellation and round-off error, which become especially significant for certain ranges of the parameters and the variable. This results in many methods of numerical computation being ineffective for all but the simplest parameter and variable ranges.

We focus in this dissertation on computing the two most commonly used hypergeometric functions, the confluent hypergeometric function ${}_1F_1(a;b;z)$ and the Gauss hypergeometric function ${}_2F_1(a,b;c;z)$, which both suffer from these problems. The program which will be used to compute these functions will be MATLAB, which employs double precision arithmetic, so it is especially important for effective methods to be found to overcome the numerical issues involved.

The goal of this project is to carry out a comprehensive survey of methods for computing ${}_1F_1$ and ${}_2F_1$, and to determine how to choose appropriate methods for different parameter and variable ranges, resulting in reliable and fast computation for as large a range of the parameters (a, b for ${}_1F_1$ and a, b, c for ${}_2F_1$) and variable z as possible. The algorithms used are required to be accurate, fast and robust for specific parameter and variable regions for which they have been selected.

In this dissertation, we will test a large variety of approaches, such as a range of series methods, ways of numerically solving the differential equations that the hypergeometric functions satisfy and quadrature methods, as well as employing recurrence relations to attempt to use computations of relatively simple cases to obtain computations for extreme parameter cases. We will also test asymptotic series for ${}_1F_1$. For ${}_2F_1$, we also explore the use of transformation formulae and expansions for special parameter cases. We aim to combine techniques that work for specific parameter regimes in order to compile a package of methods that is effective for as large a part of the complex plane for each parameter and variable as possible.

This dissertation will explain the more successful methods tested for each function, with details of other methods implemented or analysed discussed in Appendices G and H. Details will be given of the background to these methods, how they were implemented, and the results obtained from test cases taken from a wide range of examples in the complex plane. An explanation will be provided as to why the methods might be successful, and a conclusion reached as to which methods should be used in what part of the complex plane. The conclusions will be based partly on the review of the author(s) who proposed the method and partly on results and observations from our implementation of them.

2 Background on hypergeometric functions

In this section, we will introduce properties of the generalized hypergeometric function that will be exploited in this project. The motivation for computing hypergeometric functions will be discussed, with details given of some of the practical applications of these functions (with further examples given in Appendix J). We will also discuss the numerical issues that make the problem of their computation a difficult one.

2.1 Relevant properties of hypergeometric functions

The hypergeometric function ${}_pF_q$ is defined in [37] as follows for $a_1, ..., a_p, b_1, ..., b_q, z \in \mathbb{C}$:

$$ {}_pF_q(a_1, ..., a_p; b_1, ..., b_q; z) = \sum_{j=0}^{\infty} \frac{(a_1)_j ... (a_p)_j}{(b_1)_j ... (b_q)_j} \frac{z^j}{j!}, \tag{2.1} $$

where, for some parameter μ, the **Pochhammer symbol** $(\mu)_j$ is defined as

$$ (\mu)_0 = 1, \quad (\mu)_j = \mu(\mu + 1)...(\mu + j - 1), \quad j = 1, 2, ... \ . $$

For the remainder of this dissertation, the values of a_j, b_j, z will be complex unless otherwise specified. However, the hypergeometric function is not defined if any $b_j,\ j = 1, ..., q$ are real and equal to a non-positive integer, and there are numerical issues in its computation if one or more values of b_j are close to a non-positive integer.

The generalized hypergeometric function ${}_pF_q$ is known to satisfy the following differential equation [64]:

$$ \left[z\frac{\mathrm{d}}{\mathrm{d}z} \left[\left(z\frac{\mathrm{d}}{\mathrm{d}z} + b_1 - 1 \right) ... \left(z\frac{\mathrm{d}}{\mathrm{d}z} + b_q - 1 \right) \right] - z \left(z\frac{\mathrm{d}}{\mathrm{d}z} + a_1 \right) ... \left(z\frac{\mathrm{d}}{\mathrm{d}z} + a_p \right) \right] w = 0. \tag{2.2} $$

A great number of common mathematical functions are expressible in terms of hypergeometric functions. For example, as detailed in [3, 37],

$$
\begin{aligned}
{}_pF_p(a_1, ..., a_p; a_1, ..., a_p; z) &= e^z, \ p \in \mathbb{Z}^+ \cup \{0\}, \\
{}_1F_0(a; \ ; z) &= (1-z)^{-a}, \\
{}_2F_1\left(\frac{1}{2}, \frac{1}{2}; \frac{3}{2}; z\right) &= \frac{1}{z}\sin^{-1} z,
\end{aligned}
\tag{2.3}
$$

where $\mathbb{Z}^+$ denotes the set of positive integers.

A number of other special functions are also expressible in terms of hypergeometric functions. For example, as noted in [43],

$$
J_\nu(x) = \sum_{j=0}^{\infty} \frac{(-1)^j}{j!\Gamma(j+\nu-1)} \left(\frac{x}{2}\right)^{2j+\nu} = \frac{\left(\frac{x}{2}\right)^\nu}{\Gamma(\nu+1)} \, {}_0F_1\left(\ ; \nu+1; -\frac{x^2}{4}\right), \tag{2.4}
$$

where $J_\nu(x)$ is the **Bessel function of the first kind** with parameter ν, and the **Gamma function** $\Gamma(x)$ is defined in [57] as

$$
\Gamma(x) = \int_0^\infty t^{x-1} e^{-t} \mathrm{d}t \tag{2.5}
$$

for $\mathrm{Re}(x) > 0$, and by the reflection formula

$$
\Gamma(x) = \frac{\pi}{\Gamma(1-x)\sin(\pi(1-x))} \tag{2.6}
$$

for $\mathrm{Re}(x) \leq 0$.

We note that, in the above notation, there are empty spaces between the two semi-colons in ${}_1F_0(a; \ ; z)$ and before the first semi-colon in ${}_0F_1\left(\ ; \nu+1; -\frac{x^2}{4}\right)$ due to the fact that, respectively, $q = 0$ and $p = 0$ in these cases, in the notation of (2.1).

The derivative of the hypergeometric function ${}_pF_q(a_1, ..., a_p; b_1, ..., b_q; z)$ is given by

$$
\frac{\mathrm{d}}{\mathrm{d}z}\left[{}_pF_q(a_1, ..., a_p; b_1, ..., b_q; z)\right] = \frac{a_1 ... a_p}{b_1 ... b_q} \, {}_pF_q(a_1+1, ..., a_p+1; b_1+1, ..., b_q+1; z), \tag{2.7}
$$

and the n-th derivative is given by

$$
\frac{\mathrm{d}^n}{\mathrm{d}z^n}\left[{}_pF_q(a_1, ..., a_p; b_1, ..., b_q; z)\right] = \frac{(a_1)_n ... (a_p)_n}{(b_1)_n ... (b_q)_n} \, {}_pF_q(a_1+n, ..., a_p+n; b_1+n, ..., b_q+n; z).
$$

An important property that we will need to use is the convergence criteria of the hypergeometric functions depending on the values of p and q. The **radius of convergence** of a

series of a variable z is defined as a value r_c such that the series converges if $|z - d_c| < r_c$ and diverges if $|z - d_c| > r_c$, where d_c, in this case 0, is the centre of the disc of convergence. For the hypergeometric function, provided a_j and b_j are not non-positive integers for any j, the relevant convergence criteria stated below can be derived using the **ratio test**, which determines the absolute convergence of the series using the limit of the ratio of two consecutive terms, and are detailed in [37].

- If $p \leq q$, then the ratio of coefficients of z^k in the Taylor series of the hypergeometric function ${}_pF_q$ tends to 0 as $k \to \infty$; so the radius of convergence is ∞, so that the series converges for all values of $|z|$. Hence, ${}_pF_q$ is **entire**. In particular, the radius of convergence for ${}_0F_1$ and ${}_1F_1$ is ∞.

- If $p = q + 1$, the ratio of coefficients of z^k tends to 1 as $k \to \infty$, so the radius of convergence is 1, so that the series converges only if $|z| < 1$. In particular the radius of convergence of ${}_2F_1$ is 1.

- If $p > q + 1$, the ratio of coefficients of z^k tends to ∞ as $k \to \infty$, so the radius of convergence is 0, so that the series does not converge for any value of $|z|$.

We will seek approximations to the relevant hypergeometric functions for $|z|$ within the radii of convergence, and then apply analytic continuation to compute them in the rest of the complex plane where appropriate. For $p = q + 1$, as stated in [37] there is a further restriction for convergence on the unit disc; the series only converges absolutely at $|z| = 1$ if $\mathrm{Re}(\sum_{j=1}^{q} b_j - \sum_{j=1}^{p} a_j) > 0$, so the selection of values for a_j and b_j must reflect that.

In Figure 1, plots are shown of ${}_1F_1$ and ${}_2F_1$ for real z, and for a selection of parameter values.

2.2 Motivation for the computation of hypergeometric functions

The computation of the hypergeometric function is frequently sought due to the wide range of practical problems in which it appears. It arises, for example, in photon scattering from atoms [27], networks [54], Coulomb wave functions [10], binary stars [56], finance [13] and many others. Some examples from practical applications are detailed in Appendix J. Due to this wide range of applications, it is useful to provide a survey of work carried out on

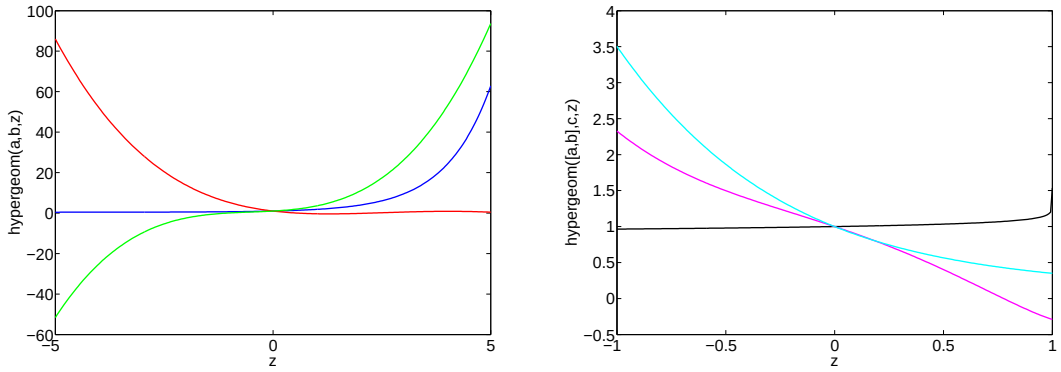

Figure 1: Graphs of ${}_1F_1(a;b;z)$, generated using MATLAB, for real $z \in [-5,5]$ with $(a,b) = (0.1,0.2)$ (dark blue), $(a,b) = (-3.8,1.5)$ (red) and $(a,b) = (-3,-2.5)$ (green), and ${}_2F_1(a,b;c;z)$ for real $z \in [-1,1]$ with $(a,b,c) = (0.1,0.2,0.4)$ (black), $(a,b,c) = (-3.6,-0.7,-2.5)$ (purple) and $(a,b,c) = (-5,1.5,6.2)$ (sky blue).

computing hypergeometric functions, and to discuss which methods are likely to work well for particular parameter regimes, as well as to supply information on how to test the reliability of a routine, test cases that a routine might have difficulty computing (as described in Appendices B and C), and how to evaluate other special functions required for computation of hypergeometric functions (as detailed in Appendix I). Research of this form will be useful in many ways. Firstly, the Numerical Algorithms Group who sponsored this project, will benefit from research on computing hypergeometric functions being carried out, to help them achieve their goal of writing a package for the NAG Library. Secondly, programmers working with software which does not have an built-in hypergeometric function evaluator may be able to use the theory of computing these functions to write such a program for themselves.

Another important reason why research into this area is desirable is that the state-of-the-art software currently used for special functions has not yet been perfected. This is illustrated by a number of cases when MATLAB R2008b is asked to compute the function ${}_2F_1(a,b;c;z)$ for certain values of a, b, c and z. Firstly, although the MATLAB routine for computing hypergeometric functions, 'hypergeom', is generally slow but tolerable for all parameter and variable values (MATLAB usually took around 10–15 seconds to compute a confluent or Gauss hypergeometric function the first time after loading the program on the processor used), the problem can sometimes be serious. For example, when the command `hypergeom([1,0.9],2,exp(1i*pi/3))` is used, over 5 minutes is taken for MATLAB

to compute the solution on the processor used (an Intel(R) Atom(TM) CPU N270, with processor speed 1.60GHz). This problem will be resolved by making use of an analytic continuation formula discussed in Section 4.7.

Secondly, there is a major issue with the MATLAB routine in some cases where c is close to a negative integer. For example, when `hypergeom([-1,-1.5],-2.000000000000001,0.5)` is called, a set of parameters for which the Gauss hypergeometric series has only 2 non-zero terms, the answer returned using 16 digit arithmetic is 0.621859216769114, giving only 2 digit accuracy on the correct answer of 0.625000000000000, which can be found by manual calculation. This motivates the need for research into effective methods that will compute hypergeometric functions accurately. Many methods considered in this project, such as those discussed in Sections 4.2 and 4.3, will be able to assist with this particular computation.

There are also problems that arise when computing ${}_1F_1(a;b;z)$ in MATLAB. When `hypergeom(1,200,1)` is called, the answer obtained is 6.69×10^{299} to 3 significant figures. By examination of (3.1), we can see that each term of the power series of ${}_1F_1(1;200;1)$ is a smaller positive real number than each term of ${}_1F_1(1;1;1)$, for example, and so the value ${}_1F_1(1;200;1)$ is smaller than that of ${}_1F_1(1;1;1)$. From (2.3), we can see that ${}_1F_1(1;1;1) = e = 2.72$ to 3 significant figures, and hence MATLAB's evaluation of ${}_1F_1(1;200;1)$ is incorrect. Furthermore, Mathematica will not generate an evaluation for ${}_1F_1(1;200;1)$ either. The methods used by MATLAB and Mathematica are not publicly known, so it is important to devise a package of routines that do not suffer from these problems.

3 Computation of the confluent hypergeometric function ${}_1F_1(a;b;z)$

In this section, we discuss the most powerful methods implemented to accurately and efficiently evaluate the confluent hypergeometric function, before providing recommendations as to the most effective approaches for each parameter regime. Other methods that we have implemented or analysed are discussed in Appendix G.

3.1 Properties of ${}_1F_1(a;b;z)$

The **confluent hypergeometric function** ${}_1F_1(a;b;z)$, also denoted by $M(a;b;z)$, is defined as

$$M(a;b;z) = \sum_{j=0}^{\infty} \frac{(a)_j}{(b)_j} \frac{z^j}{j!}, \tag{3.1}$$

which converges for any $z \in \mathbb{C}$, and is defined for any $a \in \mathbb{C}$, $b \in \mathbb{C}\backslash\{\mathbb{Z}^- \cup \{0\}\}$, where $\mathbb{Z}^-$ denotes the set of negative integers.

It should be noted that $M(a;b;0) = 1$ for any a and $b \notin \mathbb{Z}^- \cup \{0\}$. If $b = n \in \mathbb{Z}^- \cup \{0\}$, the series is not defined. On the other hand, if $a = n \in \mathbb{Z}^- \cup \{0\}$, then this series is given by a polynomial of degree $-n$ in z.

The function $M(a;b;z)$ satisfies the following differential equation, derived from (2.2):

$$z\frac{\mathrm{d}^2 w}{\mathrm{d}z^2} + (b-z)\frac{\mathrm{d}w}{\mathrm{d}z} - aw = 0, \tag{3.2}$$

for all b apart for when $b \in \mathbb{Z}^- \cup \{0\}$; this case is resolved by the fact that, as noted in [51], the following is a solution, including when b is a negative integer:

$$\mathbf{M}(a;b;z) = \sum_{j=0}^{\infty} \frac{(a)_j}{\Gamma(b+j)} \frac{z^j}{j!}. \tag{3.3}$$

When $b \notin \mathbb{Z}^- \cup \{0\}$, then the following expression holds:

$$M(a;b;z) = \Gamma(b)\mathbf{M}(a;b;z).$$

Now, $M(a;b;z)$ is closely related to another standard solution $U(a;b;z)$, which is defined as the solution to (3.2) with the property

$$U(a;b;z) \sim z^{-a}, \ z \to \infty, \ \ |\arg z| \leq \frac{3}{2}\pi - \delta, \tag{3.4}$$

with $0 < \delta \ll 1$ (i.e. δ is an arbitrary small positive constant), as discussed in [37]. The function $U(a;b;z)$ also has the expression

$$U(a;b;z) = {}_2F_0\left(a, 1+a-b; \ ; -\frac{1}{z}\right),$$

as stated in [3], where the space between the two colons is due to the fact that $q = 0$ in the notation of (2.1).

As explained in [3], $U(a;b;z)$ has a branch point at $z = 0$, with a branch cut in the z-plane along the interval $(-\infty, 0]$; when $m \in \mathbb{Z}$, the set of integers, the following expression holds:

$$U(a;b;ze^{2\pi im}) = \frac{2\pi i e^{-\pi ibm}\sin(\pi bm)}{\Gamma(1+a-b)\sin(\pi b)}\mathbf{M}(a;b;z) + e^{-2\pi ibm}U(a;b;z).$$

Hence, as discussed in [3, 68], $M(a;b;z)$ and $U(a;b;z)$ are related by

$$U(a;b;z) = \frac{\Gamma(1-b)}{\Gamma(1+a-b)}M(a;b;z) + \frac{\Gamma(b-1)}{\Gamma(a)}z^{1-b}M(1+a-b;2-b;z),\ b \notin \mathbb{Z}, \quad (3.5)$$

$$M(a;b;z) = \frac{\Gamma(b)e^{\mp a\pi i}}{\Gamma(b-a)}U(a;b;z) + \frac{\Gamma(b)e^{\pm(b-a)\pi i}}{\Gamma(a)}e^z U(b-a;b;e^{\pm\pi i}z), \qquad b \notin \mathbb{Z}, \quad (3.6)$$

so methods for computing $U(a;b;z)$ are also useful for computing ${}_1F_1$.

The following transformations are also useful when certain values of a or b prove difficult computationally:

$$M(a;b;z) = e^z M(b-a;b;z), \quad (3.7)$$

$$U(a;b;z) = z^{1-b}U(a-b+1;2-b;z). \quad (3.8)$$

As detailed in [3, 5, 40], the confluent hypergeometric function is related to various elementary and special functions as follows:

$$\begin{aligned} M(a;a;z) &= e^z, \\ M\left(\nu+\frac{1}{2};2\nu+1;2z\right) &= \left(\frac{z}{2}\right)^{-\nu}e^z I_\nu(z), \\ M(1;a+1;z) &= az^{-a}e^z\gamma(a,z), \\ M\left(\frac{1}{2};\frac{3}{2};-z^2\right) &= \frac{\sqrt{\pi}}{2z}\mathrm{erf}(z), \end{aligned} \quad (3.9)$$

where the modified Bessel function $I_\nu(z)$, incomplete gamma function $\gamma(a,z)$ and error function $\mathrm{erf}(z)$ are defined by, respectively,

$$I_\nu(z) = i^{-\nu}J_\nu(iz) = i^{-\nu}\sum_{j=0}^{\infty}\frac{1}{j!\Gamma(\nu+j+1)}\left(\frac{iz}{2}\right)^{2j+\nu}, \quad (3.10)$$

$$\gamma(a,z) = \int_0^z t^{a-1}e^{-t}\mathrm{d}t, \quad (3.11)$$

$$\mathrm{erf}(z) = \frac{2}{\sqrt{\pi}}\int_0^z e^{-t^2}\mathrm{d}t.$$

3.2 Taylor series

The simplest method for computing the confluent hypergeometric function, and the first method we try, is using the basic power series definition. There are two methods we employ to compute the power series

$$M(a;b;z) = \sum_{j=0}^{\infty} \underbrace{\frac{(a)_j}{(b)_j} \frac{1}{j!}}_{A_j} z^j. \tag{3.12}$$

These methods used for computing the basic terms in the series A_j are detailed below.

Method (a): Computation can be carried out using the following procedure:

$$A_0 = 1, \quad S_0 = A_0,$$
$$A_{j+1} = A_j \times \frac{a+j}{b+j} \times \frac{z}{j+1}, \quad S_{j+1} = S_j + A_{j+1},\ j = 0, 1, 2, \ldots ,$$

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

When programming in MATLAB, a **stopping criterion** needs to be specified. A common stopping criterion is to stop computing terms when $\frac{|A_{N+1}|}{|S_N|} < tol$ for some tol and some N and to return S_N, our approximation of $M(a;b;z)$, as the solution (as in [44]). This is equivalent to truncating the series

$$S_\infty = \sum_{j=0}^{\infty} \frac{(a)_j}{(b)_j} \frac{z^j}{j!}. \tag{3.13}$$

However, our investigation has shown that this is not sufficient in all cases. For example, we consider an example of ${}_1F_1(a;b;z)$ where a is very close to a negative integer $-m$ (by which we mean, as a guide, where the modulus of the difference between a and $-m$ is $O(tol)$). In this case, the $(m+2)$-nd term in the series, equal to $\frac{(a+m)}{(b+m)} \frac{z^m}{m!}$, is likely to be very small compared to the sum of the previous terms, but the subsequent terms might still contribute significantly to the solution. The stopping criterion we use is therefore met when $\frac{|A_{N+1}|}{|S_N|} < tol$ and $\frac{|A_N|}{|S_{N-1}|} < tol$. In other words, we need two consecutive terms to be small compared to the sum already computed.

Method (b): The following recurrence relation deducing the next approximation in terms of the previous two [44],

$$\begin{aligned} S_{-1} = S_0 = 1, \quad S_1 &= \frac{a}{b}z, \\ r_j = \frac{a+j-1}{j(b+j-1)}, \quad j &= 2, 3, \dots , \\ S_j = S_{j-1} + (S_{j-1} - S_{j-2})r_j z, \quad j &= 2, 3, \dots . \end{aligned}$$

By the same reasoning as for method (a), the stopping criterion is to truncate the series when $\frac{|S_{N+1}-S_N|}{|S_N|} < tol$ and $\frac{|S_N-S_{N-1}|}{|S_{N-1}|} < tol$ for some tol and some N, and to return S_N. As in Method (a), this is equivalent to truncating the series (3.13).

For both of these methods, and for every other series method that will be discussed in this project unless otherwise specified, we instructed MATLAB to terminate the computation when 500 terms had been computed if the stopping criterion had not been satisfied already. In all tables in this project, when this case arose, '500 terms computed' is specified. For the remainder of this project, we take $tol = 10^{-15}$, motivated by our desire for accuracy of 15 decimal places, and the fact that the smallest number that MATLAB is able to compute, *eps*, is equal to roughly 2.2×10^{-16}.

From Table 1, it can be deduced that, for the most part, methods (a) and (b) generate similar results, and the same number of terms for most computations. However, we find that the second method is in general more effective when carrying out computations involving small parameters, for example in the second row in the table.

Both methods appear to be successful for large values of $|b|$; this is fairly unsurprising as this would mean that, by examining (3.1), the coefficients of the powers of z^j would decay fast as j becomes large, so few terms are required to obtain an accurate computation for $M(a; b; z)$. However, this is not the case for large values of $|a|$ when $|b|$ is small, as shown by Table 1, as the series does not converge as quickly as for small $|a|$, so computation would be more susceptible to round-off error. Therefore, recurrence relations, which we discuss in Section 3.8, will have to be used in order to obtain a computation of acceptable accuracy, (we will usually treat 10 digit accuracy as 'acceptable', motivated by the work in [12]). It

Case	(a,b,z)	Correct $M(a;b;z)$	Time	Method (a) and (b) ($tol = 10^{-15}$)	Acc.	N	Time
1	(0.1,0.2,0.5)	1.317627178278510	11.164293s	1.317627178278510	16	15	0.032802s
				1.317627178278510	16	15	0.051618s
6	$(10^{-8},10^{-12},$ $-10^{-10}+10^{-12}i)$	0.999999000000000 $+0.000000010000000i$	9.930426s	0.999999000088899 $+0.000000010000000i$	11	3	0.028194s
				0.999999000000000 $+0.000000010000000i$	16	3	0.046520s
9	(500,511,10)	$1.779668553337393\times10^{-4}$	11.528364s	$1.779668553337393\times10^{-4}$	16	46	0.046847s
				$1.779668553337393\times10^{-4}$	16	46	0.077375s
10	(8.1,10.1,100)	$1.724131075992688\times10^{41}$	10.388182s	$1.724131075992686\times10^{41}$	15	188	0.066285s
				$1.724131075992686\times10^{41}$	15	188	0.082231s
13	$(-60,1,10)$	10.04854112964948	11.359146s	-35.241346779094869	0	58	0.058127s
				-13.585500872106090	0	58	0.088579s
15	$(60,1,-10)$	$-6.713066845459067\times10^{-4}$	11.769413s	$1.608258431433813\times10^{5}$	0	97	0.054126s
				$4.161733968914763\times10^{5}$	0	96	0.078579s
19	(500,1,-5)	0.001053895943365	11.891125s	$1.669453216927715\times10^{26}$	0	128	0.061397s
				$1.319078645590728\times10^{26}$	0	128	0.107982s
21	$(20,-10+10^{-9},$ $-2.5)$	$8.857934344815256\times10^{9}$	10.971249s	$8.857934347919209\times10^{9}$	9	49	0.053288s
				$8.857934341268038\times10^{9}$	9	49	0.080028s
30	$(2+8i,-150+i,$ $150)$	$-9.853780031496243\times10^{135}$ $+3.293888962100131\times10^{136}i$	11.731429s	$-9.853780031496170\times10^{135}$ $+3.293888962100122\times10^{136}i$	13	409	0.078265s
				$-9.853780031496204\times10^{135}$ $+3.293888962100115\times10^{136}i$	14	409	0.112663s
32	$(-5,2,-100+1000i)$	$7.196140446954445\times10^{11}$ $-1.233790613611111\times10^{12}i$	12.019623s	$7.196140446954443\times10^{11}$ $-1.233790613611111\times10^{12}i$	15	7	0.031462s
				$7.196140446954445\times10^{11}$ $-1.233790613611111\times10^{12}i$	16	7	0.069256s

Table 1: Results obtained from programming the above algorithms into MATLAB, for a selection of test cases from Appendix B. Shown is the value obtained from the routine 'hypergeom' in MATLAB, and verified in Mathematica 7, and the time taken to arrive at this result in MATLAB. Also shown are the computations obtained using Taylor series methods (a) and (b), the number of digits of accuracy generated (denoted as Acc. above), as well as the number of terms N taken before the stopping criterion was applied, and the time taken for each computation. The results for all test cases are shown in Appendix E.

should also be noted that the results are generated at least 100 times faster than they are generated using 'hypergeom'.

As the fourth row in Table 1 shows, the Taylor series method can provide accurate computation for up to and beyond $|z| = 100$ (although the approximation becomes less accurate as $|z| \to \infty$), providing that, as in this case, the value of $|b|$ is at least as large as $|a|$. As $|z|$ becomes large, another feature of the Taylor series method is the increase in the number of terms of the Taylor series required for the computation of ${}_1F_1$, as illustrated by Figure 2.

One notable case for which both Taylor series methods are very inefficient is when $\mathrm{Re}(a)$ and $\mathrm{Re}(z)$ are of reasonably large magnitude (i.e. of order at least 10), but are of different signs. This problem is illustrated in particular in the fifth and sixth rows in Table 1, for

$(a, z) = (-60, 10)$ and $(60, -10)$, for which both Taylor series methods produce results that are not even of the same order as the correct answer, whereas each method works very well for $(a, z) = (60, 10)$ and $(-60, -10)$, as shown in Appendix E. This problem can be resolved by exploiting the use of Buchholz polynomials, which we discuss in Section 3.4.

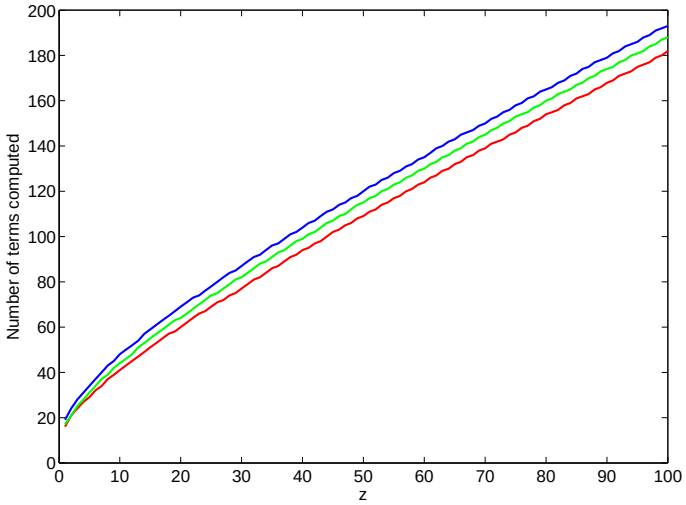

Figure 2: Number of terms computed using Taylor series method (a) for computing ${}_1F_1(a; b; z)$ for real $z \in [1, 100]$, when $a = 2$, $b = 3$ (red), $a = 2 + 10i$, $b = 10 + 5i$ (green) and $a = 20$, $b = 15$ (blue). We carried out the computation of ${}_1F_1$ for these parameters with 15 digit accuracy.

3.3 Writing the confluent hypergeometric function as a single fraction

As illustrated by the performance of the Taylor series method (a) on test case 6, which has a relatively small number of terms that require computation, the methods of Section 3.2 can be vulnerable in particular to parameter values with small modulus, even when the computation should be fairly straightforward. The method discussed in this section aims to provide an alternative method to those in Section 3.2 that is also based on the basic series definition (3.1) of the confluent hypergeometric function.

This method, explained in [45, 46], expresses the hypergeometric series of ${}_1F_1(a; b; z)$ as a single fraction, rather than a sum of many fractions. The sum S_j of the first $j + 1$ terms of

the hypergeometric series up to the term in z^j can be expressed, for $j = 0, 1, 2, 3$, as follows:

$$S_0 = \frac{0+1}{1},$$
$$S_1 = \frac{b+az}{b},$$
$$S_2 = \frac{(b+az)(2)(b+1)+a(a+1)z^2}{2b(b+1)},$$
$$S_3 = \frac{\overbrace{[(b+az)(2)(b+1)+a(a+1)z^2]}^{\alpha_3}+\overbrace{a(a+1)(a+2)z^3}^{\beta_3}}{\underbrace{(2)(3)b(b+1)(b+2)}_{\gamma_3}},$$

where α_3, β_3 and γ_3 can be calculated using (3.14)–(3.16) below. Taking $\alpha_0 = 0$, $\beta_0 = 1$, $\gamma_0 = 1$, $\zeta_0 = 1$, and defining ζ_j to be the j-th approximation, we can apply the following recurrence relations:

$$\alpha_j = (\alpha_{j-1} + \beta_{j-1}) \times j \times (b + j - 1), \tag{3.14}$$
$$\beta_j = \beta_{j-1} \times (a + j - 1) \times z, \tag{3.15}$$
$$\gamma_j = \gamma_{j-1} \times j \times (b + j - 1), \tag{3.16}$$
$$\zeta_j = \frac{\alpha_j + \beta_j}{\gamma_j}, \tag{3.17}$$

for $j = 1, 2, \ldots$. We program this method into MATLAB in order to generate a sequence of approximations to $M(a; b; z)$, $\{\zeta_j,\ j = 1, 2, ...\}$, using the stopping criterion, similar to that described in Section 3.2, that for the series to be terminated, $\frac{|\zeta_{j+1}-\zeta_j|}{|\zeta_j|}$ and $\frac{|\zeta_j-\zeta_{j-1}|}{|\zeta_{j-1}|}$ must be less than the prescribed tolerance $tol = 10^{-15}$.

The motivation behind this method is the fact that significant round-off error in division is produced by computing the individual terms of $M(a; b; z)$ using a number of other methods. Therefore, applying a method that only requires a single division to compute an approximation to $M(a; b; z)$ can be potentially advantageous.

From Table 2, one can conclude that the methods of Section 3.2 are more successful in generating accurate computations of $M(a; b; z)$ for a wide range of the parameters and the variable than the method described in this section. One possible explanation for this is that, in particular when the modulus of the parameter values are increasingly large, the numerator and denominator of ζ_j become very large for a relatively small j, and so the round-off error

Case	(a,b,z)	Correct $M(a;b;z)$	Acc. 1/2	Single fraction (tol=1e-15)	Acc.	N	Time taken
1	(0.1,0.2,0.5)	1.317627178278510	16/16	1.317627178278509	14	15	0.042757s
6	$(10^{-8},10^{-12},$ $-10^{-10}+10^{-12}i)$	0.999999000000000 $+0.000000010000000i$	11/16	0.999999000000000 $+0.000000010000000i$	16	3	0.028864s
9	(500,511,10)	$1.779668553337393\times10^{-4}$	16/16	$1.779668553337394\times10^{-4}$	15	45	0.050234s
13	$(-60,1,10)$	10.04854112964948	0/0	-19.30658974826999	0	58	0.065559s
15	$(60,1,-10)$	$-6.713066845459067\times10^{-4}$	0/0	$6.748784369464462\times10^{4}$	0	97	0.073657s
19	(500,1,-5)	0.001053895943365	0/0	$-4.223914178002353\times10^{32}$	0	90	0.061492s
21	$(20,-10+10^{-9},-2.5)$	$8.857934344815256\times10^{9}$	9/9	$8.857934344315682\times10^{9}$	10	49	0.074883s
30	$(2+8i,-150+i,$ 150)	$-9.853780031496243\times10^{135}$ $+3.293888962100131\times10^{136}i$	13/14	500 terms computed	N/A	N/A	N/A
32	$(-5,2,-100+1000i)$	$7.196140446954445\times10^{11}$ $-1.233790613611111\times10^{12}i$	15/16	$7.196140446954445\times10^{11}$ $-1.233790613611111\times10^{12}i$	16	7	0.041258s

Table 2: Table showing the accuracy of Taylor series methods (a) and (b), denoted as Acc. 1/2 above, as explained in Section 3.2, as well as the results of the single fraction method of this section and its accuracy. Also shown are the number of terms required to generate the solution using the single fraction method described in this section, and the time taken to do so. The label '500 terms computed' means that the stopping criterion for this method had not been satisfied after 500 terms have been computed. The results from applying this method on all test cases are shown in Appendix E.

will become significant when carrying out the division. Also, relatively few approximations will be able to be carried out before either the numerator or the denominator becomes very large.

Unsurprisingly however, the method is useful if the value of $|b|$ is small (especially when $|b| \lesssim 1$), provided $|a|$ is not too large. If Taylor series methods are applied when $|b|$ is small, the round-off error in division may well become costly if a large number of terms are significantly large. Therefore, a method with a single division is likely to aid accurate computation in this case, as the effect of round-off error is reduced; this hypothesis is verified by the results for this method.

3.4 Buchholz polynomials

In this Section, we discuss three methods based on Buchholz polynomials. One of these methods in particular is very effective when $\mathrm{Re}(a)$ and $\mathrm{Re}(z)$ have opposite signs, which will prove useful when compiling our package for computing $M(a;b;z)$.

As stated in [1, 2], $M(a;b;z)$ has the following known expansion in terms of **Buchholz**

polynomials $p_j(b,z)$:

$$M(a;b;z) = \Gamma(b)e^{z/2}2^{b-1}\sum_{j=0}^{\infty} p_j(b,z)\frac{J_{b-1+j}(\sqrt{z\{2b-4a\}})}{(z\{2b-4a\})^{\frac{1}{2}(b-1+j)}}, \tag{3.18}$$

where J_ν denotes the Bessel function of the first kind as defined in (2.4), and

$$p_j(b,z) = \frac{(iz)^j}{j!}\sum_{s=0}^{\lfloor \frac{j}{2} \rfloor} \left(\begin{array}{c} j \\ 2s \end{array} \right) f_s(b)g_{j-2s}(z), \tag{3.19}$$

with

$$f_0(b) = 1, \quad f_s(b) = -\left(\frac{b}{2}-1\right)\sum_{j=0}^{s-1}\left(\begin{array}{c} 2s-1 \\ 2j \end{array} \right)\frac{4^{s-j}\left|B_{2(s-j)}\right|}{s-j}f_j(b), \ s = 1,2,\dots,$$

$$g_0(z) = 1, \quad g_s(z) = -\frac{iz}{4}\sum_{j=0}^{\lfloor \frac{s-1}{2} \rfloor}\left(\begin{array}{c} s-1 \\ 2j \end{array} \right)\frac{4^{j+1}\left|B_{2(j+1)}\right|}{j+1}g_{s-2j-1}(z), \quad s = 1,2,\dots.$$

The coefficients B_j denote the **Bernoulli numbers**, which are defined as the sequence of numbers such that

$$\frac{z}{e^z-1} = \sum_{j=0}^{\infty} B_j\frac{z^j}{j!}.$$

As explained in [2], (3.18) can also be written as

$$M(a;b;z) = \Gamma(b)e^{z/2}2^{b-1}\sum_{j=0}^{\infty} D_j z^j\frac{J_{b-1+j}(\sqrt{z\{2b-4a\}})}{(z\{2b-4a\})^{\frac{1}{2}(b-1+j)}}, \tag{3.20}$$

where the coefficients D_j are

$$\begin{aligned} &D_0 = 1, \quad D_1 = 0, \quad D_2 = \frac{b}{2}, \\ &jD_j = (j-2+b)D_{j-2} + (2a-b)D_{j-3}, \quad j = 3,4,\dots, \end{aligned} \tag{3.21}$$

giving an expression for the coefficients C_j in terms of a recurrence relation.

The expressions in (3.18) and (3.20) provide expansions for the confluent hypergeometric function, which is known to be difficult to compute, in terms of Bessel functions, which are much easier to compute.

An alternative expression given in [2] is

$$M(a;b;z) = e^{z/2}\sum_{j=0}^{\infty} p_j(b,z)\frac{{}_0F_1(\ ;b+j;\chi)}{2^j(b)_j}, \quad \chi = z\left(a-\frac{b}{2}\right), \tag{3.22}$$

providing an expansion of ${}_1F_1$ in terms of a simpler hypergeometric function ${}_0F_1$.

For the rest of this section, we will denote method 1 as the method of computing (3.18), method 2 the method of calculating (3.20) and method 3 as the method of computing (3.22). For methods 1 and 3, no more than 170 terms were used to approximate the series of (3.18) and (3.22) respectively. Instances where this many terms have been computed without the individual terms becoming sufficiently small (10^{-15} multiplied by the sum of the previous terms), are indicated in Table 3 and Appendix E.

Case	(a,b,z)	Correct $M(a;b;z)$	Time taken	Method 1/2/3 ($tol = 10^{-15}$)	Acc.	N	Time taken
1	(0.1,0.2,0.5)	1.317627178278510	11.164293s	170 terms computed	N/A	N/A	N/A
				500 terms computed	N/A	N/A	N/A
				1.317839415371721	4	11	0.302097s
2	$(-0.1, 0.2, 0.5)$	0.695536565102261	11.048696s	0.695742258430131	4	11	0.080502s
				0.695536565102262	15	12	0.163154s
				0.695742258430129	4	11	0.184786s
4	$(1+i, 1+i, 1-i)$	1.468693939915885 $-2.287355287178842i$	11.125681s	1.469314879177899 $-2.286400529446476i$	14	3	0.160053s
				1.468693939915586 $-2.287355287178842i$	15	15	0.281738s
				1.469314879177899 $-2.286400529446476i$	14	3	0.302023s
12	(100,1.5,2.5)	$2.748892975858683 \times 10^{12}$	11.791825s	$2.748923904028464 \times 10^{12}$	4	10	0.112482s
				$2.748892975858687 \times 10^{12}$	15	19	0.262816s
				$2.748923904028461 \times 10^{12}$	4	10	0.306804s
13	$(-60, 1, 10)$	-10.04854112964948	11.359146s	170 terms computed	N/A	N/A	N/A
				-10.04895411296490	14	41	0.260854s
				170 terms computed	N/A	N/A	N/A
15	$(60, 1, -10)$	$-6.713066845459067 \times 10^{-4}$	11.769413s	170 terms computed	N/A	N/A	N/A
				$-6.713066845459049 \times 10^{-4}$	14	37	0.160246s
				2.771191610071790	0	18	0.190572s
19	$(500, 1, -5)$	0.001053895943365	11.891125s	0.001053940354303	7	3	0.132031s
				0.001053895943365	16	24	0.164947s
				$1.905228957818582 \times 10^{24}$	0	9	0.178824s
25	$(-5, (-5+10^{-9})$ $+(-5+10^{-9})i, -1)$	0.507421537454510 $+0.298577267504408i$	11.231796s	0.507423640026765 $+0.298580648127604i$	4	19	0.128643s
				0.50742153745410 $+0.298577267504408i$	15	15	0.283771s
				0.507423641026765 $+0.298580648127604i$	4	7	0.299613s

Table 3: Table showing the MATLAB result and time taken to generate it for a selection of test cases from Appendix B, the results from each of the three Buchholz polynomial methods detailed in this section, their accuracy, the number of terms N computed and computation times. A complete set of results for these methods is shown in Appendix E.

As illustrated by the selection of results in Table 3, method 2 seems to be the most effective out of the three methods described in this section, giving the greatest accuracy. One exception is the case where $b = 2a$, as illustrated by the first row of Table 3, which is due to the division by powers of $\sqrt{z(2b-4a)}$ in (3.18) and (3.20). In such cases however,

we can make use of recurrence relations, as discussed in Section 3.8. Methods 1 and 3 above do not perform well, only giving results of at least 10 digit accuracy for very simple cases, possibly due to the many computations involved in estimating the values of the Buchholz polynomials in (3.19). We therefore focus the rest of the discussion in this section on method 2.

Method 2 is especially valuable for moderate values of a, z ($10 \lesssim |a|, |z| \lesssim 100$, say), where the real parts of a and z have opposite signs. The Taylor series methods discussed in Section 3.2 and the single fraction method of Section 3.3 do not give accurate computations with these cases, but method 2 of this section performs very well as shown in the fifth and sixth rows of Table 3. A very accurate result is even obtained for a large value of $|a|$ ($a = 500$) in the seventh row (case 19), with real z of opposite sign. It should be noted however that the method works substantially less well when $|z|$ is large, as illustrated in Appendix E. Nonetheless, the performance of this method for a large range of parameter values makes this a very worthwhile approach for certain parameter regimes when computing $M(a; b; z)$.

3.5 Asymptotic series

We have found that the methods discussed in Sections 3.2, 3.3 and 3.4 were not at all effective for large values of $|z|$ (typically the methods cease to be effective for $|z| \gtrsim 100$, although this depends on the precise parameter values used). In this Section, we aim to address this issue by introducing the theory of asymptotics for computing the confluent hypergeometric function.

The following expansions for the hypergeometric function $M(a; b; z)$ as $|z| \to \infty$, which can be derived by considering Watson's Lemma [18], are stated in [3]:

$$M(a;b;z) \sim \frac{\Gamma(b)}{\Gamma(a)} e^z z^{a-b} \sum_{j=0}^{\infty} \frac{(b-a)_j (1-a)_j}{j!} z^{-j} \tag{3.23}$$
$$+ \frac{1}{\Gamma(b-a)} e^{\pi i a} z^{-a} \sum_{j=0}^{\infty} \frac{(a)_j (1+a-b)_j}{j!} (-z)^{-j}, \quad -\frac{1}{2}\pi + \delta \leq \arg z \leq \frac{3}{2}\pi - \delta,$$

$$M(a;b;z) \sim \frac{\Gamma(b)}{\Gamma(a)} e^z z^{a-b} \sum_{j=0}^{\infty} \frac{(b-a)_j (1-a)_j}{j!} z^{-j} \tag{3.24}$$
$$+ \frac{1}{\Gamma(b-a)} e^{-\pi i a} z^{-a} \sum_{j=0}^{\infty} \frac{(a)_j (1+a-b)_j}{j!} (-z)^{-j}, \quad -\frac{3}{2}\pi + \delta \leq \arg z \leq \frac{1}{2}\pi - \delta,$$

for $a, b \in \mathbb{C}$, and for an arbitary parameter δ such that $0 < \delta \ll 1$.

We computed these series in MATLAB using the same two techniques as for the Taylor series method in Section 3.2. These are firstly by computing each term using the previous one and summing them until the terms become small, and secondly by finding each term iteratively in terms of the previous two and then summing them. We denote these two techniques for the rest of this section as methods (a) and (b). The results from this are shown in Table 4.

Case	(a,b,z)	Correct $M(a;b;z)$	Time taken	Methods (a) and (b) ($tol = 10^{-15}$)	Acc.	N	Time taken
8	(1,3,10)	$4.403093158961343\times10^{2}$	11.513977s	$4.403093158961344\times10^{2}$	15	2/2	0.128202s
				$4.403093158961344\times10^{2}$	15	2/3	0.147024s
10	(8.1,10.1,100)	$1.724131075992688\times10^{41}$	10.388182s	$1.724131075992683\times10^{41}$	15	10/2	0.139237s
				$1.724131075992683\times10^{41}$	15	11/3	0.195584s
11	(1,2,600)	$6.288367168216566\times10^{257}$	11.892762s	$6.288367168216566\times10^{257}$	16	2/2	0.149720s
				$6.288367168216566\times10^{257}$	16	2/2	0.176241s
18	(10^{-3},1,700)	$1.46135307199289\times10^{298}$	11.761425s	$1.46135307199288\times10^{298}$	15	7/4	0.179830s
				$1.46135307199288\times10^{298}$	15	8/5	0.203889s
26	(4,80,200)	$3.448551506216654\times10^{27}$	12.043978s	$3.448551506216226\times10^{27}$	13	4/34	0.166389s
				$3.448551506216226\times10^{27}$	13	5/35	0.198278s
28	$(5,0.1,-2+300i)$	$7.208553632163922\times10^{10}$ $-1.550289119122414\times10^{10}i$	12.231468s	$7.208553632163907\times10^{10}$ $-1.550289119122399\times10^{10}i$	13	5/13	0.182537s
				$7.208553632163907\times10^{10}$ $-1.550289119122399\times10^{10}i$	13	5/13	0.219843s

Table 4: Table showing the true solution according to Mathematica and MATLAB along with the time taken to compute the solution using MATLAB, the solution obtained by methods (a) and (b) as described in this section, their accuracy, the number of terms required to compute each of the two series of (3.23) or (3.24), and the time taken to do so. Full results are shown in Appendix E.

As with the Taylor series method, there is a considerable similarity between the results obtained using methods (a) and (b). Both methods work well for large z and moderate values of the parameters a and b, meaning in this case b not extremely close to 0 and the real or imaginary parts of a not exceeding roughly 100. For the case where a or b is very close to zero, a variant of the method described in Section 3.3, where each series is written as a single fraction, could be used.

However, as the asymptotic series are expressed in terms of hypergeometric series of the form ${}_2F_0$ instead of ${}_1F_1$, there is no longer a $(b)_j$ term in the denominator of the terms of the series, so that parameter regimes involving b with large modulus can no longer be treated in a straightforward manner. The cases tested suggest that the methods cope reasonably well whether $\mathrm{Re}(a) > \mathrm{Re}(b)$ or $\mathrm{Re}(b) > \mathrm{Re}(a)$, provided neither $|a|$ nor $|b|$ is very large (as

a guide, the computations can become less accurate if $|a|$ or $|b|$ is greater than 50, although this depends on the value of z). For these cases, recurrence relations will need to be applied, as explained in [45, 46], and detailed in Section 3.8.

In [70], it is suggested that the asymptotic relations should be used if $|z| > 30 + |b|$ on the basis of numerical experiments conducted by the authors, and it is suggested in [44], using experimental evidence, that they should be used if $|z| > 50$; it seems in fact that the range in which using the asymptotic approximations is valid can be wider still, as long as the values of $|a|$ or $|b|$ are not large also. If $|a|$ or $|b|$ are large, recurrence relations as described in Section 3.8 can be applied.

3.6 Quadrature methods

So far in this dissertation, all the methods we have considered for computing the confluent hypergeometric function have been based on series methods. In this section, we introduce another class of methods for computing $M(a;b;z)$ using its integral representation for $\mathrm{Re}(b) > \mathrm{Re}(a) > 0$, and discuss its effectiveness. Other methods for computing this integral are discussed in Appendix G.

As stated in [3], the function $M(a;b;z)$ has the following integral representation:

$$M(a;b;z) = \frac{\Gamma(b)}{\Gamma(a)\Gamma(b-a)} \int_0^1 e^{zt} w_{a,b}(t) \mathrm{d}t, \quad \mathrm{Re}(b) > \mathrm{Re}(a) > 0, \tag{3.25}$$

where

$$w_{a,b}(t) = (1-t)^{b-a-1} t^{a-1}.$$

Applying the transformation $t \mapsto \frac{1}{2}\tilde{t} + \frac{1}{2}$ and using Jacobi parameters $\tilde{\alpha} = b-a-1$, $\tilde{\beta} = a-1$, as in [26], we find that

$$\begin{aligned} \int_0^1 e^{zt} w_{a,b}(t) \mathrm{d}t &= \frac{1}{2^{b-1}} \int_{-1}^1 e^{z\left(\frac{1}{2}\tilde{t}+\frac{1}{2}\right)} (1-\tilde{t})^{b-a-1} (1+\tilde{t})^{a-1} \mathrm{d}\tilde{t} \\ &= \frac{e^{z/2}}{2^{b-1}} \sum_{j=1}^{N_{mesh}} w_j^{GJ} e^{z t_j^{GJ}/2} + E_{N_{mesh}}(a;b;z), \end{aligned}$$

where t_j^{GJ} and w_j^{GJ} are the Gauss-Jacobi nodes and weights on $[-1,1]$. In [59], t_j^{GJ} are defined as the roots of the j-th **Jacobi polynomial**,

$$P_j^{(\tilde{\alpha},\tilde{\beta})}(z) = \frac{\Gamma(\tilde{\alpha}+j+1)}{j!\Gamma(\tilde{\alpha}+\tilde{\beta}+j+1)} \sum_{k=0}^{j} \binom{j}{k} \frac{\Gamma(\tilde{\alpha}+\tilde{\beta}+j+k+1)}{\Gamma(\tilde{\alpha}+k+1)} \left(\frac{z-1}{2}\right)^k, \quad j = 1, 2, ..., N_{mesh},$$

and w_j^{GJ} are defined as

$$w_j^{GJ} = \frac{2^{\tilde{\alpha}+\tilde{\beta}+1}\Gamma(\tilde{\alpha}+N_{mesh}+1)\Gamma(\tilde{\beta}+N_{mesh}+1)}{\Gamma(N_{mesh}+1)\Gamma(\tilde{\alpha}+\tilde{\beta}+N_{mesh}+1)(1-x_j^{GJ})^2[P_{N_{mesh}}^{(\tilde{\alpha},\tilde{\beta})}]^2}, \quad j=1,2,...,N_{mesh}.$$

This method is known as **Gauss-Jacobi quadrature**. In theory, the error for this method $E_{N_{mesh}}$ can be controlled by the number of mesh points N_{mesh}. This is shown for real a and b using a result shown in [26]:

$$N_{mesh} \geq \frac{e\,|z|}{8} \times t\left(\frac{4}{e\,|z|}\left[x_+ + (3-2b)\log 2 + \log\left(\frac{1}{E_{N_{mesh}}}\right)\right]\right), \; x_+ = \begin{cases} x & \text{if } x \geq 0, \\ 0 & \text{if } x < 0, \end{cases} \quad (3.26)$$

where $x = \mathrm{Re}(z)$. Here, t denotes the inverse of the function $s = t\log t$. Low-order approximations are stated in [25] for different real values of s. For example,

$$t(s) \approx \frac{1}{e} + \frac{e-1}{\sqrt{e}}\left(x+\frac{1}{e}\right)^{\frac{1}{2}}, \quad -\frac{1}{e} \leq s \leq 0; \qquad t(s) \approx \frac{s}{\log s\left(1-\frac{\log\log s}{1+\log s}\right)}, \; s \geq 2. \quad (3.27)$$

This should give an idea as to the required number of points to generate a specified accuracy. For example, if $b=-15$, $z=100$ and the error E_N is desired to be less than 10^{-10}, then $s = \frac{4}{e|z|}\left[x_+ + (3-2b)\log 2 + \log\left(\frac{1}{E_N}\right)\right] \approx 2.14694$, $t(s) \approx 2.43802$ using (3.27), so using (3.26), we deduce that roughly 83 points are desired to generate the required accuracy. The routines used to carry out Gauss-Jacobi quadrature were `gaussq.m` and `qrule.m` from [72], and were obtained from the MathWorks website.

Case	(a,b,z)	Correct $M(a;b;z)$	Time taken	Gauss-Jacobi ($N_{mesh}=200$)	Acc.	N_{crit}	Time taken
1	(0.1,0.2,0.5)	1.317627178278510	11.164293s	1.317627178278510	16	10	0.299203s
8	(1,3,10)	$4.403093158961343 \times 10^2$	11.513977s	$4.403093158961341 \times 10^2$	15	10	0.314343s
10	(8.1,10.1,100)	$1.724131075992688 \times 10^{41}$	10.388182s	$1.724131075992687 \times 10^{41}$	15	30	0.316557s
11	(1,2,600)	$6.288367168216566 \times 10^{257}$	11.892762s	$6.288367168215225 \times 10^{257}$	12	70	0.392941s
18	$(10^{-3},1,700)$	$1.461353307199289 \times 10^{298}$	11.761425s	$1.461353307199045 \times 10^{298}$	13	70	0.422357s
26	(4,80,200)	$3.448551506216654 \times 10^{27}$	12.043978s	$6.470060431231330 \times 10^{28}$	0	N/A	N/A

Table 5: Table showing the true value of $M(a;b;z)$ for a selection of test cases and the time taken to compute this using 'hypergeom'. Also shown is the result obtained using Gauss-Jacobi quadrature with 200 mesh points, the critical number of mesh points N_{crit} required to obtain 10 digit accuracy (where N_{mesh} is increased by increments of 10 until 10 digit accuracy is obtained), and the computation time with N_{crit} mesh points. When 'N/A' is written, 5000 mesh points were not sufficient to give 10 digit accuracy. Full results are given in Appendix E.

Gauss-Jacobi quadrature is a natural choice due to the form of the integrand in (3.25) and the fact that the integrand blows up at the end-points of the integral. As illustrated by Table 5 and Appendix E, we find that the method of Gauss-Jacobi quadrature deals with most values of $|z|$ (small or large), provided z does not have an imaginary part with magnitude greater than roughly 100. The third, fourth and fifth rows of Table 5 illustrate this, but as shown by the sixth row (case 26), a problem arises when either $|a|$ or $|b|$ becomes fairly large. The number of mesh points required to generate 10 digit accuracy for the cases above seems to correspond to the number of mesh points predicted by (3.26).

For small values of $|a|$ and $|b|$ (usually up to 30–40), the method of Gauss-Jacobi quadrature is extremely useful for evaluating the confluent hypergeometric function when $\mathrm{Re}(b) > \mathrm{Re}(a) > 0$, and should play a part in any package for this reason.

Other methods implemented for computing the integral (3.25) are detailed in Appendix G.5.

3.7 Solving the confluent hypergeometric differential equation

Another class of methods for computing $M(a;b;z)$ is based on solving the differential equation (3.2). We wish to explore the effectiveness of computations involving the use of the RK4 method, a 4th order accurate **Runge-Kutta method**. Further methods for solving the problem using (3.2) are discussed in Appendix G.6.

As stated in [3], a fundamental pair of solutions of (3.2) near the origin is

$$M(a;b;z),\ z^{1-b}M(a-b+1;2-b;z).$$

We note that $U(a;b;z)$ is a linear combination of these two solutions, and that the second solution is only valid if $b \notin \mathbb{Z}$. Solutions for the case $b \in \mathbb{Z}$ are discussed in [22], but the solutions do not take the form of a standard hypergeometric function as discussed, which renders the differential equation method less suitable in this case.

As noted in [51], a fundamental pair of solutions of (3.2) in the neighbourhood of infinity is

$$U(a;b;z),\ e^{z}U(b-a;b;e^{-\pi i}z),\ \ -\frac{1}{2}\pi \leq \arg z < \frac{3}{2}\pi.$$

We now consider whether it is more efficient and accurate to solve an initial value problem or a boundary value problem. By examining and differentiating the Taylor series for $M(a;b;z)$ in (3.1), we can see that the following two initial conditions can be used:

$$w(0) = 1,\ w'(0) = \frac{a}{b}, \tag{3.28}$$

and we can integrate along outward rays from the origin to the value of z where the computation is desired to generate an approximation of $M(a;b;z)$. Although solving a boundary value problem is in general more accurate (methods for solving boundary value problems are detailed in [73]), it can only be done in a region between the origin and another point where the hypergeometric function has already been calculated. As we are not necessarily able to calculate $M(a;b;z)$ at another point, we focus for the remainder of this section on solving the initial value problem satisfied by $M(a;b;z)$.

We found that the RK4 method was the most effective way of solving the differential equation out of those tested. We recall that for $k = 0, 1, 2, ...$, the RK4 method is defined as

$$z_{k+1} = z_k + h, \tag{3.29}$$

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \frac{1}{6}h(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4), \tag{3.30}$$

where

$$\begin{aligned}
\mathbf{k}_1 &= \mathbf{f}(z_k, \mathbf{w}_k), \\
\mathbf{k}_2 &= \mathbf{f}\left(z_k + \frac{1}{2}h, \mathbf{w}_k + \frac{1}{2}h\mathbf{k}_1\right), \\
\mathbf{k}_3 &= \mathbf{f}\left(z_k + \frac{1}{2}h, \mathbf{w}_k + \frac{1}{2}h\mathbf{k}_2\right), \\
\mathbf{k}_4 &= \mathbf{f}(z_k + h, \mathbf{w}_k + h\mathbf{k}_3),
\end{aligned}$$

with z_0 taken to be 0 so that the initial conditions can be applied, $\mathbf{w} = (w_1, w_2)^T = (w, w')^T$, $\mathbf{w}_0 = (1, \frac{a}{b})^T$, and $\mathbf{f}(z) = (w_2, -\frac{1}{z}\{(b-z)w_2 - aw_1\})^T$.

We tested the RK4 method, along with the Dormand-Prince method which is detailed in Appendix G.6. Also to provide a better insight into the differential equation method, we tested three built-in MATLAB solvers: '**ode45**', a method that is most suitable for non-stiff problems and generates medium accuracy; '**ode113**', which is another non-stiff solver

generating low to high accuracy; and '**ode15s**', a stiff solver that generates low-to-medium accuracy. The disadvantage of using these three methods is that MATLAB will not generate a solution to the differential equation (3.2) with initial conditions (3.28) because there is a singular point where the initial conditions are located. Instead, we integrated (3.2) from $z = 10^{-3}$ for all cases tested, (as shown in Table 6), except for case 5, which we integrated from $z = 10^{-15}$, and case 17, which we integrated from $z = 10^{-8}$. The settings 'RelTol' and 'AbsTol' were both taken to be 10^{-15}.

Case	(a,b,z)	RK4 ($N_{mesh} = 500$)	Acc.	N_{crit}	Time taken	ode45/ode113/ode15s	Acc.	N
1	(0.1,0.2,0.5)	1.317627178272184	12	100	0.461925s	1.317627126236872	8	41
						1.317627885961121	7	13
						1.318321919289360	3	12
2	$(-0.1, 0.2, 0.5)$	0.695536565117007	10	250	0.679498s	0.695536686554347	6	41
						0.695536602807524	6	13
						0.696313392446649	2	12
5	$(10^{-8}, 10^{-8}, 10^{-10})$	1.000000000100030	14	50	0.433277s	1.000000000100001	15	41
						1.000000000100001	15	11
						1.000000000100001	15	11
10	(8.1,10.1,100)	$1.722537193851716 \times 10^{41}$	3	18650	16.459445s	$1.725041594930256 \times 10^{41}$	3	393
						$1.711843732591705 \times 10^{41}$	2	158
						$1.285670097530191 \times 10^{41}$	1	217
17	$(1000, 1, 10^{-3})$	2.279929853883460	11	350	0.701499s	2.279933221635035	5	41
						2.279969898504121	5	15
						2.361501968192591	1	14
22	$(20, 10 - 10^{-9}, 2.5)$	98.353133200849229	8	1500	2.231877s	98.354028221444111	4	97
						98.290032237005732	2	40
						$1.28232933799147 \times 10^{2}$	0	22
36	$(1, -1 + 10^{-12}i, 1)$	$-5.5333556053310802 \times 10^{-1}$	2	N/A	N/A	$-5.537120346107741 \times 10^{-1}$	2	69
		$+2.718218662119980 \times 10^{12}i$				$+2.718129726676962 \times 10^{12}i$		
						$-5.531273187639527 \times 10^{-1}$	2	26
						$+2.718163893880989 \times 10^{12}i$		
						-1.258764325129202	0	33
						$+2.697519329272323 \times 10^{12}i$		

Table 6: Table showing the true value of $M(a; b; z)$ for a selection of test cases, the result obtained using the RK4 method with 500 mesh points, the critical number of mesh points N_{crit} required to obtain 10 digit accuracy (increasing the N_{mesh} by increments of 50 until 10 digit accuracy was obtained), and the computation time with N_{crit} mesh points; here 'N/A' is written when 20000 mesh points were not sufficient to give 10 digit accuracy. Also stated are the results using 'ode45', 'ode113' and 'ode15s', their accuracy and the number of points N that MATLAB uses to produce the solution vector. Full results are detailed in Appendix E.

Table 6 illustrates that although the RK4 method generates fairly accurate results when $|z|$ is sufficiently close to zero (less than about 5 when 500 mesh points are used, although this depends on the precise values of $|a|$ and $|b|$), the method struggles greatly when $|z|$ is far away from zero, as illustrated powerfully by the fourth row of the table (case 10). It seems

that even the built-in MATLAB solvers struggle to solve this problem, possibly due to the fact that we instructed the solvers to start the numerical integration from a point close to the singular point at $z = 0$. Therefore, we conclude that the differential equation method does not work as well as a number of others previously discussed for computing ${}_1F_1$, due to its poor performance when $|z|$ is far away from 0.

3.8 Recurrence relations

Frequently, the robustness of a method for computing the confluent hypergeometric function is greatly reduced by its poor performance as $|\mathrm{Re}(a)|$ or $|\mathrm{Re}(b)|$ gets larger. This section details the recurrence relation techniques, which can reduce the problem of computation with these large parameter values to a simpler problem of computing $M(a; b; z)$ with values of $\mathrm{Re}(a)$ and $\mathrm{Re}(b)$ whose modulus is much closer to 0. Another method can then be applied to solve the simpler problem, usually with much greater success, as our results so far have shown.

It is known from [30] that the function $M(a; b; z)$ satisfies the following recurrence relations:

$$M(a+n;b;z) - \frac{2n+2a+z-b}{n+a}M(a;b;z) + \frac{a+n-b}{n+a}M(a-n;b;z) = 0, \quad (3.31)$$

$$M(a;b+n;z) + \left(\frac{1-b-n}{z} - 1\right)M(a;b;z) + \frac{b+n-a-1}{z}M(a;b-n;z) = 0, \quad (3.32)$$

$$M(a+n;b+n;z) + \frac{b+n-z-1}{(a+n)z}M(a;b;z) - \frac{1}{(a+n)z}M(a-n;b-n;z) = 0. \quad (3.33)$$

A solution f_n of a recurrence relation

$$y_{n+1} + b_n y_n + a_n y_{n-1} = 0 \quad (3.34)$$

is said to be a **minimal solution** if there is a linearly independent solution g_n (called a **dominant solution**) such that

$$\lim_{n\to\infty} \frac{f_n}{g_n} = 0.$$

We use the following theorem, discussed in [29, 30, 62], in our subsequent investigation of recurrence relations.

Poincaré's Theorem: Consider the recurrence relation (3.34), where $\lim_{b\to+\infty} b_n = b_\infty$ and $\lim_{n\to+\infty} a_n = a_\infty$. Denote the zeros of the equation $t^2 + b_\infty t + a_\infty = 0$ by t_1 and t_2. Then if $|t_1| \neq |t_2|$, the recurrence relation (3.34) has two linearly independent solutions f_n, g_n such that:

$$\lim_{n\to+\infty} \frac{f_n}{f_{n-1}} = t_1, \quad \lim_{n\to+\infty} \frac{g_n}{g_{n-1}} = t_2,$$

and if $|t_1| = |t_2|$, then

$$\limsup_{n\to+\infty} |y_n|^{1/n} = |t_1|$$

for any non-trivial solution y_n of (3.34).

Further, when $|t_1| \neq |t_2|$, the solution whose ratio of consecutive terms tends to the root of smallest modulus is always the minimal solution.

Now, we consider the three recurrence relations (3.31)–(3.33), discussed in [29, 62], and denoted as (+0), (0+) and (++) respectively for the rest of this section. Stated below are the two solutions of (3.31), (3.32) and (3.33) respectively, along with known relations as $n \to +\infty$, as discussed in [30]:

$$f_n = \Gamma(1+a+n-b)U(a+n;b;z),\ g_n = M(a+n;b;z),\ \frac{f_n}{g_n} \sim e^{-4\sqrt{nz}}, \tag{3.35}$$

$$f_n = \frac{\Gamma(b+n-a)M(a;b+n;z)}{\Gamma(b+n)},\ g_n = U(a;b+n;z),\ \frac{f_n}{f_{n-1}} \sim 1,\ \frac{g_{n+1}}{g_n} \sim \frac{n}{z}, \tag{3.36}$$

$$f_n = \frac{M(a+n;b+n;z)}{\Gamma(b+n)},\ g_n = (-1)^n U(a+n;b+n;z),\ \frac{f_n}{f_{n-1}} \sim \frac{1}{n},\ \frac{g_n}{g_{n-1}} \sim -\frac{1}{z}. \tag{3.37}$$

Hence, using (3.35)–(3.37), the definition of a minimal solution, and Poincaré's Theorem, one can deduce that the minimal solutions of the above three relations (+0), (0+) and (++) are, respectively,

$$\Gamma(1+a+n-b)U(a+n;b;z),\ \frac{\Gamma(b+n-a)M(a;b+n;z)}{\Gamma(b+n)},\ \frac{M(a+n;b+n;z)}{\Gamma(b+n)}. \tag{3.38}$$

As we are considering the computation of $M(a;b;z)$, we will consider recurrence relations (0+) and (++) for the remainder of this section.

Suppose we seek the solution of the general three term recurrence relation (3.34). The following algorithm, called **Miller's algorithm** [30], aims to compute numerical approximations $\tilde{f}_n,\ n = 0, ..., k$ to f_n, the minimal solution of (3.34). The values that need to be specified are some tolerance tol, an initial value f_0, and a number k.

Miller's Algorithm: Choose a value $N \gg k$ such that:

$$\left| \frac{y_k/y_{k-1}}{f_k/f_{k-1}} - 1 \right| < tol$$

$$y_N = 1,\ y_{N-1} = 0$$

$$\text{for } n = N-1, ..., 1$$

$$y_{n-1} = -\frac{1}{a_n}(y_{n+1} + b_n y_n)$$

$$\text{for } n = 0, ..., k$$

$$\tilde{f}_n = \frac{f_0}{y_0} y_n.$$

As explained in [30], the motivation for Miller's algorithm is that if we choose $N \gg k$ then the ratio $\frac{y_k}{y_{k-1}}$ will approach $\frac{f_k}{f_{k-1}}$ due to the minimality of f_n, so $\tilde{f}_n = \frac{f_0}{y_0} y_n$ represents a good approximation to f_n.

We therefore have two methods that we can potentially exploit: firstly, we can take the minimal solution to (0+) or (++) and apply the recurrence relations backwards; secondly, we can start with the minimal solutions and apply the recurrence relations forwards using Miller's algorithm.

Table 7 illustrates that the technique of using recurrence relations can be used to effectively compute test cases on which previous methods have not performed well. The ideas introduced in this section can be extended to compute recurrence relations with large $|\mathrm{Re}(a)|$. For instance, if we wish to compute $M(100.2; 0.1; 1)$, we could compute $M(0.2; 0.1; 1)$ and $M(0.2; -0.9; 1)$ using methods discussed in Sections 3.2 and 3.3, apply Miller's algorithm with $k = 100$ on (++) to both of these to obtain $M(100.2; 100.1; 1)$ and $M(100.2; 99.1; 1)$ respectively, and then apply backward recursion of (0+) using these two results to compute $(100.2; 0.1; 1)$.

It should be noted however that due to the fact that the minimal solutions (3.38) involve Gamma functions, the effectiveness of this method is restricted by the fact that MATLAB is unable to handle the Gamma function of a variable with large modulus (for example,

Function desired	Method and function(s) used	Correct solution	Recurrence solution	Acc.
$M(0.3;-79.3;2.5)$	Backward recursion of (0+) $M(0.3;0.7;2.5)/M(0.3;-0.3;2.5)$	0.990733787354197	0.990733787354306	12
$M(0.9;-119.8;-5)$	Backward recursion of (0+) $M(0.9;0.2;-5)/M(0.9;-0.8;-5)$	1.039128783613421	1.039128783613467	15
$M(-149.5;149.2;6)$	Backward recursion of (++) $M(0.5;0.8;6)/M(-0.5;-0.2;6)$	$4.084281216374062 \times 10^2$	$4.084281216374482 \times 10^2$	13
$M(-44.25;-44.7;-1)$	Backward recursion of (++) $M(0.75;0.3;-1)/M(-0.25;-0.7;-1)$	0.371559897558854	0.371559897558854	16
$M(0.8;65.7;4)$	Forward recursion of (0+) (M) $M(0.8;0.7;4)$	1.051488516006696	1.051488516006697	15
$M(0.85;90.5;5.5)$	Forward recursion of (0+) (M) $M(0.85;0.5;5.5)$	1.054701645960524	1.054701645960513	14
$M(70.3;70.9;1)$	Forward recursion of (++) (M) $M(0.3;0.9;1)$	2.695530979957562	2.695530979957563	15
$M(90.4;90.9;-30.25)$	Forward recursion of (++) (M) $M(0.4;0.9;-30.25)$	$8.913072834489234 \times 10^{-14}$	$8.913072834488962 \times 10^{-14}$	12

Table 7: Table showing a variety of examples of the application of the recurrence relation techniques of this section. Shown is the function we wish to compute, the easily computable functions we need to compute to generate the solution, the method used to obtain the solution from these results, the solution we obtain, the actual solution, and the number of digits accuracy we obtain by using our method. The designation (M) in the second column denotes that Miller's algorithm was used.

$\Gamma(171)$ is finite according to MATLAB but $\Gamma(172)$ is infinite). It is therefore ideal to apply the technique of using recurrence relations to software which can compute Gamma functions with variable of larger modulus.

3.9 Summary and analysis of results

For ${}_1F_1(a;b;z)$, the methods we have implemented and analysed have included series methods as in Sections 3.2, 3.3, 3.4 and 3.5, as well as the use of quadrature (Section 3.6), numerical solution of differential equations (Section 3.7) and recurrence relations (Section 3.8), along with other, less effective methods that we detail in Appendix G.

For the most part, the series methods analysed seemed to generate the most accurate results, and with very fast computation times in comparison to the built-in MATLAB function 'hypergeom'. For values of $|a|$ and $|z|$ less than around 50 and $|b|$ not too close to zero, the Taylor series methods described in Section 3.2, and the method of expressing ${}_1F_1$ as a single fraction as in Section 3.3 seem to be sufficiently robust (although when $|b| < 1$ we recommend that only the latter be used). In other instances, we note that for all cases tested, whenever the Taylor series methods and the single fraction method generated the same answer to 10

or more digits, both methods generated the correct answer, and conversely whenever the two solutions were different, they were both incorrect. This suggests that, as these two methods are of the same 'family' (they both compute the power series expression of ${}_1F_1(a;b;z)$ but in different ways), it would be useful to apply both methods for this parameter regime, so that each might check the validity of the solution generated by the other.

We also found that, when $\mathrm{Re}(b) > \mathrm{Re}(a) > 0$, the method of Gauss-Jacobi quadrature was successful, providing the values of $|a|$, $|b|$ were less than about 30. For ${}_1F_1$ at least, we found the method of solving the differential equation (3.2) to be ineffective, due to its poor performance for large $|z|$. If $|\mathrm{Re}(a)|$ or $|\mathrm{Re}(b)| > 50$, we can apply the ideas of recurrence relations detailed in Section 3.8 to reduce the problem to one of computing hypergeometric functions with parameter values that have real parts of smaller absolute value.

One parameter regime in which both these classes of methods fail when we would expect them to succeed is when $|a|$ and $|z|$ are roughly between 10 and 100 with their real parts of opposite signs, in which case we recommend the use of the method involving Buchholz polynomials discussed in Section 3.4. We can deal with another important case, that of large $|z|$, by applying the asymptotic expansions of Section 3.5. Expansions with exponentially-improved accuracy, known as **hyperasymptotic expansions**, are detailed in Appendix G.4. As MATLAB is unable to compute the incomplete gamma function $\Gamma(\varpi, z)$ for complex or negative real parameter ϖ, we were unable to examine the simplest such expansion, which is detailed in [52, 53]. However, if software with high precision and programs that could compute the incomplete gamma function were available, we conclude from the literature that the use of hyperasymptotic expansions could be a viable alternative for the computation of ${}_1F_1$ for large $|z|$.

To provide a guide to the most effective methods we investigated, a list of recommendations is shown in Table 8.

Two cases where we found that none of the methods we tried generated 10 digit accuracy reliably were the cases of large $\mathrm{Im}(z)$ and the cases where the parameters a and b have large imaginary parts. The latter is a problem because unlike for the cases of large $|\mathrm{Re}(a)|$ or $|\mathrm{Re}(b)|$, the recurrence relation ideas of Section 3.8 cannot be applied. A major element of future work on this subject area could involve finding more effective methods for these parameter and variable regimes, as discussed in Section 5.

Regions for a, b, z	Recommended method(s)	Relevant sections
$\lvert a\rvert, \lvert z\rvert < 50$, $\lvert b\rvert > 1$, sign(Re(a))=sign(Re(z))	Taylor series methods Single fraction method Gauss-Jacobi quadrature, if Re(b) > Re(a) > 0, $\lvert a\rvert, \lvert b\rvert \lesssim 30$	3.2 3.3 3.6
$\lvert a\rvert < 20$, $\lvert z\rvert < 50$, $\lvert b\rvert > 1$, sign(Re(a))=sign(Re(z))	Taylor series methods Single fraction method	3.2 3.3
$\lvert a\rvert, \lvert z\rvert < 30$, $\lvert b\rvert < 1$	Single fraction method	3.3
$\lvert a\rvert, \lvert z\rvert < 50$, $\lvert b\rvert < 50$, sign(Re(a))=$-$sign(Re(z))	Buchholz polynomial method 2	3.4
$\lvert z\rvert > 100$, $\lvert a\rvert, \lvert b\rvert < 50$	Asymptotic expansions Hyperasymptotic expansions	3.5 Appendix G.4
$\lvert z\rvert > 100$, $\lvert a\rvert > 50$ or $\lvert b\rvert > 50$	Recurrence relations then asymptotic expansions	3.8 3.5
$\lvert$Re(a)$\rvert$ or $\lvert$Re(b)$\rvert > 50$	Recurrence relations then another method	3.8

Table 8: Recommendations as to methods that should be used for computation of the confluent hypergeometric function for different parameter and variable regimes, and the sections where they are discussed. Here, the 'sign' function is defined to be 1 if the (real) argument is greater than 0, -1 if the argument is less than 0, and 0 if the argument is equal to 0.

4 Computation of the Gauss hypergeometric function ${}_2F_1(a, b; c; z)$

In this section, we discuss the best methods we found to compute the Gauss hypergeometric function ${}_2F_1(a, b; c; z)$ accurately and quickly, before providing recommendations as to the most effective methods for each parameter and variable regime. We implement ideas of similar form to those used for the confluent hypergeometric function, such as those in Sections 3.2, 3.3, 3.6 and 3.7, as well as methods that are only applicable to ${}_2F_1$. Other methods that we have implemented or analysed are shown in Appendix H.

4.1 Properties of ${}_2F_1$

The **Gauss hypergeometric function** ${}_2F_1(a, b; c; z)$ is defined as the series

$$ {}_2F_1(a, b; c; z) = \sum_{j=0}^{\infty} \frac{(a)_j (b)_j}{(c)_j} \frac{z^j}{j!}, \tag{4.1} $$

when z is in the radius of convergence of the series $|z| < 1$, which we deduce from the theory in Section 2.1. This series is defined for any $a \in \mathbb{C}$, $b \in \mathbb{C}$, $c \in \mathbb{C}\backslash\{\mathbb{Z}^- \cup \{0\}\}$. For z outside

this range, ${}_2F_1(a,b;c;z)$ is defined by analytic continuation formulae, as detailed in Section 4.7. This will allow us to consider the computation of ${}_2F_1$ for any $z \in \mathbb{C}$.

It should be noted that ${}_2F_1(a,b;c;0) = 1$ for any a, b and $c \notin \mathbb{Z}^- \cup \{0\}$. If $c = n$, $n \in \mathbb{Z}^- \cup \{0\}$, then this series is given by a polynomial of degree $-n$ in z. As noted in [3], on the unit disc $|z| = 1$, the series in (4.1) converges absolutely when $\mathrm{Re}(c-a-b) > 0$ (converging to the value $\frac{\Gamma(c)\Gamma(c-a-b)}{\Gamma(c-a)\Gamma(c-b)}$ at $z = 1$, as stated in [58]), converges conditionally when $-1 < \mathrm{Re}(c-a-b) \leq 0$ apart from at $z = 1$, and does not converge if $\mathrm{Re}(c-a-b) \leq -1$.

As explained in [3], the Gauss hypergeometric function satisfies the differential equation

$$z(1-z)\frac{\mathrm{d}^2 w}{\mathrm{d}z^2} + [c-(a+b+1)z]\frac{\mathrm{d}w}{\mathrm{d}z} - abw = 0 \tag{4.2}$$

whenever none of a, b or c differ pairwise by an integer, and when $c \in \mathbb{Z}^- \cup \{0\}$. When $c \in \mathbb{Z}^- \cup \{0\}$, the case is resolved by the fact that the following is a solution for $|z| < 1$ and any $c \in \mathbb{C}$:

$$\mathbf{F}(a,b;c;z) = \sum_{j=0}^{\infty} \frac{(a)_j (b)_j}{\Gamma(c+j)} \frac{z^j}{j!} = \frac{{}_2F_1(a,b;c;z)}{\Gamma(c)}. \tag{4.3}$$

The differential equation (4.2) has three singular points: $z = 0$, $z = 1$ and $z = \infty$. Other solutions near these singular points are discussed in Section 4.5.

As discussed in [51], there is a branch cut between $z = 1$ and $z = +\infty$; the branch in the sector $|\arg(1-z)| < \pi$ is defined as the **principle branch**, and we shall aim to compute ${}_2F_1$ in the principle branch.

As noted in [51], if z is replaced by $\frac{z}{b}$ with $|b| \to \infty$, and c is replaced by b, then we obtain the confluent hypergeometric differential equation (3.2). Consequently, as noted in [61],

$${}_1F_1(a;c;z) = \lim_{|b|\to\infty} {}_2F_1\left(a,b;c;\frac{z}{b}\right),$$

so that for large $|b|$, the Gauss hypergeometric function could in theory be computed using methods for computing the confluent hypergeometric function.

It is known that ${}_2F_1(a,b;c;z)$ satisfies the following recurrence relations [29]:

$$\begin{aligned}(c-a-n){}_2F_1(a-n,b;c;z)+(2a-c+2n+(b-a)z){}_2F_1(a,b;c;z)\\ +(a+n)(z-1){}_2F_1(a+n,b;c;z)=0,\\ (c+n)(c+n-1)(z-1){}_2F_1(a,b;c-n;z)\\ +(c+n)(c+n-1-(2c-a-b+2n-1)z){}_2F_1(a,b;c;z)\\ +(c-a+n)(c-b+n)z\ {}_2F_1(a,b;c+n;z)=0,\end{aligned}$$

which can prove useful in this project, as detailed in Section 4.8.

The following transformation, stated in [3], is also useful when certain values of a, b, c prove difficult computationally:

$${}_2F_1(a,b;c;z)=(1-z)^{c-a-b}\ {}_2F_1(c-a,c-b;c;z). \tag{4.4}$$

This can provide a useful transformation of parameters for computational purposes.

4.2 Taylor series

In this section, we aim to carry out computations of the Gauss hypergeometric function using its basic Taylor series representation (4.1) and analyse their accuracy for different parameter regimes.

As with ${}_1F_1(a;b;z)$ in Section 3.2, we used two methods to compute the power series of ${}_2F_1(a,b;c;z)$ itself,

$${}_2F_1(a,b;c;z)=\sum_{j=0}^{\infty}\underbrace{\frac{(a)_j(b)_j}{(c)_j}\frac{1}{j!}}_{C_j}z^j. \tag{4.5}$$

Method (a): We compute

$$\begin{gathered}C_0=1,\quad S_0=C_0,\\ C_{j+1}=C_j\times\frac{(a+j)(b+j)}{c+j}\times\frac{z}{j+1},\quad S_{j+1}=S_j+C_{j+1},\ j=0,1,2,\dots\ ,\end{gathered}$$

where C_j denotes the $(j+1)$-st term of the Taylor series (4.1) and S_j denotes the sum of the first $j+1$ terms.

We stop the summation when $\frac{|C_{N+1}|}{|S_N|} < tol$, $\frac{|C_N|}{|S_{N-1}|} < tol$ and $\frac{|C_{N-1}|}{|S_{N-2}|} < tol$ for some tol and some N, and S_N is returned as the solution. The reason for the more stringent stopping criterion than that discussed in Section 3.2 for computing ${}_1F_1(a;b;z)$ is that if both a and b are close to negative integers $-m$ and $-m+1$ say, then the $(m+1)$-st and $(m+2)$-nd terms are likely to be very small compared to the size of the previous terms, but the subsequent terms might make a substantial contribution to the solution. Therefore a stopping criterion should involve three terms having a relatively small modulus rather than two as for ${}_1F_1$.

Method (b): Similarly to the recommended method of [44] discussed in Section 3.2, a recurrence relation deducing the next approximation in terms of the previous two can be computed as follows:

$$\begin{aligned} S_{-1} = S_0 = 1, \quad S_1 &= \frac{ab}{c}z, \\ r_j = \frac{(a+j-1)(b+j-1)}{j(c+j-1)}, \quad j &= 2,3,\dots, \\ S_j = S_{j-1} + (S_{j-1} - S_{j-2})r_j z, \quad j &= 2,3,\dots. \end{aligned}$$

The summation is stopped when $\frac{|S_{N+1}-S_N|}{|S_N|} < tol$, $\frac{|S_N-S_{N-1}|}{|S_{N-1}|} < tol$, and $\frac{|S_{N-1}-S_{N-2}|}{|S_{N-2}|} < tol$ for some tol and some N, and S_N is returned as the solution.

Methods (a) and (b) are both equivalent to truncating the series

$$S_\infty = \sum_{j=0}^{\infty} \frac{(a)_j (b)_j}{(c)_j} \frac{z^j}{j!}. \tag{4.6}$$

As shown in Table 9, Taylor series methods (a) and (b) work very similarly for computing ${}_2F_1(a,b;c;z)$ in terms of accuracy and number of terms required for computation. Both methods work very successfully for cases with small magnitudes of parameter values, such as rows 1–4 in Table 9. The time taken is substantially less than the time taken by the built-in MATLAB program, but the larger number of terms that need to be computed before the stopping criterion can be applied relative to similar cases tested on Taylor series methods for computing ${}_1F_1$ (such as those in rows 1 and 3 in Table 1) illustrate that computing ${}_2F_1$ is a much more difficult problem in this regard. The number of points required for computation for three test cases and real $z \in [-1,1]$ is shown in Figure 3, illustrating that many more points are required for computation the closer z is to the unit disc.

Case	(a,b,c,z)	Correct ${}_2F_1(a,b;c;z)$	Time	Method (a) and (b)	Acc.	N	Time
1	(0.1,0.2,0.3,0.5)	1.046432811217352	10.471900s	1.046432811217352	16	41	0.034263s
				1.046432811217351	15	41	0.055681s
4	$(10^{-8},10^{-8},10^{-8},10^{-6})$	1.000000000000010	9.971552s	1.000000000000010	16	3	0.041208s
				1.000000000000010	16	3	0.047378s
8	$(2+8i,3-5i,\sqrt{2}-\pi i,0.75)$	$6.882463762011614 \times 10^3$	10.771072s	$6.882463762011581 \times 10^3$	13	163	0.066219s
		$-6.596555778724484 \times 10^3 i$		$-6.596555778724495 \times 10^3 i$			
				$6.882463762011582 \times 10^3$	13	163	0.079952s
				$-6.596555778724483 \times 10^3 i$			
12	$(-1,-1.5,-2-10^{-15},0.5)$	0.625000000000000*	10.997514s	0.625000000000000	16	2	0.018291s
				0.625000000000000	16	2	0.046936s
15	$(-1000,-2000,-4000.1,-0.5)$	$5.233580403196932 \times 10^{94}$	13.034198s	$5.233580403196953 \times 10^{94}$	14	293	0.067488s
				$5.233580403196921 \times 10^{94}$	14	293	0.068359s
18	(5,−300,10,0.5)	$1.661006238211309 \times 10^{-7}$	11.922410s	$4.628142177960427 \times 10^{29}$	0	199	0.071125s
				$3.698084043503173 \times 10^{28}$	0	201	0.109863s
20	$(2+200i,5,10,0.6)$	$1.499739394713933 \times 10^{-7}$	11.418625s	$-8.206946157063342 \times 10^{31}$	0	399	0.086884s
		$+5.771450716812297 \times 10^{-7} i$		$+8.961768586845500 \times 10^{31} i$			
				$3.168834584274869 \times 10^{32}$	0	396	0.152469s
				$+5.250473854631711 \times 10^{31} i$			
21	$(2+200i,5-100i,$	-4.103442641430799	11.226622s	-4.102898902166944	3	146	0.070058s
	$10+500i,0.8)$	$+6.013632243569482i$		$+6.016364243229925i$			
				-4.100998516155065	3	146	0.160582s
				$+6.017586881185369i$			
22	$(2,5,10-500i,-0.8)$	0.999450314116122	11.844837s	0.999450314116122	16	9	0.034498s
		$-0.015980509652011i$		$-0.015980509652011i$			
				0.999450314116122	16	9	0.084371s
				$-0.015980509652011i$			

Table 9: Table showing a variety of test cases from Appendix C, their correct solution and the time taken to generate them using MATLAB, the solution computed using Taylor series methods (a) and (b), the number of digits of accuracy they have, the number of terms computed N, and the time taken using that method. Full results are shown in Appendix F. Note that for case 12, MATLAB did not generate the correct result, as deatiled in Section 2.2, and the correct result is instead shown in this table.

The function is computed very accurately for the cases in rows 1–5 and 9, which include cases with large real parameter values (with $c < a < b < 0$) and one with a large imaginary part for c. However, rows 6–8 in Table 9 illustrate that the Taylor series method struggles greatly in cases in which either $|a|$ or $|b|$ is much greater than $|c|$; for these cases, other methods such as recurrence relations (explained in Section 4.8) should be used. We conclude that the region in which the Taylor series methods seem to be effective is $|z| \lesssim 0.9$.

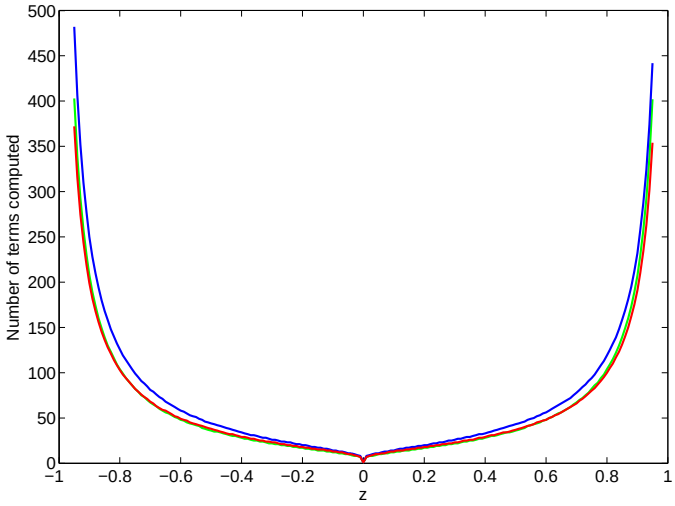

Figure 3: Graph showing the number of terms which need to be computed using Taylor series method (a) for evaluating ${}_2F_1(a,b;c;z)$ for real $z \in [-0.95, 0.95]$, when $a = 1.5$, $b = 1 + 2i$, $c = 4.5 + 5i$ (red), $a = 0.15$, $b = 0.2$, $c = 1.1$ (green) and $a = 3$, $b = 2$, $c = 6.5$ (blue). We generated 14 digit accuracy when computing ${}_2F_1$ for these parameters and for values of z shown in this graph.

4.3 Writing the Gauss hypergeometric function as a single fraction

In this section, we aim to compute the Gauss hypergeometric function by representing it as a single fraction. We will analyse the accuracy and robustness of this approach, outlining any specific parameter regimes for which it is particularly effective.

As discussed for the confluent hypergeometric function in Section 3.3 and in reference [46], the goal of this method is to express ${}_2F_1(a,b;c;z)$ as a single fraction by using recurrence relations. The recurrence relation on this occasion reads $\alpha_0 = 0,\ \beta_0 = 1,\ \gamma_0 = 1,\ \zeta_0 = 1$, and for $j = 1, 2, \ldots$:

$$\begin{aligned} \alpha_j &= (\alpha_{j-1} + \beta_{j-1}) \times j \times (c + j - 1), \\ \beta_j &= \beta_{j-1} \times (a + j - 1) \times (b + j - 1) \times z, \\ \gamma_j &= \gamma_{j-1} \times j \times (c + j - 1), \\ \zeta_j &= \frac{\alpha_j + \beta_j}{\gamma_j}, \end{aligned} \tag{4.7}$$

This generates a sequence of approximations, $\{\zeta_j,\ j = 1, 2, \ldots\}$, to ${}_2F_1(a,b;c;z)$. The stopping criterion we use is that, for some j, $\frac{|\zeta_{j+1}-\zeta_j|}{|\zeta_j|}$, $\frac{|\zeta_j-\zeta_{j-1}|}{|\zeta_{j-1}|}$ and $\frac{|\zeta_{j-1}-\zeta_{j-2}|}{|\zeta_{j-2}|}$ must be less than

the prescribed tolerance *tol*.

Case	(a,b,c,z)	Correct ${}_2F_1(a,b;c;z)$	Time taken	Single fraction ($tol = 10^{-15}$)	Acc.	N	Time taken
1	(0.1,0.2,0.3,0.5)	1.046432811217352	10.471900s	1.046432811217352	16	42	0.128695s
4	$(10^{-8},10^{-8},10^{-8},10^{-6})$	1.000000000000010	9.971552s	1.000000000000010	16	3	0.070016s
5	$(10^{-8},-10^{-6},10^{-12},$ $-10^{-10}+10^{-12}i)$	1.000000000001000 $+0.000000010000000i$	10.031475s	1.000000000001000 $+0.000000010000000i$	16	3	0.084714s
7	$(1,-1+10^{-12}i,1,-0.8)$	1.800000000000000 $-0.000000000001058i$	10.507413s	1.800000000000000 $-0.000000000001058i$	16	10	0.115477s
10	$(2+10^{-9},3,5,-0.75)$	0.492238858852651	11.833417s	0.492238858852701	13	97	0.148518s
15	$(-1000,-2000,-4000.1,-0.5)$	$5.233580403196932 \times 10^{94}$	13.034198s	$3.206764029878514 \times 10^{55}$	0	53	0.172314s
19	(10,5,−300.5,0.5)	$-3.852027081523919 \times 10^{32}$	11.941827s	0.921182716632848	0	12	0.092340s

Table 10: Table showing the result of using 'hypergeom' for a variety of test cases and the times taken, the results from the single fraction method described in this section, the number of digits of accuracy, the number of terms computed and the time taken. Full numerical results are shown in Appendix F.

The method of expressing the Gauss hypergeometric function as a single fraction is, as shown in Table 10, much faster than the built-in MATLAB function, and works well for small values of the parameters and variable (for example, $|a|, |b|, |c| \lesssim 20$, $|z| \lesssim 0.95$ as a guide). In particular, for the same reason that the method is successful for computing ${}_1F_1$ for small $|b|$ or b close to $-m$, $m \in \mathbb{Z}^+ \cup \{0\}$, as explained in Section 3.3, the method is more successful for computing ${}_2F_1$ the smaller $|c|$ is or the closer c is to an integer.

However, this method does struggle more than the Taylor series methods when either a or b has large magnitude (roughly greater than 50), due to greater risk of overflow (meaning the computer is attempting to compute values that are larger than it is able to compute) due to the potentially large numerators and denominators in (4.7). For these cases, other methods, including the use of recurrence relations as described in Section 4.8, should be applied.

4.4 Quadrature methods

As discussed for $M(a;b;z)$ in Section 3.6, we now explore applying the method of Gauss-Jacobi quadrature to compute ${}_2F_1(a,b;c;z)$, when $\mathrm{Re}(c) > \mathrm{Re}(b) > 0$, $|\arg(1-z)| < \pi$. As stated in [3], the function ${}_2F_1(a,b;c;z)$ has a known integral representation,

$$ {}_2F_1(a,b;c;z) = \frac{\Gamma(c)}{\Gamma(b)\Gamma(c-b)} \int_0^1 (1-zt)^{-a} w_{b,c}(t) \mathrm{d}t, \quad \mathrm{Re}(c) > \mathrm{Re}(b) > 0, \tag{4.8} $$

valid for $|\arg(1-z)| < \pi$, where

$$w_{b,c}(t) = (1-t)^{c-b-1}t^{b-1}.$$

We note that in (4.8), the parameters a and b can be interchanged due to the basic series definition (4.1). Transforming $t \mapsto \frac{1}{2}\tilde{t} + \frac{1}{2}$, with Jacobi parameters $\tilde{\alpha} = c-b-1$, $\tilde{\beta} = b-1$, as recommended in [26], we obtain

$$\begin{aligned}\int_0^1 (1-zt)^{-a} w_{b,c}(t)\mathrm{d}\tilde{t} &= \frac{1}{2^{c-1}}\int_{-1}^{1}\left(\{1-\frac{1}{2}z\} - \frac{1}{2}z\tilde{t}\right)^{-a}(1-\tilde{t})^{c-b-1}(1+\tilde{t})^{b-1}\mathrm{d}t \\ &= \sum_{j=1}^{N_{mesh}} w_j^{GJ}\left(\{1-\frac{1}{2}z\} - \frac{1}{2}zt_j^{GJ}\right)^{-a} + E_{N_{mesh}}(a;b;z),\end{aligned}$$

where t_j^{GJ} and w_j^{GJ} are the Gauss-Jacobi nodes and weights on $[-1,1]$ as defined in Section 3.6, and N_{mesh} is the number of mesh points. Error bounds for this method are discussed in [26].

We note that if $\mathrm{Re}(c) > \mathrm{Re}(a) > 0$, then the parameters a and b can be switched in the definition of ${}_2F_1(a,b;c;z)$, and the method of Gauss-Jacobi quadrature can be applied. As was the case for ${}_1F_1$, the integrand in (4.8) blows-up at the end-points of the integral, which motivates the choice of Gauss-Jacobi quadrature to perform the required integration numerically.

Case	(a,b,c,z)	Correct ${}_2F_1(a,b;c;z)$	Time taken	Gauss-Jacobi ($N_{mesh}=200$)	Acc.	N_{crit}	Time taken
1	(0.1,0.2,0.3,0.5)	1.046432811217352	10.471900s	1.046432811217352	16	10	0.226160s
9	$(100, 200, 350, i)$	$5.686708048303445 \times 10^{155}$ $+4.471204020179333 \times 10^{155}i$	12.194382s	NaN+NaNi	0	N/A	N/A
10	$(2+10^{-9}, 3, 5, -0.75)$	0.492238858852651	11.833417s	0.492238858852651	16	10	0.239335s
18	$(5, -300, 10, 0.5)$	$1.661006238211309 \times 10^{-7}$	11.922410s	$1.661006238211367 \times 10^{-7}$	14	40	0.269030s
25	$(1, 0.9, 2, e^{i\pi/3})$	0.932633569241998 $+0.475200538581622i$	>5 minutes	0.932633569241997 $+0.475200538581622i$	14	10	0.261462s
28	$(4, 1.1, 2,$ $0.5+(0.5\sqrt{3}-0.01)i)$	-0.470097672835090 $+0.500986178581549i$	104.287255s	-0.470097672835091 $+0.500986178581549i$	15	20	0.287255s

Table 11: Table showing the true value of ${}_2F_1(a,b;c;z)$ for a range of test cases and the time taken to compute this using 'hypergeom', the result obtained using Gauss-Jacobi quadrature with 200 mesh points, the critical number of mesh points N_{crit} required to obtain 10 digit accuracy (where N_{mesh} is increased by increments of 10 until 10 digit accuracy is obtained), and the computation time with N_{crit} mesh points. Where 'N/A' is written, 5000 mesh points were not sufficient to give 10 digit accuracy. Full results are in Appendix F.

The results from Table 11 illustrate that the method of applying Gauss-Jacobi quadrature to the integral in (4.8) is a useful method for computing the Gauss hypergeometric function

when $\mathrm{Re}(c) > \mathrm{Re}(b) > 0$, apart from parameter values with modulus at least 50–100. The method works well near the points $e^{\pm i\pi/3}$, which are difficult for computational purposes as explained in Sections 4.6 and 4.7, as shown by the fifth and sixth rows of Table 11.

Therefore, as for ${}_1F_1$ (as detailed in Section 3.6), applying Gauss-Jacobi quadrature for computing ${}_2F_1$ is an extremely useful method when the parameters do not have too large a modulus.

4.5 Solving the hypergeometric differential equation

We now aim to solve the differential equation (4.2) numerically, and analyse their effectiveness for the computation of the Gauss hypergeometric function. As discussed for ${}_1F_1(a;b;z)$ in Section 3.7, the function ${}_2F_1(a,b;c;z)$ is known to satisfy a differential equation; this is the **hypergeometric differential equation** stated in (4.2). Solving this differential equation is the primary method, apart from using the Taylor series (4.1), that is recommended in [57] for computing the function ${}_2F_1$.

As detailed in [3, 8], when none of c, $c-a-b$ or $a-b$ is equal to an integer, two fundamental solutions are known for z near each of the three singular points of the differential equation. Near the singular point $z = 0$, the two fundamental solutions are

$$ {}_2F_1(a,b;c;z), \quad z^{1-c}\,{}_2F_1(a-c+1,b-c+1;2-c;z); \tag{4.9} $$

near the singular point $z = 1$, the fundamental solutions are

$$ {}_2F_1(a,b;a+b+1-c;1-z), \quad (1-z)^{c-a-b}\,{}_2F_1(c-a,c-b;c-a-b+1;1-z); \tag{4.10} $$

and near $z = \infty$, the two fundamental solutions are given by

$$ z^{-a}\,{}_2F_1\left(a,a-c+1;a-b+1;\frac{1}{z}\right), \quad z^{-b}\,{}_2F_1\left(b,b-c+1;b-a+1;\frac{1}{z}\right). \tag{4.11} $$

The above 6 solutions can be transformed using the transformation formulae detailed in Section 4.6 to obtain 18 further solutions. These 24 solutions are known as **Kummer's solutions** to (4.2), as discussed in [3, 9].

The case where a, $c-a-b$ or $a-b$ are equal to an integer are discussed in [3]. For the purposes of computing the Gauss hypergeometric function, solving the differential equation

is not a suitable method in this case due to the relative complexity of determining the solutions.

For other values of a, b, c, we consider the differential equation method for z such that $|z| \leq 1$. It is expected that, if the value of the hypergeometric function is required near $z = 0$, the differential equation (4.2) could be numerically integrated along outward rays from the origin, using initial conditions found from the series definition (4.1) of ${}_2F_1(a, b; c; z)$,

$$w(0) = 1, \ w'(0) = \frac{ab}{c}, \tag{4.12}$$

in order to find an approximation to ${}_2F_1(a, b; c; z)$ near $z = 0$. This is the method we pursue in this section.

We find that this method starts to fail when the numerical integration passes through points that are close to $z = 1$, which is explained by the fact that the differential equation (4.2) has different fundamental solutions near $z = 1$ from those near $z = 0$. In this case, one can use the following known result for $\mathrm{Re}(c - a - b) > 0$ stated in [58]:

$${}_2F_1(a, b; c; 1) = \frac{\Gamma(c)\Gamma(c - a - b)}{\Gamma(c - a)\Gamma(c - b)},$$

and the limiting cases discussed in [6],

$$\lim_{z\to 1^-} \frac{{}_2F_1(a, b; c; z)}{\log\left(1 - \frac{1}{z}\right)} = \frac{\Gamma(c)}{\Gamma(a)\Gamma(b)}, \quad \lim_{z\to 1^-} \frac{{}_2F_1(a, b; c; z)}{(1 - z)^{c-a-b}} = \frac{\Gamma(c)\Gamma(a + b - c)}{\Gamma(a)\Gamma(b)},$$

for $\mathrm{Re}(c - a - b) = 0$ and $\mathrm{Re}(c - a - b) < 0$ respectively.

For example, for the case $\mathrm{Re}(c - a - b) > 0$, the following substitutions can be made, which are motivated by the first fundamental solution near $z = 1$ given in (4.10):

$$z = 1 - \hat{z}, \quad a = \hat{a}, \quad b = \hat{b}, \quad c = \hat{a} + \hat{b} + 1 - \hat{c} \ \Leftrightarrow \ \hat{c} = a + b + 1 - c.$$

Then, the following boundary conditions for the differential equation (4.2) can be found using (2.7):

$$w(\hat{z} = 0) = \ w(z = 1) = \frac{\Gamma(c)\Gamma(c - a - b)}{\Gamma(c - a)\Gamma(c - b)} = \frac{\Gamma(\hat{a} + \hat{b} + 1 - \hat{c})\Gamma(1 - \hat{c})}{\Gamma(\hat{b} + 1 - \hat{c})\Gamma(\hat{a} + 1 - \hat{c})}, \tag{4.13}$$

$$\begin{aligned} w'(\hat{z} = 0) = w'(z = 1) &= \frac{ab}{c} \times \frac{\Gamma(c + 1)\Gamma(c - a - b - 1)}{\Gamma(c - a)\Gamma(c - b)} \\ &= \frac{\hat{a}\hat{b}}{\hat{a} + \hat{b} + 1 - \hat{c}} \times \frac{\Gamma(\hat{a} + \hat{b} + 2 - \hat{c})\Gamma(2 - \hat{c})}{\Gamma(\hat{b} + 1 - \hat{c})\Gamma(\hat{a} + 1 - \hat{c})}. \end{aligned} \tag{4.14}$$

These are then used to solve the differential equation

$$\hat{z}(1-\hat{z})\frac{\mathrm{d}^2 w}{\mathrm{d}\hat{z}^2} - [\hat{c} - (\hat{a}+\hat{b}+1)\hat{z}]\frac{\mathrm{d}w}{\mathrm{d}\hat{z}} - \hat{a}\hat{b}w = 0. \tag{4.15}$$

For each case, the RK4 method detailed in (3.29)–(3.30) can be applied with $z_0 = 0$, $\mathbf{w} = (w, w')^T$, $\mathbf{w}_0 = (1, \frac{ab}{c})^T$ and $\mathbf{f}(z) = (w_2, -\frac{1}{z(1-z)}\{[c-(a+b+1)z]w_2 - abw_1\})^T$.

Case	(a,b,c,z)	RK4 ($N_{mesh} = 500$)	Acc.	N_{crit}	Time	ode45/ode113/ode15s	Acc.	N
1	(0.1,0.2,0.3,0.5)	1.046432811211848	12	150	0.417635s	1.046432754293317	7	41
						1.046431632552768	6	13
						1.046556420197934	4	13
2	$(-0.1, 0.2, 0.3, 0.5)$	0.95643421097278	10	250	0.474317s	0.956434253273899	7	41
						0.956433620931461	5	13
						0.956431095172177	5	14
4	$(10^{-8}, 10^{-8}, 10^{-8}, 10^{-6})$	1.000000000000000	14	50	0.344963s	1.000000000000011	15	41
						1.000000000000011	15	11
						1.000000000000011	15	11
8	$(2+8i, 3-5i,$	$6.882465637979511 \times 10^3$	6	4800	3.858745s	$6.869602468684762 \times 10^3$	2	85
	$\sqrt{2}-\pi i$,0.75)	$-6.596556746271624 \times 10^3 i$				$-6.584405220506018 \times 10^3 i$		
						$6.992518366536974 \times 10^3$	1	36
						$-6.464586065423973 \times 10^3 i$		
						$6.443566948572507 \times 10^3$	1	57
						$-6.856945927744196 \times 10^3 i$		
10	$(2+10^{-9}, 3, 5, -0.75)$	0.492238858850299	11	250	0.466852s	0.493421438461613	2	73
						0.493420802980853	2	28
						0.515594065014172	0	37
18	$(5, -300, 10, 0.5)$	$1.660425565716309 \times 10^{-7}$	3	16850	17.931442s	$1.664060389193657 \times 10^{-7}$	3	397
						$1.661875913325594 \times 10^{-7}$	4	194
						$1.537384779497713 \times 10^{-7}$	1	90
20	$(2+200i, 5, 10, 0.6)$	$1.486356299813440 \times 10^{-7}$	2	N/A	N/A	$1.574766444934557 \times 10^{-7}$	1	737
		$+5.756173666779157 \times 10^{-7} i$				$+5.841973382083921 \times 10^{7} i$		
						$1.810879448347993 \times 10^{-7}$	1	335
						$+5.627011113641864 \times 10^{-7} i$		
						$1.426772970328000 \times 10^{-7}$	1	420
						$+5.656148380810558 \times 10^{-7} i$		

Table 12: Table showing the true value of ${}_2F_1(a, b; c; z)$ for a selection of test cases, the results obtained using the RK4 method with 500 mesh points, the number of mesh points required to obtain 10 digit accuracy (where we increased N_{mesh} by increments of 50 until 10 digit accuracy was obtained), and the computation time with N_{crit} mesh points; here 'N/A' is written when 20000 mesh points were not sufficient to give 10 digit accuracy. Also stated are the results using 'ode45', 'ode113' and 'ode15s', their accuracy and the number of points N that MATLAB uses to produce the solution vector. Full results are detailed in Appendix F.

As was the case for the function ${}_1F_1$, we found the RK4 method to be the most effective method for solving the differential equation out of those tested; details of another method tried are included in Appendix H.2. We applied the RK4 method to (4.2), with results shown in Table 12 and Appendix F. Also in Table 12 are results obtained by applying the three

built-in MATLAB solvers ode45, ode113 and ode15s to the problem, integrating from 10^{-3}, apart from case 4 when we integrated from 10^{-11} and case 10 where we integrated from -10^{-3}.

Using the results shown in Table 12, we deduce that the RK4 method is more effective for computing ${}_2F_1(a,b;c;z)$ than it is for computing ${}_1F_1(a;b;z)$ due to the fact that we only need to apply the method in the region $|z| \leq 1$. Figure 4 shows three cases and the profile of their errors on $z \in [-1,1]$ for real z. We conclude from this graph and the other results obtained that the RK4 method is useful provided $|a|, |b|, |c| \leq 5$ if 500 mesh points are used, although if more mesh points are used, the method should work over a larger set of values of a, b and c.

Table 12 shows results generated when we tried to compute the first solution of (4.9). However using the method described by the equation (4.15) and boundary conditions (4.13) and (4.14), we can generate solutions close to $z = 1$. For example, this method produced 10 digit accuracy when computing ${}_2F_1(0.2, 0.3; 0.4; 0.9)$ using 450 mesh points. We therefore find that computing solutions near $z = 1$ using the RK4 method is also a viable method.

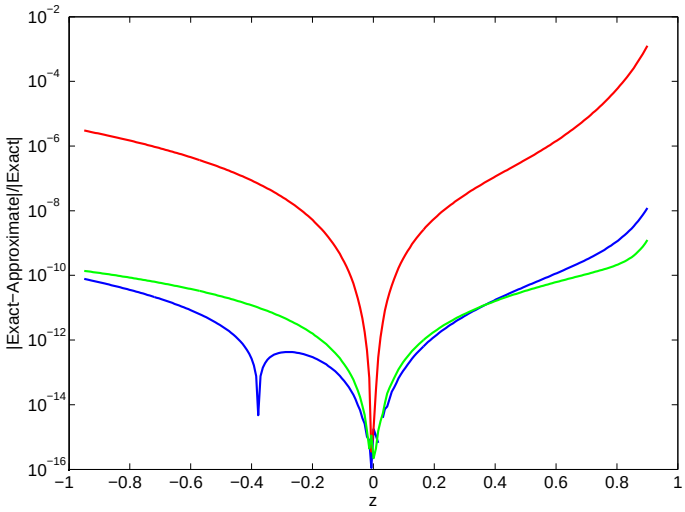

Figure 4: Graph showing the profile of relative error (which we define as the magnitude of the difference between exact and computed solutions divided by the magnitude of the exact solution) when applying the RK4 method over real $z \in [-1,1]$ for $(a,b,c) = (1, 2.5, 3.75)$ (blue), $(12.5, 8.25, 10)$ (red) and $(0.1, 0.2, 0.4)$ (green) using 200 mesh points.

4.6 Transformation formulae

As the Gauss hypergeometric series (4.1) converges only for $|z| < 1$, and as it converges more rapidly the smaller $|z|$ is, it is important to use transformation formulae that reduce the problem of carrying out a computation for a value of $|z|$ close to or greater than 1 to a problem of computing the series for a new variable w, where the value of $|w|$ is much smaller. We describe such transformation formulae in this section.

The idea of these transformations is to map as large a region of the complex plane as possible onto discs $|w| \leq \rho$, for a positive real number $0 < \rho \leq 1$, preferably as close to 0 as possible. This is desirable because the function ${}_2F_1$ can be computed faster and more accurately when $|z|$ is close to 0. If we can find representations of ${}_2F_1$ which allow us to carry out the computation in terms of the new variable w, we are likely to obtain more accurate results than we obtained using methods previously described. For real z, transformation formulae are written in Table 13, which map any $z \in \mathbb{R}$ to a new variable $w \in [0, \frac{1}{2}]$, in other words a special case where the variable z is real, and with $\rho = \frac{1}{2}$.

Case	Interval	Transformation
1	$-\infty < z < -1$	$w = \frac{1}{1-z}$
2	$-1 \leq z < 0$	$w = \frac{z}{z-1}$
3	$0 \leq z \leq \frac{1}{2}$	$w = z$
4	$\frac{1}{2} < z \leq 1$	$w = 1 - z$
5	$1 < z \leq 2$	$w = 1 - \frac{1}{z}$
6	$2 < z < +\infty$	$w = \frac{1}{z}$

Table 13: List of transformations of $z \in \mathbb{R}$, stated in [24], for which $0 \leq w \leq \rho = \frac{1}{2}$.

From Table 13, we observe that, for real variable z, we can compute the hypergeometric function in terms of hypergeometric functions of a new real variable w with $|w| \leq \rho = \frac{1}{2}$ using (4.16)–(4.20), which we hope will ensure faster convergence than using the original variable z.

However, for complex z, the problem is more complicated. In Figure 5, we show plots of $|w| = \rho$ for each of the 6 expressions for w shown in Table 13 for $\rho = 0.6$ and $\rho = 0.8$. If we wish to apply one of the 6 transformations of Table 13, we require that $|w| < \rho$ be satisfied for at least one representation of w in the table. The region in Figure 5 in which none of the

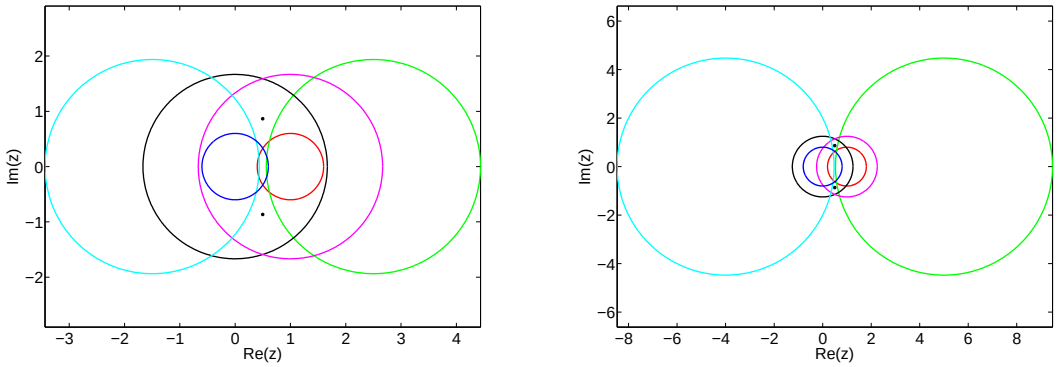

Figure 5: Illustrations of the curves $|z| = \rho$ (dark blue), $\left|\frac{1}{z}\right| = \rho$ (black), $|1-z| = \rho$ (red), $\left|\frac{1}{1-z}\right| = \rho$ (purple), $\left|\frac{z}{z-1}\right| = \rho$ (sky blue) and $\left|1-\frac{1}{z}\right| = \rho$ (green), along with the points $z = e^{\pm i\pi/3}$, for $\rho = 0.6$ (left) and $\rho = 0.8$ (right). [Adapted from illustrations in [28, 30].]

representations of w satisfy $|w| < \rho$ is the region around the points $z = e^{\pm i\pi/3} = \frac{1}{2}(1 \pm i\sqrt{3})$, which are marked as dots. As ρ is increased towards 1, the region in which none of the transformations satisfy $|w| < \rho$ gets smaller, but remains around the points $z = e^{\pm i\pi/3}$ due to the fact that the set $\{e^{i\pi/3}, e^{-i\pi/3}\}$ is mapped to itself by each of the 6 transformations of Table 13. The case $z \approx e^{\pm i\pi/3}$ is discussed in Section 4.7.

Now that we have established the regions in which transformations can be applied, we consider the known transformations (4.16)–(4.20) and their regions of validity, discussed in [3, 22, 24, 70]. These correspond to cases 1,2,4,5 and 6 in Table 13 respectively.

$$
\begin{aligned}
{}_2F_1(a,b;c;z) &= (1-z)^{-a}\frac{\Gamma(c)\Gamma(b-a)}{\Gamma(b)\Gamma(c-a)}\ {}_2F_1\left(a,c-b;a-b+1;\frac{1}{1-z}\right) \\
&\quad +(1-z)^{-b}\frac{\Gamma(c)\Gamma(a-b)}{\Gamma(a)\Gamma(c-b)}\ {}_2F_1\left(b,c-a;b-a+1;\frac{1}{1-z}\right), \\
&\qquad\qquad |\arg(1-z)| < \pi;
\end{aligned}
\tag{4.16}
$$

$$
{}_2F_1(a,b;c;z) = (1-z)^{-a}\ {}_2F_1\left(a,c-b;c;\frac{z}{z-1}\right), \qquad z \in \mathbb{C}; \tag{4.17}
$$

$$
\begin{aligned}
{}_2F_1(a,b;c;z) &= \frac{\Gamma(c)\Gamma(c-a-b)}{\Gamma(c-a)\Gamma(c-b)}\ {}_2F_1(a,b;a+b-c+1;1-z) \\
&\quad +(1-z)^{c-a-b}\frac{\Gamma(c)\Gamma(a+b-c)}{\Gamma(a)\Gamma(b)}\ {}_2F_1(c-a,c-b;c-a-b+1;1-z), \\
&\qquad\qquad |\arg(1-z)| < \pi;
\end{aligned}
\tag{4.18}
$$

$$
\begin{aligned}
{}_2F_1(a,b;c;z) &= z^{-a}\frac{\Gamma(c)\Gamma(c-a-b)}{\Gamma(c-a)\Gamma(c-b)}\ {}_2F_1\left(a,a-c+1;a+b-c+1;1-\frac{1}{z}\right) \qquad (4.19)\\
&\quad +z^{a-c}\,(1-z)^{c-a-b}\,\frac{\Gamma(c)\Gamma(a+b-c)}{\Gamma(a)\Gamma(b)}\ {}_2F_1\left(c-a,1-a;c-a-b+1;1-\frac{1}{z}\right),\\
&\qquad\qquad |\arg z|<\pi, \quad |\arg(1-z)|<\pi;\\
{}_2F_1(a,b;c;z) &= (-z)^{-a}\frac{\Gamma(c)\Gamma(b-a)}{\Gamma(b)\Gamma(c-a)}\ {}_2F_1\left(a,a-c+1;a-b+1;\frac{1}{z}\right) \qquad (4.20)\\
&\quad +(-z)^{-b}\frac{\Gamma(c)\Gamma(a-b)}{\Gamma(a)\Gamma(c-b)}\ {}_2F_1\left(b-c+1,b;b-a+1;\frac{1}{z}\right),\\
&\qquad\qquad |\arg z|<\pi, \quad |\arg(1-z)|<\pi.
\end{aligned}
$$

We tested these formulae on a large range of parameters and variable and found that they succesfully computed ${}_2F_1$ for a variable which is close to the unit disc or which has modulus greater than 1, in terms of new variables with smaller magnitude. However, due to the presence of $\Gamma(a-b)$, $\Gamma(b-a)$, $\Gamma(c-a-b)$ and $\Gamma(a+b-c)$ in the numerators of (4.16)–(4.20), the cases $b-a\in\mathbb{Z}$ and $c-a-b\in\mathbb{Z}$ cannot be handled using these formulae. These cases are discussed in Appendix A.

4.7 Analytic continuation formulae for z near $e^{\pm i\pi/3}$

A major problem when applying the transformation formulae of Section 4.6 is that this is not a viable method for the region near the points $z=e^{\pm i\pi/3}=\frac{1}{2}(1\pm i\sqrt{3})$. The reason for this is that, whatever value $0<\rho<1$ is taken for $|z|<\rho$, it is not possible to map the points $z=e^{\pm i\pi/3}$ onto w within a disc of radius less than 1; the points are mapped to themselves or each other when any of the 6 transformations in Table 13 is applied. As ρ is increased and approaches 1, an increasing number of points close to $z=e^{\pm i\pi/3}$ are mapped onto such a disc, but the points themselves can never be, and the points very close to them will require a computation involving a prohibitively large value of ρ, for which the methods discussed so far will not generate an accurate result.

In [14], an expansion is discussed that attempts to resolve this issue. This is given in the

form of the *continuation formula*:

$$\frac{{}_2F_1(a,b;c;z)}{\Gamma(c)} = \frac{\Gamma(b-a)}{\Gamma(b)\Gamma(c-a)}(z_0-z)^{-a}\sum_{j=0}^{\infty} d_j(a,z_0)(z-z_0)^{-n} \tag{4.21}$$
$$+\frac{\Gamma(a-b)}{\Gamma(a)\Gamma(c-b)}(z_0-z)^{-b}\sum_{j=0}^{\infty} d_j(b,z_0)(z-z_0)^{-n}, \quad |\arg(z_0-z)|<\pi,$$

where $b-a$ is not an integer, and d_j is defined by

$$\begin{aligned} d_{-1}(\upsilon,z_0) &= 0,\ d_0(\upsilon,z_0)=1, \\ d_j(\upsilon,z_0) &= \frac{j+\upsilon-1}{j(j+2\upsilon-a-b)}\left[\{(j+\upsilon)(1-2z_0)+(a+b+1)z_0-c\}d_{j-1}(\upsilon,z_0)\right. \\ &\qquad \left.+z_0(1-z_0)(j+\upsilon-2)d_{j-2}(\upsilon,z_0)\right], \quad j=1,2,\ldots . \end{aligned} \tag{4.22}$$

This series converges everywhere outside the disc $|z-z_0| = \max\{|z_0|, |z_0-1|\}$, so an appropriate z_0 must be chosen to carry out this method. The case noted in particular in [14] is that of $z_0=\frac{1}{2}$, which would result in convergence outside the disc $\left|z-\frac{1}{2}\right|=\frac{1}{2}$, including in particular, points close to $e^{\pm i\pi/3}$. At these points, as shown in Section 4.6, it is difficult to compute ${}_2F_1$ due to the ineffectiveness of applying transformation formulae.

Now, the above expansion is not valid if $b-a$ is exactly equal to an integer. In particular, if $b-a$ were equal to a non-zero integer, one of the Gamma functions in the numerators of the two terms in (4.21) would be infinitely large. An alternative formula is given for the case $b-a=0$ in [14]; this is

$$\frac{{}_2F_1(a,a;c;z)}{\Gamma(c)} = \frac{1}{\Gamma(a)\Gamma(c-a)}(z_0-z)^{-a}\sum_{j=0}^{\infty}\frac{(a)_j}{j!}(z-z_0)^{-j} \tag{4.23}$$
$$\times\left[e_j(a,z_0)\{2\psi(1+j)-\psi(a+j)-\psi(c-a)+\log(z_0-z)\}-f_j(z_0)\right],$$

where e_j and f_j are defined as follows when we wish to compute the hypergeometric function ${}_2F_1(a,b;c;z)$:

$$\begin{aligned} e_{-1}(\upsilon,z_0) &= 0,\ e_0(\upsilon,z_0)=1, \\ je_j(\upsilon,z_0) &= [(j+\upsilon)(1-2z_0)+(a+b+1)z_0-c]e_{j-1}(\upsilon,z_0) \\ &\qquad +z_0(1-z_0)(j-1+2\upsilon-a-b)e_{j-2}(\upsilon,z_0), \quad j=1,2,..., \end{aligned} \tag{4.24}$$

$$\begin{aligned} f_{-1}(z_0) &= 0,\ f_0(z_0)=0, \\ jf_j(z_0) &= [(j+a)(1-2z_0)+(2a+1)z_0-c]f_{j-1}(z_0)+z_0(1-z_0)(j-1)f_{j-2}(z_0) \\ &\qquad +(1-2z_0)e_{j-1}(a,z_0)+2z_0(1-z_0)e_{j-2}(a,z_0), \quad j=1,2,\ldots . \end{aligned} \tag{4.25}$$

Here, $\psi(y)$ is the **polygamma function**, which is defined to be

$$\psi(y) = \frac{\Gamma'(y)}{\Gamma(y)}, \tag{4.26}$$

and is discussed further in Appendix I.

This series is again defined outside $|z - z_0| = \max\{|z_0|, |z_0 - 1|\}$. The general case $b - a \in \mathbb{Z}\backslash\{0\}$ is discussed in [37] using a limiting process of the Gamma function, but due to time constraints we did not implement it for this project.

Case	(a,b,c,z)	Correct $M(a;b;z)$	Time taken	Analytic cont. $(tol = 10^{-15})$	Acc.	N	Time taken
25	$(1, 0.9, 2, e^{i\pi/3})$	0.932633569241998 $+0.475200538581623i$	> 5 minutes	0.932633569242002 $+0.475200538581619i$	12	63	0.170717s
29	$(5, 2.2, -2.5,$ $0.49 + 0.5\sqrt{3}i)$	$1.084589030597025 \times 10^3$ $-5.115786480030682 \times 10^3 i$	107.090833s	$1.084589030597452 \times 10^3$ $-5.115786480028667 \times 10^3 i$	11	125	0.180274s
30	$(\frac{2}{3}, 1, \frac{4}{3}, e^{i\pi/3})$	0.883319375142725 $+0.509984679019064i$	> 5 minutes	0.883319375142725 $+0.509984679019064i$	16	58	0.146477s
–	$(5, 7.5, 2.5, 5)$	$1.738170683483183 \times 10^{-4}$	14.338713s	$1.738170683483184 \times 10^{-4}$	15	27	0.147836s
–	$(1, 1.5, 3, -4 + 3i)$	0.323077064587889 $+0.130063349815216i$	12.791476s	0.323077064587889 $+0.130063349815216i$	16	17	0.129981s

Table 14: Table showing the true result for ${}_2F_1(a, b; c; z)$ for a number of test cases and the time taken to generate the solution using 'hypergeom', the results obtained from the theory of analytic continuation discussed in this section, the accuracy, the number of terms taken and the time taken using this method.

The results in Table 14 show that using the expansion given in (4.21) generates excellent accuracy in a region that methods previously discussed could not produce results of similar accuracy in. The method is not only effective when z is equal or close to $e^{\pm i\pi/3}$, but also when z lies outside the unit disc, as shown by the fourth and fifth rows of Table 14. The method generally struggles with $|a|$ or $|b| \gtrsim 30$ or $|c| \gtrsim 70$; this problem can be resolved by applying recurrence relations as detailed in Section 4.8.

4.8 Recurrence relations

As for the confluent hypergeometric function in Section 3.8, we aim to overcome the lack of accuracy that occurs when attempting to compute the Gauss hypergeometric function when one or more of the values of $|\mathrm{Re}(a)|$, $|\mathrm{Re}(b)|$ and $|\mathrm{Re}(c)|$ is large. We explain how this problem can be addressed by using the technique of recurrence relations for this function. Using these ideas, we can reformulate such a problem as one involving values of $|\mathrm{Re}(a)|$,

$|\mathrm{Re}(b)|$, $|\mathrm{Re}(c)|$ closer to zero, which are computed much more accurately by most methods implemented, as verified by results so far.

The 4 main recurrence relations involving the Gauss hypergeometric function ${}_2F_1(a,b;c;z)$ discussed in literature such as [23, 28, 29, 32, 66], and the most basic hypergeometric functions that are solutions of these relations, are shown below, in the notation of [28]:

1. **Recurrence:** $(c-a-n)(c-b-n)(c-a-b-2n-1)y_{n-1}$
$+(c-a-b-2n)\left[c(a+b-c+2n)+c-2(a+n)(b+n)\right.$
$\left.+z\{(a+b+2n)(c-a-b-2n)+2(a+n)(b+n)-c+1\}\right]y_n$
$+(a+n)(b+n)(c-a-b-2n+1)(1-z)^2y_{n+1}=0,$

   Solution: $y_n={}_2F_1(a+n,b+n;c;z),$

2. **Recurrence:** $-(a-c+2n)(a-c+2n-1)(b-c+2n-1)(b-c+2n)zUy_{n-1}$
$+(c-n)(c_1U+c_2V+c_3UV)y_n$
$+(a+n)(b+n)(c-n)(c-n-1)(1-z)^3Vy_{n+1}=0,$

   where: $c_1=(1-z)(b-c)(b-1)[a-1+z(b-c-1)],$
$c_2=b(b+1-c)(1-z)[a+z(b-c+2)],$
$c_3=c-2b-(a-b)z,$
$U=z(a+b-c+1)(a+b-c+2)+ab(1-z),$
$V=(1-z)(1-a-b-ab)+z(a+b-c-1)(a+b-c-2),$

   Solution: $y_n={}_2F_1(a+n,b+n;c-n;z),$

3. **Recurrence:** $z(a-c+2n)(a-c+2n-1)(b-c+n)[a+n+z(b-c+n+1)]y_{n-1}$
$+(c-n)\left[(a+n-1)(c-n-1)(b-c+n)+(a+n)(a+n-1)\right.$
$+(a+3b-4c+5n+2)z+(b-c+n)(b-c+n+1)(4a-c+5n-1)z^2$
$\left.-(a-b+n)(b-c+n)(b-c+n+1)z^3\right]y_n$
$-(a+n)(c-n)[a+n-1+z(b-c+n)]\,(1-z)^2y_{n+1}=0,$

   Solution: $y_n={}_2F_1(a+n,b;c-n;z),$

4. **Recurrence:** $(c+n)(c+n-1)(z-1)y_{n-1} + (c+n)\left[c+n-1\right.$
$\left. - \{2(c+n)-a-b-1\}z\right] y_n + (c-a+n)(c-b+n)zy_{n+1} = 0,$

Solution: $y_n = {}_2F_1(a,b;c+n;z).$

However, the solutions of the recurrence relations that we are interested in, much as in Section 3.8 for ${}_1F_1$, are the minimal solutions. But, unlike for ${}_1F_1$, the 4 recurrence relations have different minimal solutions in different regions of the complex plane. The minimal solutions of each of the above 4 recurrence relations, as stated in [29], are shown in Table 15.

Relation	Region of $\mathbb{C}$	Minimal solution
1	$\mathbb{C}\backslash\{z \le 0\}$	$\frac{\Gamma(1+a-c+n)\Gamma(1+b-c+n)}{\Gamma(1+a+b-c+2n)}\,{}_2F_1(a+n,b+n;1+a+b-c+2n;1-z)$
2	Inside curve in Fig. 6 (left)	$\left(\frac{z}{(z-1)^3}\right)^n \frac{\Gamma(b-c+1+2n)\Gamma(a-c+1+2n)}{\Gamma(a+n)\Gamma(b+n)\Gamma(1-c+n)\Gamma(2-c+n)}\,{}_2F_1(1-a+n,1-b+n;2-c+n;z)$
2	Outside curve in Fig. 6 (left)	$\frac{\Gamma(b-c+1+2n)\Gamma(a-c+1+2n)}{\Gamma(1-c+n)\Gamma(1+a+b-c+3n)}\,{}_2F_1(a+n,b+n;1+a+b-c+3n;1-z)$
3	Inside inner curve in Fig. 6 (right)	$\left(\frac{-z}{(1-z)^2}\right)^n \frac{\Gamma(b-c+1+n)\Gamma(a-c+1+n)}{\Gamma(a+n)\Gamma(1-c+n)\Gamma(2-c+n)}\,{}_2F_1(1-a+n,1-b;2-c+n;z)$
3	Between curves in Fig. 6 (right)	$\frac{\Gamma(b-c+1+n)\Gamma(a-c+1+2n)}{\Gamma(1-c+n)\Gamma(1+a+b-c+2n)}\,{}_2F_1\left(a+n,b;1+a+b-c+2n;1-z\right)$
3	Outside curves in Fig. 6 (right)	$\left(\frac{-z}{(1-z)^2}\right)^n \frac{\Gamma(1+a-c+2n)}{\Gamma(1-c+n)\Gamma(1+a-b+n)}\,{}_2F_1\left(1-b,-b+c-n;1+a-b+n;\frac{1}{z}\right)$
4	$\mathrm{Re}(z) < \frac{1}{2}$	${}_2F_1(a,b;c+n;z)$
4	$\mathrm{Re}(z) > \frac{1}{2}$	$\left(\frac{z-1}{z}\right)^n \frac{\Gamma(c+n)}{\Gamma(1-a-b+c+n)}\,{}_2F_1(1-a,1-b;1-a-b+c+n;1-z)$

Table 15: Minimal solutions of the 4 recurrence relations for ${}_2F_1(a,b;c;z)$ in different regions of the complex plane for z, as stated in [29].

Figure 6 illustrates the relevant regions for determining the minimal solutions of recurrence relations 2 (left on figure) and 3 (right on figure), which are referred to in Table 16. As detailed in [29], the curve described on the left of Figure 6 is defined to be the region which satisfies, in polar coordinates,

$$r = \sqrt{-9 + 6\sqrt{3}\cos\theta}, \quad -\frac{\pi}{6} \le \theta \le \frac{\pi}{6}, \quad \text{and} \;\; \mathrm{Re}(\sqrt{8z+1}) = 0,$$

together with the half-line $z \le \frac{1}{8}$ for real z, and the region shown on the right of Figure 6 is defined as follows in polar coordinates:

$$r = 2 + \cos\theta \pm \sqrt{\cos^2\theta + 4\cos\theta + 3}, \quad -\pi < \theta \le \pi.$$

Therefore, as for ${}_1F_1$ in Section 3.8, we apply two different methods; firstly, we take the minimal solutions of the 4 recurrence relations within specific regions and apply the

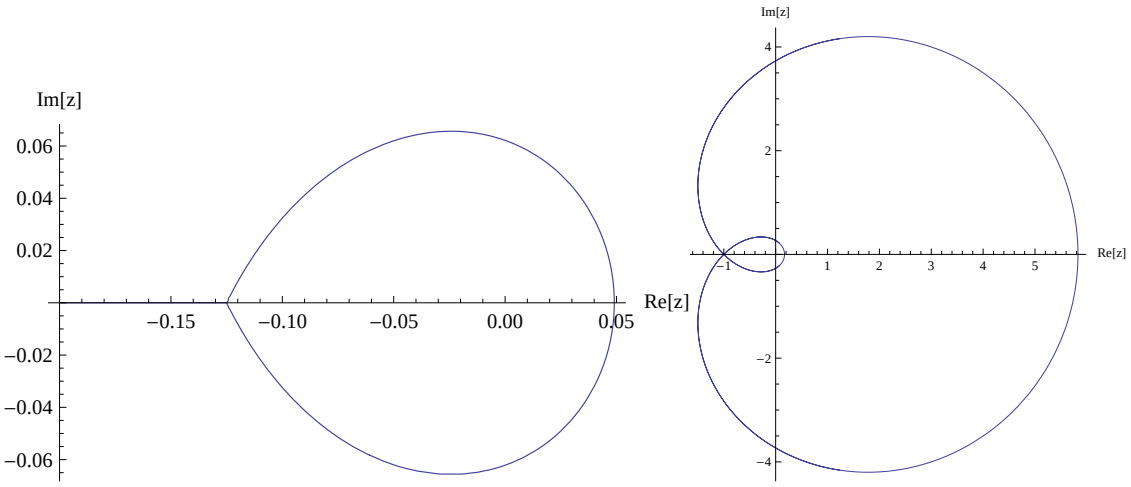

Figure 6: Diagrams of the relevant regions for minimal solutions of recurrence relations 2 (left) and 3 (right).

recurrence relations backwards, and secondly, we use the minimal solutions of the recurrence relations to apply the recurrence relations forwards using Miller's algorithm. A selection of results we obtained by carrying out these two methods is shown in Table 16.

Function desired	Method and function(s) used	Correct solution	Recurrence solution	N
${}_2F_1(55.3, 55.7; 111.6; 0.3)$	Recurrence 1 forward (M) ${}_2F_1(0.3, 0.7; 0.4; 0.7)$	$1.215235338639146 \times 10^4$	$1.215235338639048 \times 10^4$	13
${}_2F_1(40.1, 40.4; 120.7; 0.5 + 0.5i)$	Recurrence 2 forward (M) ${}_2F_1(0.1, 0.4; 0.8; 0.5 + 0.5i)$	$-3.651039314091793 \times 10^2$ $+2.183535248985720 \times 10^2 i$	$-3.651039314091991 \times 10^2$ $+2.183535248985840 \times 10^2 i$	13
${}_2F_1(-59.9, 0.8; 61.2; -0.4)$	Recurrence 3 forward (M) ${}_2F_1(0.9, 0.2; 0.8; -0.4)$	1.474715522389158	1.474715522388619	12
${}_2F_1(0.2, 0.3; 80.9; 0.4)$	Recurrence 4 forward (M) ${}_2F_1(0.2, 0.3; 0.9; 0.4)$	1.00029782004629	1.00029782004027	13
${}_2F_1(0.5, 0.2; -79.3; 0.3 + 0.2i)$	Recurrence 4 backward ${}_2F_1(0.5, 0.2; -0.3; 0.3 + 0.2i)/$ ${}_2F_1(0.5, 0.2; 0.7; 0.3 + 0.2i)$	0.999622417465352 $-0.000250483115171i$	0.999622417465357 $-0.000250483115169i$	11

Table 16: Table showing a selection of functions we wished to compute using recurrence relations, the functions and method we used to compute the solution, the computed solution we obtained and the correct solution generated using MATLAB. The designation (M) in the second column means that Miller's algorithm was used.

The results shown in Table 16 suggest that applying recurrence relations as detailed in this section is a viable way of reducing problems of computing a hypergeometric function with parameters whose real parts have large modulus to a simpler problem of computing one or two hypergeometric functions whose real parts have smaller modulus. However, considering the minimal solutions of recurrence relations 1–4, which are shown in Table 15, we can

see that this method is restricted by MATLAB's inability to compute the Gamma function when its variable has large modulus (in the same way as the recurrence relation techniques of Section 3.8 were restricted when computing ${}_1F_1$). Nevertheless, the methods discussed in this section were found to be useful for carrying out computations of ${}_2F_1$ with arguments of large real part.

4.9 Summary and analysis of results

For the purposes of computing ${}_2F_1(a, b; c; z)$, we have implemented and analysed methods such as Taylor series methods in Section 4.2, the single fraction method in Section 4.3, and quadrature and differential equation methods discussed in Sections 4.4 and 4.5 and Appendix H. Also detailed in Appendix H are other methods that we found to be less effective. In addition, we applied transformations and analytic continuation formulae as in Sections 4.6–4.7 and Appendix A in order to find ways to compute ${}_2F_1$ accurately and efficiently for all $z \in \mathbb{C}$.

We found that the series methods produced accurate results for computing ${}_2F_1$ for certain parameter regimes, specifically for values of $|a|$ and $|b|$ less than 50. We recommend that the single fraction method is used particularly if $|c| < 1$ and $|a|, |b| < 30$. When $\mathrm{Re}(c) > \mathrm{Re}(b) > 0$ or $\mathrm{Re}(c) > \mathrm{Re}(a) > 0$, the Gauss-Jacobi quadrature method is effective. For z inside the unit disc, the RK4 method for numerically solving (4.2) is also seen to be an effective method of computation for cases where $|a|$, $|b|$ and $|c|$ are relatively small (especially for $|a|, |b|, |c| \lesssim 5$). All these methods seem to work well for the parameter values specified and $|z| \lesssim 0.9$.

A problem arises when values of ${}_2F_1$ are required outside the unit disc. On these occasions, the transformation formulae of Section 4.6, or the formulae stated in Appendix A for the special cases $b - a \in \mathbb{Z}$ or $c - a - b \in \mathbb{Z}$ can be applied. A further issue arises when $|\mathrm{Re}(a)|$, $|\mathrm{Re}(b)|$ or $|\mathrm{Re}(c)|$ is too large for a method to work effectively on its own (as a guide, when any of these values exceeds 50). In this case, the recurrence relation techniques of Section 4.8 can be exploited.

As a summary, Table 17 states recommendations as to which of the methods we have researched should be used for certain parameter regimes.

Region of a, b, c, z	Recommended method(s)	Relevant section
$\|a\|, \|b\| < 50$, $\|c\| > 1$, $\|z\| \lesssim 0.9$	Taylor series methods	4.2
	Single fraction method	4.3
	Gauss-Jacobi quadrature, if $\mathrm{Re}(c) > \mathrm{Re}(b) > 0$	4.4
	RK4 method, if $\|a\|, \|b\|, \|c\| < 5$	4.5
$\|a\|, \|b\| < 30$, $\|c\| < 1$, $\|z\| \lesssim 0.9$	Single fraction method	4.3
	RK4 method, if $\|a\|, \|b\| < 5$	4.5
$\|a\|, \|b\| < 50$, $z \approx e^{\pm i\pi/3}$	Analytic continuation	4.7
$\|a\|, \|b\| < 50$, $\|z\| \geq 0.9$	Transformation methods	4.6, Appendix A
$\|\mathrm{Re}(a)\|$, $\|\mathrm{Re}(b)\|$ or $\|\mathrm{Re}(c)\| > 50$	Recurrence relations	4.8
	then another method	

Table 17: Recommendations as to methods that should be used for the computation of the Gauss hypergeometric function for different parameter regimes and the sections where they are discussed.

One major drawback was the computation of ${}_2F_1$ for large values of $|\mathrm{Im}(a)|$ or $|\mathrm{Im}(b)|$. Unlike for large $|\mathrm{Re}(a)|$ or $|\mathrm{Re}(b)|$, the techniques of computing recurrence relations cannot be exploited, so further work on this problem will involve finding effective methods for these parameter regimes. Research into transformations, analytic continuation formulae and recurrence relations in particular was hampered by MATLAB's lack of a Gamma function package that can deal with complex or large real variable input, so it would be useful to devise a routine to do this to advance our knowledge of computing ${}_2F_1$. Details of how this could be done are given in Appendix I.

5 Conclusions, Discussion and Future Considerations

Computing the hypergeometric function ${}_pF_q$ is an important problem due to its wide variety of applications in problems in mathematical and theoretical physics, networks, finance, and many other areas. However, as explained in Section 2.2, it is a difficult problem in practice, and the state-of-the-art software has significant drawbacks. It is therefore important to conduct research into the computation of this class of functions, and provide recommendations as to which methods are useful in order to overcome these problems.

In this project, we researched and implemented a large number of methods for computing the confluent and Gauss hypergeometric functions ${}_1F_1$ and ${}_2F_1$, the two most commonly used hypergeometric functions. These methods have come from a wide range of areas in numerical

analysis, such as series computations, quadrature, numerical solution of differential equations and recurrence relations. We can conclude that for both of these functions, there is no single method that is optimal for their computation for all parameter and variable values; instead a satisfactory package for computing hypergeometric functions will involve making use of a variety of methods, as each is most effective for a specific parameter regime.

As detailed in Section 3.9, the most effective methods for computing ${}_1F_1(a;b;z)$ for smaller values of $|a|$ and $|z|$ involve direct computation of the power series, either by one of the two methods for computing the Taylor series detailed in Section 3.2, or by expressing ${}_1F_1$ as a single fraction as in Section 3.3, the latter being more effective for small $|b|$. Applying Gauss-Jacobi quadrature is also an effective method for $\mathrm{Re}(b) > \mathrm{Re}(a) > 0$, as is solving the confluent hypergeometric differential equation (3.2) provided $\mathrm{Re}(z)$ and $\mathrm{Im}(z)$ are sufficiently close to zero, and $|a|$ and $|b|$ are close to zero. For cases where $\mathrm{Re}(a)$ and $\mathrm{Re}(z)$ are moderately large and of opposite sign, we conclude that using the expression (3.20) as in Section 3.4 is very effective. For more difficult cases where $|\mathrm{Re}(a)|$ or $|\mathrm{Re}(b)|$ are greater than about 50, we can use the techniques of computing recurrence relations introduced in Section 3.8 to reduce the problem to one where this is not the case, and apply another method to the simpler problem.

For the problem of computing ${}_2F_1(a,b;c;z)$, one can similarly use Taylor series methods and the method of expressing ${}_2F_1$ as a single fraction, for regimes where $|a|$, $|b|$ and $|z|$ are comparatively small, as detailed in Sections 4.2 and 4.3, as well as quadrature methods (when $\mathrm{Re}(c) > \mathrm{Re}(b) > 0$) and solving the hypergeometric differential equation (4.2) numerically. A combination of these methods will compute ${}_2F_1$ accurately in most of the unit disc; in order to carry out computation within the remainder of the unit disc and outside it, we may use analytic continuation formulae as detailed in Section 4.7 (for computation near $z = e^{\pm i\pi/3}$) and transformation formulae detailed in Section 4.6 and Appendix A. When $|\mathrm{Re}(a)|$, $|\mathrm{Re}(b)|$ or $|\mathrm{Re}(c)|$ exceed about 50, the recurrence relations techniques can be applied, as for ${}_1F_1$. All the methods recommended for ${}_2F_1$, as well as ${}_1F_1$, have much faster computation times than the built-in MATLAB routine 'hypergeom'.

We note that in order to make full use of the variety of methods available, it would be useful to implement them using higher precision software, as this should generate accurate results due to reduced effects of round-off error and overflow. We also recommend to the

Numerical Algorithms Group that software be designed for computing the Gamma function $\Gamma(z)$, incomplete gamma functions $\gamma(a, z)$ and $\Gamma(a, z)$ and Bessel function $J_\nu(z)$ for all parameter and variable values in the complex plane, as these were required for a large number of the methods tested, and will be required to compute both the confluent and Gauss hypergeometric functions with complex variable and parameters. We present ideas as to which methods could be included in such software in Appendix I.

Apart from implementing the methods we have examined on higher precision software, the major further work that one could undertake is handling parameter regimes where methods discussed in this dissertation did not work universally. One important example of such a regime is when there are large imaginary parts of the parameters a, b (and c for ${}_2F_1$), for which, unlike parameter values with large real parts, recurrence relation techniques cannot be applied. Another regime for ${}_1F_1$ that would merit further investigation is the case of large $\mathrm{Im}(z)$ when computing ${}_1F_1$; this was observed to be particularly vulnerable to round-off error and other computational issues when the methods discussed in this project were applied to it. Other further work in this wide subject area could involve applying the theory and methods we have presented in this dissertation, and investigating new methods, to tackle the problem of computing other hypergeometric functions, for example ${}_2F_2$ and ${}_3F_2$. The techniques discussed could also be applied to devise a package for the effective computation of a variety of other special functions with important practical applications.

A Transformation formulae for ${}_2F_1(a,b;c;z)$ when $b-a \in \mathbb{Z}$ or $c-a-b \in \mathbb{Z}$

This appendix provides information on research that aims to tackle a major computational issue that occurs when trying to apply the transformation formulae in Table 13 of Section 4.6 when either $b-a \in \mathbb{Z}$ or $c-a-b \in \mathbb{Z}$. The issue arises due to the fact that the Gamma function $\Gamma(x)$ has a singularity when x is equal to a non-positive integer, so when applying the transformation formulae (4.16) and (4.20) (equivalent to transformations 1 and 6 respectively in Table 13) for $b-a \in \mathbb{Z}$, or (4.18) and (4.19) (equivalent to transformations 4 and 5 respectively in Table 13) for $c-a-b \in \mathbb{Z}$, the sum of the two terms of the tranformations is finite, but each individual term is infinite. Transformations 2 and 3 from Section 4.6 do not require this theory, as the second transformation, involving the variable $\frac{z}{z-1}$ of ${}_2F_1$, does not entail the computation of any Gamma functions and the third transformation simply maps the variable z to itself.

In [3, 22], formulae are provided that avoid this issue when either $b-a$ or $c-a-b$ is exactly equal to an integer. These may be computed using the same ideas detailed in Section 3.2. The appropriate formulae are stated below:

- When $b-a=m \in \mathbb{Z}^+ \cup \{0\}$, then an expression that serves the same purpose as (4.16) (namely computing ${}_2F_1$ in terms of variable $\frac{1}{1-z}$ rather than z) is given by

$$\begin{aligned} {}_2F_1(a,b;c;z) = {} & \frac{(1-z)^{-a}}{\Gamma(b)\Gamma(c-a)} \sum_{j=0}^{m-1} \frac{(a)_j (c-b)_j (m-j-1)!}{j!} (z-1)^{-j} \\ & + \frac{(-1)^m (1-z)^{-b}}{\Gamma(a)\Gamma(c-b)} \sum_{j=0}^{\infty} \frac{(b)_j (c-a)_j}{j!(j+m)!} (1-z)^{-j} \\ & \times [\log(1-z) + \psi(j+1) + \psi(j+m+1) - \psi(b+j) - \psi(c-a-j)], \end{aligned} \tag{A.1}$$

  for $|z-1|>1$, $|\arg(1-z)|<\pi$, where $\psi(y)$ is defined in (4.26).

  If $b-a=-m \in \mathbb{Z}^-$, the parameters a and b are exchanged, and, using the fact that ${}_2F_1(a,b;c;z) = {}_2F_1(b,a;c;z)$ by the basic series definition, (A.1) is again used.

- When $c-a-b=m\in\mathbb{Z}^+\cup\{0\}$, then an alternative formulation for (4.18) is given by

$$
\begin{aligned}
{}_2F_1(a,b;c;z) = {} & \frac{1}{\Gamma(c-a)\Gamma(c-b)}\sum_{j=0}^{m-1}\frac{(a)_j(b)_j(m-j-1)!}{j!}(z-1)^j \\
& +\frac{(z-1)^{c-a-b}}{\Gamma(a)\Gamma(b)}\sum_{j=0}^{\infty}\frac{(c-a)_j(c-b)_j}{j!(j+m)!\Gamma(c-b-j)}(-1)^j z^{-j-m} \qquad \text{(A.2)} \\
& \times[\log(1-z)-\psi(j+1)-\psi(j+m+1)+\psi(c-a+j)+\psi(c-b-j)],
\end{aligned}
$$

for $|z|<1,\ |\arg(1-z)|<\pi$. If $c-a-b=-m\in\mathbb{Z}^-$, the transformation (4.4) can be used to write

$$
\begin{aligned}
{}_2F_1(a,b;a+b-m;z) &= (1-z)^{c-a-b}{}_2F_1(c-a,c-b;c;z) && \text{(A.3)} \\
&= (1-z)^{c-a-b}{}_2F_1(b-m,a-m;a+b-m), && \text{(A.4)}
\end{aligned}
$$

at which point (A.2) can be applied using the new parameters $b-m$, $a-m$ and $a+b-m$.

- When $c-a-b=m\in\mathbb{Z}^+\cup\{0\}$, then the equivalent formulation for (4.19) is given by

$$
\begin{aligned}
{}_2F_1(a,b;c;z) = {} & \frac{z^{-a}}{\Gamma(c-b)}\sum_{j=0}^{m-1}\frac{(a)_j(m-j-1)!}{j!\Gamma(c-a-j)}\left(1-\frac{1}{z}\right)^j \\
& +\frac{(z)^{-a}}{\Gamma(a)}\sum_{j=0}^{\infty}\frac{(c-b)_j}{j!(j+m)!\Gamma(b-j)}(-1)^j\left(1-\frac{1}{z}\right)^{j+m} \qquad \text{(A.5)} \\
& \times\left[\log\left(\frac{1-z}{z}\right)-\psi(j+1)-\psi(j+m+1)+\psi(b-j)+\psi(c-b-j)\right],
\end{aligned}
$$

for $\mathrm{Re}(z)>\frac{1}{2},\ |\arg z|<\pi,\ |\arg(1-z)|<\pi$. If $c-a-b=-m\in\mathbb{Z}^-$, the transformation described by (A.3) and (A.4) should again be used before applying (A.5) with the new parameters $b-m$, $a-m$ and $a+b-m$.

- When $b-a=m\in\mathbb{Z}^+\cup\{0\}$, then the equivalent formulation for (4.20) is given by:

$$
\begin{aligned}
{}_2F_1(a,b;c;z) = {} & \frac{(-z)^{-a}}{\Gamma(b)}\sum_{j=0}^{m-1}\frac{(a)_j(m-j-1)!}{j!\Gamma(c-a-j)}z^{-j} \\
& +\frac{(-z)^{-a}}{\Gamma(a)}\sum_{j=0}^{\infty}\frac{(b)_j}{j!(j+m)!\Gamma(c-b-j)}(-1)^j z^{-j-m} \qquad \text{(A.6)} \\
& \times[\log(-z)+\psi(j+1)+\psi(j+m+1)-\psi(b+j)-\psi(c-b-j)],
\end{aligned}
$$

for $|z| > 1$, $|\arg(-z)| < \pi$. As above, if $b-a = -m \in \mathbb{Z}^-$, the parameters a and b are exchanged, and (A.6) is again used.

There is also a numerical issue when either $a-b$ or $c-a-b$ is close to an integer (meaning the real part is close to an integer and the imaginary part is close to 0), as the two terms in (4.16), (4.18), (4.19) or (4.20) are both large due to the presence of Gamma functions with variables close to negative integers. Therefore alternative expressions are required to avoid large round-off error. Expressions for values of $b-a$ or $c-a-b$ close to an integer are stated in [24] and detailed below:

- An alternative formulation for (4.16) in Section 4.6, if $a-b = k+\epsilon$ with $k \in \mathbb{Z}^+ \cup \{0\}$, and $|\epsilon|$ small, reads as follows:

$$
\begin{aligned}
{}_2F_1(a,b;c;z) = {} & \frac{\Gamma(c)}{\Gamma(a)\Gamma(c-b)} \sum_{j=0}^{k-1} \frac{(b)_j (c-a)_j \Gamma(k-j+\epsilon)(-1)^j}{j!} \left(\frac{1}{1-z}\right)^{b+j} \\
& + \Gamma(c) \sum_{j=0}^{\infty} \frac{\Gamma(a+j)\Gamma(c-a+k+j+\epsilon)\Gamma(-k-j-\epsilon)(-1)^j}{\Gamma(a)\Gamma(b)\Gamma(c-a)\Gamma(c-b)j!} \left(\frac{1}{1-z}\right)^{a+j} \\
& + \Gamma(c) \sum_{j=0}^{\infty} \frac{\Gamma(a+j-\epsilon)\Gamma(c-a+k+j)\Gamma(-j+\epsilon)(-1)^{j+k}}{\Gamma(a)\Gamma(b)\Gamma(c-a)\Gamma(c-b)(j+k)!} \left(\frac{1}{1-z}\right)^{a+j-\epsilon}.
\end{aligned}
\tag{A.7}
$$

  If $\mathrm{Re}(a-b) < 0$, then the parameters a and b are again exchanged before (A.7) is applied.

- An alternative formulation for (4.18) in Section 4.6, if $c-a-b = k+\epsilon$ with $k \in \mathbb{Z}^+ \cup \{0\}$ and $|\epsilon|$ small, is given by

$$
\begin{aligned}
{}_2F_1(a,b;c;z) = {} & \frac{\Gamma(c)}{\Gamma(a)\Gamma(c-b)} \sum_{j=0}^{k-1} \frac{(a)_j (b)_j \Gamma(k+\epsilon-j)(-1)^j}{j!} (1-z)^j \\
& + \Gamma(c) \sum_{j=0}^{\infty} \frac{\Gamma(a+j)\Gamma(c-a+k+j+\epsilon)\Gamma(-k-j-\epsilon)(-1)^j}{\Gamma(a)\Gamma(b)\Gamma(c-a)\Gamma(c-b)j!} (1-z)^{a+j} \\
& + \Gamma(c) \sum_{j=0}^{\infty} \frac{\Gamma(a+j+k)\Gamma(b+j+k)\Gamma(\epsilon-j)(-1)^{j+k}}{\Gamma(a)\Gamma(b)\Gamma(c-a)\Gamma(c-b)(j+k)!} (1-z)^{a+j-\epsilon}.
\end{aligned}
\tag{A.8}
$$

On the other hand, if $c-a-b=-k+\epsilon$ with $k\in\mathbb{Z}^+$, then (4.18) becomes

$$
\begin{aligned}
{}_2F_1(a,b;c;z) = {} & \frac{\Gamma(c)}{\Gamma(a)\Gamma(b)}\sum_{j=0}^{k-1}\frac{(c-a)_j(c-b)_j\Gamma(k-\epsilon-j)(-1)^j}{j!}(1-z)^{j-k+\epsilon} \\
& +\Gamma(c)\sum_{j=0}^{\infty}\frac{\Gamma(a+j+\epsilon)\Gamma(b+j+\epsilon)\Gamma(-j-\epsilon)(-1)^{j+k}}{\Gamma(a)\Gamma(b)\Gamma(c-a)\Gamma(c-b)(j+k)!}(1-z)^{j+\epsilon} \\
& +\Gamma(c)\sum_{j=0}^{\infty}\frac{\Gamma(a+j)\Gamma(b+j)\Gamma(\epsilon-j-k)(-1)^{j}}{\Gamma(a)\Gamma(b)\Gamma(c-a)\Gamma(c-b)j!}(1-z)^{j}.
\end{aligned}
\tag{A.9}
$$

- If $c-a-b=k+\epsilon$ with $k\in\mathbb{Z}^+\cup\{0\}$, and $|\epsilon|$ is small, then the equivalent expression for (4.19) in Section 4.6 is given by

$$
\begin{aligned}
& \lim_{\delta\to 0^+}{}_2F_1(a,b;c;z+i\delta) \\
& \quad = z^{-a}\Gamma(c)\times\left[\sum_{j=0}^{k-1}\frac{(a)_j(1+a-c)_j\Gamma(k-j+\epsilon)(-1)^j}{\Gamma(c-a)\Gamma(c-b)j!}\left(1-\frac{1}{z}\right)^j\right. \\
& \qquad +\sum_{j=0}^{\infty}\frac{(a)_{j+k}(1+a-c)_{j+k}\Gamma(\epsilon-j)(-1)^{j+k}}{\Gamma(c-a)\Gamma(c-b)(j+k)!}\left(1-\frac{1}{z}\right)^{j+k} \\
& \qquad \left.+e^{-i\pi(k+\epsilon)}\sum_{j=0}^{\infty}\frac{(1-b)_j(c-b)_j\Gamma(-k-j-\epsilon)(-1)^j}{\Gamma(a)\Gamma(b)j!}\left(1-\frac{1}{z}\right)^{j+k+\epsilon}\right].
\end{aligned}
\tag{A.10}
$$

If $c-a-b=-k+\epsilon$, when $k\in\mathbb{Z}^+$ and $|\epsilon|$ is small, the formula can instead be written as:

$$
\begin{aligned}
& \lim_{\delta\to 0^+}{}_2F_1(a,b;c;z+i\delta) \\
& \quad = z^{-a}\Gamma(c)\times\left[e^{i\pi(k-\epsilon)}\sum_{j=0}^{k-1}\frac{(1-b)_j(c-b)_j\Gamma(k-j-\epsilon)(-1)^j}{\Gamma(a)\Gamma(b)j!}\left(1-\frac{1}{z}\right)^{j-k+\epsilon}\right. \\
& \qquad +e^{i\pi(k-\epsilon)}\sum_{j=0}^{\infty}\frac{(1-b)_{j+k}(c-b)_{j+k}\Gamma(-\epsilon-j)(-1)^{j+k}}{\Gamma(a)\Gamma(b)(j+k)!}\left(1-\frac{1}{z}\right)^{j+\epsilon} \\
& \qquad \left.+z^{-a}\Gamma(c)\sum_{j=0}^{\infty}\frac{(a)_j(a-c+1)_j\Gamma(\epsilon-k-j)(-1)^j}{\Gamma(a-k+\epsilon)\Gamma(b-k+\epsilon)j!}\left(1-\frac{1}{z}\right)^{j}\right].
\end{aligned}
\tag{A.11}
$$

- Finally, an alternative formulation for (4.20) in Section 4.6, if $a - b = k + \epsilon$ where $k \in \mathbb{Z}^+ \cup \{0\}$ and $|\epsilon|$ is small, reads as follows:

$$\lim_{\delta \to 0^+} {}_2F_1(a, b; c; z + i\delta) = \Gamma(c)e^{i\pi b} \sum_{j=0}^{k-1} \frac{(b-c+1)_j (b)_j \Gamma(k+\epsilon-j)(-1)^j}{\Gamma(a)\Gamma(c-b)j!} \left(\frac{1}{z}\right)^{j+b} \tag{A.12}$$
$$+\Gamma(c)e^{i\pi a} \sum_{j=0}^{\infty} \frac{(a)_j (a-c+1)_j \Gamma(-k-\epsilon-j)(-1)^j}{\Gamma(b)\Gamma(c-a)j!} \left(\frac{1}{z}\right)^{j+a}$$
$$+\Gamma(c)e^{i\pi b} \sum_{j=0}^{\infty} \frac{(b)_{j+k} (b-c+1)_{j+k} \Gamma(\epsilon-j)(-1)^{j+k}}{\Gamma(a)\Gamma(c-b)(j+k)!} \left(\frac{1}{z}\right)^{j+k+b}.$$

  Again, if $\mathrm{Re}(a - b) < 0$, then the parameters a and b are exchanged before (A.12) is applied.

B List of test cases used for ${}_1F_1(a;b;z)$

We show in Table 18 the 40 test cases that represent the range of problems that, as a result of testing the methods and carrying out a literature review, are deemed to be likely to cause code written for the purpose of computing ${}_1F_1(a;b;z)$ to fail. This could be due to excessive round-off error or cancellation, the fact that the case represents a particular instance or regime for which the method being tested will not work, or the fact that the method is not sufficient for any regime. The aim is to find, for each test case, a method that computes it quickly and accurately. The comments column explains the properties of each case that motivate its place on the list.

Shown next to a number of cases is the paper that influenced the choice of test case. Test cases 9, 10 and 11 were taken from [44], the discussion in [2] inspired the choice of test cases 17 and 18, and the combination of values of b and z (although not the value of a) in cases 26 and 27 were taken from [34].

Case	a	b	z	Comments
1	0.1	0.2	0.5	Basic case, positive a, with $b = 2a$
2	-0.1	0.2	0.5	Basic case, negative a
3	0.1	0.2	$-0.5 + i$	Basic case, complex z, with $b = 2a$
4	$1 + i$	$1 + i$	$1 - i$	Complex a, b, z
5	10^{-8}	10^{-8}	10^{-10}	Very small parameters and variable
6	10^{-8}	10^{-12}	$-10^{-10} + 10^{-12}i$	Very small complex variable
7	1	1	$10 + 10^{-9}i$	Larger variable with small imaginary part, and $a = b$
8	1	3	10	$b > a > 0$ and larger z
9	500	511	10	Large $b > a > 0$ [44]
10	8.1	10.1	100	Larger z, with $b > a > 0$ [44]
11	1	2	600	Very large z, with $b = 2a$ [44]
12	100	1.5	2.5	Large positive a
13	-60	1	10	Large negative a, positive z
14	60	1	10	Large positive a, positive z
15	60	1	-10	Large positive a, negative z
16	-60	1	-10	Large negative a, negative z
17	1000	1	10^{-3}	Very large $a > 0$, small $z > 0$, $z = \frac{1}{a}$ [2]
18	10^{-3}	1	700	Very large $z > 0$, small $a > 0$ [2]
19	500	1	-5	Very large $a > 0$, $z < 0$
20	-500	1	5	Very large $a < 0$, $z > 0$
21	20	$-10 + 10^{-9}$	-2.5	b close to positive integer, $z < 0 < a$
22	20	$10 - 10^{-9}$	2.5	b close to positive integer, $a, z > 0$
23	-20	$-10 + 10^{-12}$	2.5	a negative integer, b close to negative integer
24	50	10	$200i$	Very large, purely imaginary z
25	-5	$(-5 + 10^{-9}) + (-5 + 10^{-9})i$	-1	b with real and imaginary parts close to negative integer
26	4	80	200	Large b and larger z [34]
27	-4	500	300	Very large z and larger b [34]
28	5	0.1	$-2 + 300i$	Very large imaginary part of z, negative real part
29	-5	0.1	$2 + 300i$	Very large imaginary part of z, positive real part
30	$2 + 8i$	$-150 + i$	150	Large values of $\mathrm{Re}(b) < 0 < \mathrm{Re}(z)$
31	5	2	$100 - 1000i$	Large values of z in fourth quadrant of complex plane
32	-5	2	$-100 + 1000i$	Large real and imaginary parts of z in second quadrant of complex plane
33	-5	$-2 - i$	$1 + (2 - 10^{-10})i$	Complex b and z, with mixed signs
34	1	10^{-12}	1	Very small b
35	10	10^{-12}	10	Very small b with larger a and z
36	1	$-1 + 10^{-12}i$	1	Very small imaginary part of b
37	1000	1	-1000	Very large positive real a and very large negative real z, with $a = -z$
38	-1000	1	1000	Very large negative real a and very large positive real z, with $a = -z$
39	$-10 + 500i$	$5i$	10	Large imaginary part of a, imaginary b
40	20	$10 + 1000i$	-5	Large imaginary part of b

Table 18: List of test cases used for ${}_1F_1(a; b; z)$.

C List of test cases used for ${}_2F_1(a, b; c; z)$

We show in Table 19 the 30 test cases written to represent the range of problems that, as a result of testing the methods and carrying out a literature review, are deemed to be likely to cause code written for the purpose of computing ${}_2F_1(a, b; c; z)$ to fail. As is the case for ${}_1F_1(a; b; z)$, this could be due to excessive round-off error or cancellation, the fact that the case represents a particular instance or regime for which the method being tested will not work, or the fact that the method is not sufficient for any regime. Again, the aim is to find, for each test case, a method that computes it quickly and accurately. The comments column explains the properties of each case that motivate its place on the list.

Shown next to test case 30 is the paper from which it was taken.

Case	a	b	c	z	Comments
1	0.1	0.2	0.3	0.5	Basic case with $b = 2a$, $c = a + b$
2	-0.1	0.2	0.3	0.5	Basic case with $c = b - a$
3	0.1	0.2	-0.3	$-0.5 + 0.5i$	Basic case, complex z, with $b = 2a$, $c = -a - b$
4	10^{-8}	10^{-8}	10^{-8}	10^{-6}	Very small parameters and variable
5	10^{-8}	-10^{-6}	10^{-12}	$-10^{-10} + 10^{-12}i$	Very small complex variable
6	1	10	1	$0.5 + 10^{-9}i$	Variable with $\mathrm{Re}(z) \gg \mathrm{Im}(z)$
7	1	$-1 + 10^{-12}i$	1	-0.8	Parameter b differs from a negative integer by very small imaginary part
8	$2 + 8i$	$3 - 5i$	$\sqrt{2} - \pi i$	0.75	All parameters imaginary
9	100	200	350	i	z on unit disc, large real parameters
10	$2 + 10^{-9}$	3	5	-0.75	Real a close to positive integer, negative z
11	-2	-3	$-5 + 10^{-9}$	0.5	c close to negative integer, real $a, b < 0$
12	-1	-1.5	$-2 - 10^{-15}$	0.5	c close to negative integer, real $b < a < 0$
13	500	-500	500	0.75	Large real $a = c > 0$, $b < 0$, ${}_2F_1(a, b; c; z) = {}_1F_0(b;\ ; z)$
14	500	500	500	-0.6	Large real $a = b = c > 0$, ${}_2F_1(a, b; c; z) = {}_1F_0(a;\ ; z)$
15	-1000	-2000	-4000.1	-0.5	Very large negative $c < b < a < 0$
16	-100	-200	$-300 + 10^{-9}$	$0.5\sqrt{2}$	Large negative real parameters, c close to negative integer
17	300	10	5	0.5	Large real $a > 0$
18	5	-300	10	0.5	Large real $b < 0$
19	10	5	-300.5	0.5	Large real $c < 0$
20	$2 + 200i$	5	10	0.6	a with large positive imaginary part
21	$2 + 200i$	$5 - 100i$	$10 + 500i$	0.8	Parameters all with large imaginary parts
22	2	5	$10 - 500i$	-0.8	Large negative imaginary part of c
23	2.25	3.75	-0.5	-1	Special case of ${}_2F_1(a, b; 1 + a - b; -1)$ $= \frac{\Gamma(1+a-b)\Gamma(1+a/2)}{\Gamma(1+a)\Gamma(1+a/2+b/2)}$
24	1	2	$4 + 3i$	$0.6 - 0.8i$	z on unit disc with $\mathrm{Re}(c - a - b) > 0$
25	1	0.9	2	$e^{i\pi/3}$	$z = e^{i\pi/3}$ with $\mathrm{Re}(c - a - b) > 0$
26	1	1	4	$e^{i\pi/3}$	$z = e^{i\pi/3}$ with $a = b$, c positive integers
27	-1	0.9	2	$e^{-i\pi/3}$	$z = e^{-i\pi/3}$ with real $a < 0$, $\mathrm{Re}(c - a - b) > 0$
28	4	1.1	2	$0.5 + (0.5\sqrt{3} - 0.01)i$	z near $e^{-i\pi/3}$, different imaginary part
29	5	2.2	-3	$0.49 + 0.5\sqrt{3}i$	z near $e^{-i\pi/3}$, different real part
30	$\frac{2}{3}$	1	$\frac{4}{3}$	$e^{i\pi/3}$	$z = e^{i\pi/3}$, known value is $\frac{2\pi e^{i\pi/6}\Gamma(1/3)}{9[\Gamma(2/3)]^2}$ [28]

Table 19: List of test cases used for ${}_2F_1(a, b; c; z)$.

D Methods of testing the robustness of code selected

There are a number of different *a posteriori* error tests that can be applied to test the accuracy and robustness of the code selected to compute the hypergeometric functions ${}_1F_1$ and ${}_2F_1$ for a specific parameter regime. We carried out a variety of such tests on the code that was found to be comparatively robust. We recommend these tests, which are detailed below, to any programmer who wishes to test a routine for the computation of a hypergeometric function for robustness:

- We tested each method for computing ${}_1F_1$ and ${}_2F_1$ that was fairly robust on each test case, as detailed in Appendices B, C, E and F. These test cases were devised on the basis of a literature review and extensive testing of each method implemented and are intended to represent a list of the classes of problems likely to find a flaw in a routine for computing the particular hypergeometric function. Reasons are given in Appendices B and C for why each case is potentially troublesome for the code, and reference is made to any literature that directly led to the opinion that the case might be difficult. The true solutions for each of the cases are found using Mathematica and verified using MATLAB, and are given to 16 significant figures.

- Methods deemed to be effective for specific parameter regimes were extensively tested on tabulated values from [64, 70] for ${}_1F_1$ and [65] for ${}_2F_1$.

- Effective code for each parameter regime was tested on cases based on the following known relations and values [3, 7, 37]:

$$
\begin{aligned}
M(1;2;2z) &= \frac{e^z}{z}\sinh z,\\
M(a;a+1;-z) &= az^{-a}\gamma(a,z),\\
M\left(\frac{1}{2};\frac{3}{2};-z^2\right) &= \frac{\sqrt{\pi}}{2z}\mathrm{erf}(z),\\
M\left(a+\frac{1}{2};2a+1;2z\right) &= \Gamma(1+a)e^z\left(\frac{z}{2}\right)^{-a}I_a(z),\\
M\left(-n;\frac{1}{2};z^2\right) &= (-1)^n\frac{n!}{(2n)!}H_{2n}(z),\\
{}_2F_1(1,1;2;-z) &= \frac{1}{z}\log(1+z),\\
{}_2F_1\left(1,\frac{1}{2};\frac{3}{2};-z^2\right) &= \frac{1}{z}\tan^{-1}z,\\
{}_2F_1\left(a,1-a;b;\frac{1}{2}\right) &= \frac{2^{1-b}\sqrt{\pi}\Gamma(b)}{\Gamma\left(\frac{1}{2}a+\frac{1}{2}b\right)\Gamma\left(\frac{1}{2}b-\frac{1}{2}a+\frac{1}{2}\right)},\\
{}_2F_1\left(-n,n;\frac{1}{2};\frac{1-z}{2}\right) &= T_n(z),\\
{}_2F_1\left(-n,n+1;1;\frac{1-z}{2}\right) &= P_n(z).
\end{aligned}
$$

  In the relations above, the **Hermite polynomials** $H_n(z)$ for $n \in \mathbb{Z}^+ \cup \{0\}$ are defined as $(-1)^n e^{z^2/2}\frac{\mathrm{d}^n}{\mathrm{d}z^n}\left(\frac{1}{2}e^{-z^2}\right)$, the **Chebyshev polynomials** $T_n(z)$ for $n \in \mathbb{Z}^+ \cup \{0\}$ are defined by $T_0(z) = 1$, $T_1(z) = z$, $T_{n+1}(z) = 2zT_n(z) - T_{n-1}(z)$ for $n = 1, 2, ...$ (or as $T_n(z) = \cos(n\cos^{-1}z)$), and the **Legendre polynomials** $P_n(z)$ are defined by $P_n(z) = \frac{1}{2^n n!}\frac{\mathrm{d}^n}{\mathrm{d}z^n}\left[(z^2-1)^n\right]$).

- The numerical results for the most robust routines for $M(a;b;z)$ obtained were tested on the following known recurrence relations from [3], which provide a good *a posteriori* test for the accuracy of a computation:

$$\begin{aligned}
(b-a)M(a-1;b;z)+(2a-b+z)M(a;b;z)-aM(a+1;b;z)&=0,\\
b(b-1)M(a;b-1;z)+b(1-b-z)M(a;b;z)+z(b-a)M(a;b+1;z)&=0,\\
(a-b+1)M(a;b;z)-aM(a+1;b;z)+(b-1)M(a;b-1;z)&=0,\\
bM(a;b;z)-bM(a-1;b;z)-zM(a;b+1;z)&=0,\\
b(a+z)M(a;b;z)+z(a-b)M(a;b+1;z)-abM(a+1;b;z)&=0,\\
(a-1+z)M(a;b;z)+(b-a)M(a;b+1;z)+(1-b)M(a;b-1;z)&=0,
\end{aligned}$$

as well as the following recurrence relation, which can be shown to be equivalent to the differential equation (3.2) for $M(a;b;z)$ by applying (2.7):

$$(a+1)zM(a+2;b+2;z)+(b+1)(b-z)M(a+1;b+1;z)-b(b+1)M(a;b;z)=0.$$

Robust routines devised for computing ${}_2F_1(a,b;c;z)$ were tested on the following recurrence relations, stated in [3, 15, 17]:

$$\begin{aligned}
&(c-a)\ {}_2F_1(a-1,b;c;z)+(2a-c+(b-a)z)\ {}_2F_1(a,b;c;z)\\
&\qquad\qquad+a(z-1)\ {}_2F_1(a+1,b;c;z)=0,\\
&(b-a)\ {}_2F_1(a,b;c;z)+a\ {}_2F_1(a+1,b;c;z)-b\ {}_2F_1(a,b+1;c;z)=0,\\
&(c-a-b)\ {}_2F_1(a,b;c;z)+a(1-z)\ {}_2F_1(a+1,b;c;z)\\
&\qquad\qquad-(c-b)\ {}_2F_1(a,b-1;c;z)=0,\\
&c(a+(b-c)z)\ {}_2F_1(a,b;c;z)-ac(1-z)\ {}_2F_1(a+1,b;c;z)\\
&\qquad\qquad+(c-a)(c-b)z\ {}_2F_1(a,b;c+1;z)=0,\\
&(c-a-1)\ {}_2F_1(a,b;c;z)+a\ {}_2F_1(a+1,b;c;z)-(c-1)\ {}_2F_1(a,b;c-1;z)=0,\\
&c(1-z)\ {}_2F_1(a,b;c;z)-c\ {}_2F_1(a-1,b;c;z)+(c-b)z\ {}_2F_1(a,b;c+1;z)=0,\\
&(a-1+(b+1-c)z)\ {}_2F_1(a,b;c;z)+(c-a)\ {}_2F_1(a-1,b;c;z)\\
&\qquad\qquad-(c-1)(1-z)\ {}_2F_1(a,b;c-1;z)=0,\\
&c(c-1)(z-1)\ {}_2F_1(a,b;c-1;z)+c(c-1-(2c-a-b-1)z)\ {}_2F_1(a,b;c;z)\\
&\qquad\qquad+(c-a)(c-b)z\ {}_2F_1(a,b;c+1;z)=0,
\end{aligned}$$

as well as the following recurrence relation, which by (2.7) is equivalent to the differential equation (4.2) for ${}_2F_1(a,b;c;z)$:

$$z(1-z)(a+1)(b+1){}_2F_1(a+2,b+2;c+2;z) \\ +[c-(a+b+1)z](c+1){}_2F_1(a+1,b+1;c+1;z)-c(c+1){}_2F_1(a,b;c;z)=0.$$

- The solutions were tested on the following Wronskians for $M(a;b;z)$ and ${}_2F_1(a,b;c;z)$, from [36, 67]:

$$\mathscr{W}\{M(a;b;z),U(a;b;z)\} \\ =-\frac{\Gamma(b)z^{-b}e^z}{\Gamma(a)},$$

$$\mathscr{W}\{{}_2F_1(a,b;c;z),z^{1-c}\ {}_2F_1(a-c+1,b-c+1;2-c;z)\} \\ =(1-c)z^{-c}(1-z)^{c-a-b-1},$$

$$\mathscr{W}\{{}_2F_1(a,b;a+b+1-c;1-z),(1-z)^{c-a-b}\ {}_2F_1(c-a,c-b;c-a-b+1;1-z)\} \\ =(a+b-c)z^{-c}(1-z)^{c-a-b-1},$$

$$\mathscr{W}\{z^{-a}\ {}_2F_1(a,a-c+1;a-b+1;\frac{1}{z}),z^{-b}\ {}_2F_1(b,b-c+1;b-a+1;\frac{1}{z})\} \\ =(a-b)z^{-c}(z-1)^{c-a-b-1},$$

where the Wronskian of two functions $g_1(z)$, $g_2(z)$ is defined as

$$\mathscr{W}\{g_1(z),g_2(z)\}=g_1(z)g_2'(z)-g_1'(z)g_2(z).$$

E Numerical results for ${}_1F_1(a;b;z)$

Case	(a,b,z)	Correct $M(a;b;z)$	Taylor (a) ($tol = 10^{-15}$)	N
1	(0.1,0.2,0.5)	1.317627178278510	1.317627178278510 (16)	15
2	$(-0.1, 0.2, 0.5)$	0.695536565102261	0.695536565102261 (16)	15
3	$(0.1, 0.2, -0.5 + i)$	0.667236640109150	0.667236640109149	19
		$+0.274769720129335i$	$+0.274769720129335i$ (14)	
4	$(1 + i, 1 + i, 1 - i)$	1.468693939915885	1.468693939915885	21
		$-2.287355287178842i$	$-2.287355287178842i$ (15)	
5	$(10^{-8}, 10^{-8}, 10^{-10})$	1.000000000100000	1.000000000100000 (16)	3
6	$(10^{-8}, 10^{-12}, -10^{-10} + 10^{-12}i)$	0.999999000000000	0.999999000088899	3
		$+0.000000010000000i$	$+0.000000010000000i$ (11)	
7	$(1, 1, 10 + 10^{-9}i)$	$2.202646579480672 \times 10^{-4}$	$2.202646579480671 \times 10^{-4}$	46
		$+2.02646579480672 \times 10^{-5}i$	$+2.02646579480672 \times 10^{-5}i$ (15)	
8	(1,3,10)	$4.403093158961343 \times 10^{2}$	$4.403093158961343 \times 10^{2}$ (16)	44
9	(500,511,10)	$1.779668553337393 \times 10^{-4}$	$1.779668553337393 \times 10^{-4}$ (16)	46
10	(8.1,10.1,100)	$1.724131075992688 \times 10^{41}$	$1.724131075992686 \times 10^{41}$ (15)	188
11	(1,2,600)	$6.288367168216566 \times 10^{257}$	500 terms computed	N/A
12	(100,1.5,2.5)	$2.748892975858683 \times 10^{12}$	$2.748892975858683 \times 10^{12}$ (16)	48
13	$(-60, 1, 10)$	10.04854112964948	-35.241346779094869 (0)	58
14	(60,1,10)	$1.818086887618945 \times 10^{22}$	$1.818086887618948 \times 10^{22}$ (15)	72
15	$(60, 1, -10)$	$-6.713066845459067 \times 10^{-4}$	$1.608258431433813 \times 10^{5}$ (0)	97
16	$(-60, 1, -10)$	$1.233142540998589 \times 10^{18}$	$1.233142540998589 \times 10^{18}$ (16)	46
17	$(1000, 1, 10^{-3})$	2.279929853828663	2.279929853828663 (16)	12
18	$(10^{-3}, 1, 700)$	$1.46135307199289 \times 10^{298}$	500 terms computed	N/A
19	(500,1,-5)	0.001053895943365	$1.669453216927715 \times 10^{26}$ (0)	128
20	(-500,1,5)	0.251406264291805	$6.711437272547195 \times 10^{23}$ (0)	115
21	$(20, -10 + 10^{-9}, -2.5)$	$8.857934344815256 \times 10^{9}$	$8.857934347919209 \times 10^{9}$ (9)	49
22	$(20, 10 - 10^{-9}, 2.5)$	98.353133058093164	98.353133058093178 (15)	29
23	$(-20, -10 + 10^{-12}, 2.5)$	$-1.051351454763442 \times 10^{14}$	$-1.051351454763442 \times 10^{14}$ (16)	22
24	$(50, 10, 200i)$	$-3.000605782805072 \times 10^{35}$	$1.600646031599127 \times 10^{108}$	425
		$+3.046849261045972 \times 10^{35}i$	$-4.042278036419474 \times 10^{107}i$ (0)	
25	$(-5, (-5 + 10^{-9}) + (-5 + 10^{-9})i, -1)$	0.507421537454510	0.507421537454510	7
		$+0.298577267504408i$	$+0.298577267504408i$ (16)	
26	(4,80,200)	$3.448551506216654 \times 10^{27}$	$3.448551506216651 \times 10^{27}$ (15)	248
27	$(-4, 500, 300)$	0.024906201315854	0.024906201315854 (14)	6
28	$(5, 0.1, -2 + 300i)$	$7.208553632163922 \times 10^{10}$	$9.670266564937197 \times 10^{139}$	452
		$-1.550289119122414 \times 10^{10}i$	$+1.252328293832722 \times 10^{139}i$ (0)	
29	$(-5, 0.1, 2 + 300i)$	$2.897045042631838 \times 10^{10}$	$2.897045042631835 \times 10^{10}$	7
		$-8.276253515853658 \times 10^{11}i$	$-8.276253515853651 \times 10^{11}i$ (15)	
30	$(2 + 8i, -150 + i, 150)$	$-9.853780031496243 \times 10^{135}$	$-9.853780031496170 \times 10^{135}$	409
		$+3.293888962100131 \times 10^{136}i$	$+3.293888962100122 \times 10^{136}i$ (13)	
31	$(5, 2, 100 - 1000i)$	$7.002864442038879 \times 10^{50}$	500 terms computed	N/A
		$+8.973775767458327 \times 10^{50}i$		
32	$(-5, 2, -100 + 1000i)$	$7.196140446954445 \times 10^{11}$	$7.196140446954443 \times 10^{11}$	7
		$-1.233790613611111 \times 10^{12}i$	$-1.233790613611111 \times 10^{12}i$ (15)	
33	$(-5, -2 - i, 1 + (2 - 10^{-10})i)$	61.699999992549998	61.699999992550005	7
		$+9.899999997100000i$	$+9.899999971000000i$ (11)	
34	$(1, 10^{-12}, 1)$	$2.718281828457880 \times 10^{12}$	$2.718281828457886 \times 10^{12}$ (15)	20
35	$(10, 10^{-12}, 10)$	$1.332534440778499 \times 10^{23}$	$1.332415988221225 \times 10^{23}$ (4)	53
36	$(1, -1 + 10^{-12}i, 1)$	$-5.528996131321089 \times 10^{-1}$	$-5.528996131321089 \times 10^{-1}$	21
		$+2.718281828459045 \times 10^{12}i$	$+2.718281828459045 \times 10^{12}i$ (16)	
37	$(1000, 1, -1000)$	$1.805334147110282 \times 10^{-53}$	500 terms computed	N/A
38	$(-1000, 1, 1000)$	$2.593820783362006 \times 10^{215}$	500 terms computed	N/A
39	$(-10 + 500i, 5i, 10)$	$7.086198763185099 \times 10^{43}$	$7.020404376373322 \times 10^{43}$	151
		$+2.328576049934718 \times 10^{43}i$	$+2.431781751237888 \times 10^{43}i$ (1)	
40	$(20, 10 + 1000i, -5)$	0.993763703678828	0.993763703678828	11
		$+0.099687801957356i$	$+0.099687801957356i$ (16)	

Table 20: Numerical results for ${}_1F_1(a;b;z)$ part 1/6; shown is the correct value using MATLAB and verified using Mathematica, the results from Taylor series method (a) from Section 3.2 with the number of digits of accuracy in brackets, and the number of terms computed using this method.

Case	(a, b, z)	Taylor (b) ($tol = 10^{-15}$)	N	Single fraction ($tol = 10^{-15}$)	N
1	(0.1,0.2,0.5)	1.317627178278510 (16)	15	1.317627178278509 (14)	15
2	$(-0.1, 0.2, 0.5)$	0.695536565102261 (16)	15	0.695536565102261 (16)	15
3	$(0.1, 0.2, -0.5 + i)$	0.667236640109150 $+0.274769720129335i$ (16)	19	0.667236640109149 $+0.274769720129335i$ (14)	19
4	$(1 + i, 1 + i, 1 - i)$	1.468693939915885 $-2.287355287178843i$ (15)	21	1.468693939915886 $-2.287355287178842i$ (15)	21
5	$(10^{-8}, 10^{-8}, 10^{-10})$	1.000000000100000 (16)	3	1.000000000100000 (16)	3
6	$(10^{-8}, 10^{-12}, -10^{-10} + 10^{-12}i)$	0.999999000000000 $+0.000000010000000i$ (16)	3	0.999999000000000 $+0.000000010000000i$ (16)	3
7	$(1, 1, 10 + 10^{-9}i)$	$2.202646579480672 \times 10^{-4}$ $+2.02646579480669 \times 10^{-5}i$ (14)	46	$2.202646579480671 \times 10^{-4}$ $+2.02646579480672 \times 10^{-5}i$ (15)	46
8	(1,3,10)	$4.403093158961347 \times 10^{2}$ (15)	44	$4.403093158961341 \times 10^{2}$ (15)	44
9	(500,511,10)	$1.779668553337393 \times 10^{-4}$ (16)	46	$1.779668553337394 \times 10^{-4}$ (15)	45
10	(8.1,10.1,100)	$1.724131075992686 \times 10^{41}$ (15)	188	$1.443638146242888 \times 10^{40}$ (0)	85
11	(1,2,600)	500 terms computed	N/A	$2.708933063093695 \times 10^{94}$ (0)	73
12	(100,1.5,2.5)	$2.748892975858682 \times 10^{12}$ (15)	48	$2.748892975858684 \times 10^{12}$ (15)	48
13	$(-60, 1, 10)$	-13.585500872106090 (0)	58	-19.306589748926299 (0)	58
14	(60,1,10)	$1.818086887618946 \times 10^{22}$ (15)	72	$1.818086887618944 \times 10^{22}$ (15)	97
15	$(60, 1, -10)$	$4.161733968914763 \times 10^{5}$ (0)	96	$6.748784369464462 \times 10^{4}$ (0)	97
16	(-60,1,-10)	$1.233142540998589 \times 10^{18}$ (16)	46	$1.233142540998589 \times 10^{18}$ (16)	46
17	$(1000,1,10^{-3})$	2.279929853828663 (16)	12	2.279929853828663 (16)	12
18	$(10^{-3}, 1, 700)$	500 terms computed	N/A	$1.788934052781859 \times 10^{96}$ (0)	73
19	$(500, 1, -5)$	$1.319078645590728 \times 10^{26}$ (0)	128	$-4.223914178002353 \times 10^{32}$ (0)	90
20	$(-500, 1, 5)$	$2.807220260994878 \times 10^{23}$ (0)	116	$-2.959206777727982 \times 10^{23}$ (0)	92
21	$(20, -10 + 10^{-9}, -2.5)$	$8.857934341268038 \times 10^{9}$ (9)	49	$8.857934344315682 \times 10^{9}$ (10)	49
22	$(20, 10 - 10^{-9}, 2.5)$	98.353133058093192 (14)	29	98.353133058093263 (13)	29
23	$(-20, -10 + 10^{-12}, 2.5)$	$-1.051351454763440 \times 10^{14}$ (15)	22	$-1.051351454763444 \times 10^{14}$ (15)	22
24	$(50, 10, 200i)$	$-8.164726863126853 \times 10^{107}$ $-1.893898004263770 \times 10^{108}i$ (0)	425	NaN+NaNi (0)	95
25	$(-5, (-5 + 10^{-9}) + (-5 + 10^{-9})i, -1)$	0.507421537454510 $+0.298577267504408i$ (16)	7	0.507421537454510 $+0.298577267504408i$ (16)	7
26	(4,80,200)	$3.448551506216642 \times 10^{27}$ (14)	247	$1.020523345666494 \times 10^{24}$ (0)	80
27	$(-4, 500, 300)$	0.024906201315854 (14)	6	0.024906201315854 (14)	6
28	$(5, 0.1, -2 + 300i)$	$9.670266564937197 \times 10^{139}$ $+1.252328293832722 \times 10^{139}i$ (0)	452	NaN+NaNi (0)	99
29	$(-5, 0.1, 2 + 300i)$	$2.897045042631837 \times 10^{10}$ $-8.276253515853658 \times 10^{11}i$ (15)	6	$2.897045042631837 \times 10^{10}$ $-8.276253515853661 \times 10^{11}i$ (14)	7
30	$(2 + 8i, -150 + i, 150)$	$-9.853780031496204 \times 10^{135}$ $+3.293888962100115 \times 10^{136}i$ (14)	409	500 terms computed	N/A
31	$(5, 2, 100 - 1000i)$	500 terms computed	N/A	NaN+NaNi (0)	98
32	$(-5, 2, -100 + 1000i)$	$7.196140446954445 \times 10^{11}$ $-1.233790613611111 \times 10^{12}i$ (16)	7	$7.196140446954445 \times 10^{11}$ $-1.233790613611111 \times 10^{12}i$ (16)	7
33	$(-5, -2 - i, 1 + (2 - 10^{-10})i)$	61.699999992550005 $+9.899999997100002i$ (11)	7	61.699999992549998 $+9.899999997099993i$ (10)	7
34	$(1, 10^{-12}, 1)$	$2.718281828457880 \times 10^{12}$ (16)	20	$2.718281828457880 \times 10^{12}$ (16)	20
35	$(10, 10^{-12}, 10)$	$1.332534440778498 \times 10^{23}$ (15)	53	$1.332534440778500 \times 10^{23}$ (13)	54
36	$(1, -1 + 10^{-12}i, 1)$	$-5.528996131321089 \times 10^{-1}$ $+2.718281828459045 \times 10^{12}i$ (16)	21	$-5.528996131321099 \times 10^{-1}$ $+2.718281828459045 \times 10^{12}i$ (15)	21
37	$(1000, 1, -1000)$	500 terms computed	N/A	$-1.453008575645283 \times 10^{174}$ (0)	52
38	$(-1000, 1, 1000)$	500 terms computed	N/A	$-1.133008385978182 \times 10^{173}$ (0)	52
39	$(-10 + 500i, 5i, 10)$	$7.142144460489667 \times 10^{43}$ $+2.336672709767327 \times 10^{43}i$ (1)	151	500 terms computed	N/A
40	$(20, 10 + 1000i, -5)$	0.993763703678828 $+0.099687801957356i$ (16)	10	0.993763703678828 $+0.099687801957356i$ (16)	12

Table 21: Numerical results for ${}_1F_1(a; b; z)$ part 2/6; shown are the results from Taylor series method (b) in Section 3.2, the single fraction method of Section 3.3, and the number of terms computed for each. The number of digits of accuracy of each method is placed in brackets; this notation will will continue for the remainder of this appendix.

Case	(a, b, z)	Buchholz 1 ($tol = 10^{-15}$)	N	Buchholz 2 ($tol = 10^{-15}$)	N
1	(0.1,0.2,0.5)	170 terms computed	N/A	500 terms computed	N/A
2	$(-0.1, 0.2, 0.5)$	0.695742258430131 (4)	11	0.695536565102262 (15)	12
3	$(0.1, 0.2, -0.5 + i)$	170 terms computed	N/A	500 terms computed	
4	$(1 + i, 1 + i, 1 - i)$	1.469314879177899 $-2.286400529446476i$ (3)	14	1.468693939915886 $-2.287355287178842i$ (15)	15
5	$(10^{-8}, 10^{-8}, 10^{-10})$	1.000000005007177 (9)	3	1.000000005007180 (9)	3
6	$(10^{-8}, 10^{-12}, -10^{-10} + 10^{-12}i)$	1.000004373569878 $+0.000000010000000i$ (0)	3	1.000004373569878 $+0.000000010000000i$ (0)	3
7	$(1, 1, 10 + 10^{-9}i)$	170 terms computed	N/A	$2.202646579480671 \times 10^4$ $+2.2026631199777 \times 10^{-5}i$ (5)	32
8	(1,3,10)	170 terms computed	N/A	$4.403093158961342 \times 10^2$ (15)	32
9	(500,511,10)	170 terms computed	N/A	500 terms computed	N/A
10	(8.1,10.1,100)	170 terms computed	N/A	$1.724131075992711 \times 10^{41}$ (13)	32
11	(1,2,600)	170 terms computed	N/A	500 terms computed	N/A
12	(100,1.5,2.5)	$2.748923904028464 \times 10^{12}$ (4)	10	$2.748892975858687 \times 10^{12}$ (15)	110
13	$(-60, 1, 10)$	170 terms computed	N/A	-10.048954112964900 (14)	41
14	(60,1,10)	170 terms computed	N/A	$1.818086887618946 \times 10^{22}$ (15)	31
15	$(60, 1, -10)$	170 terms computed	N/A	$-6.713066845459049 \times 10^{-4}$ (14)	37
16	$(-60, 1, -10)$	170 terms computed	N/A	$1.233142540998591 \times 10^{18}$ (14)	31
17	$(1000, 1, 10^{-3})$	2.279929853828661 (15)	3	2.279929853828660 (15)	4
18	(10^{-3},1,700)	170 terms computed	N/A	500 terms computed	N/A
19	$(500, 1, -5)$	0.001053940354303 (7)	3	0.001053895943365 (16)	24
20	$(-500, 1, 5)$	0.251412647967877 (5)	11	0.251406264291806 (15)	22
21	$(20, -10 + 10^{-9}, -2.5)$	170 terms computed	N/A	$8.857934360394379 \times 10^9$ (8)	26
22	$(20, 10 - 10^{-9}, 2.5)$	98.355163835291791 (4)	15	98.353132437185800 (7)	17
23	$(-20, -10 + 10^{-12}, 2.5)$	170 terms computed	N/A	$-1.051290662997823 \times 10^{14}$ (4)	26
24	$(50, 10, 200i)$	170 terms computed	N/A	500 terms computed	N/A
25	$(-5, (-5 + 10^{-9}) + (-5 + 10^{-9})i, -1)$	0.507423640026765 $+0.298580648127604i$ (4)	19	0.507421537454510 $+0.298577267504409i$ (15)	15
26	(4,80,200)	170 terms computed	N/A	500 terms computed	N/A
27	$(-4, 500, 300)$	170 terms computed	N/A	500 terms computed	N/A
28	$(5, 0.1, -2 + 300i)$	170 terms computed	N/A	500 terms computed	N/A
29	$(-5, 0.1, 2 + 300i)$	170 terms computed	N/A	500 terms computed	N/A
30	$(2 + 8i, -150 + i, 150)$	170 terms computed	N/A	500 terms computed	N/A
31	$(5, 2, 100 - 1000i)$	170 terms computed	N/A	500 terms computed	N/A
32	$(-5, 2, -100 + 1000i)$	170 terms computed	N/A	500 terms computed	N/A
33	$(-5, -2 - i, 1 + (2 - 10^{-10})i)$	170 terms computed	N/A	500 terms computed	N/A
34	$(1, 10^{-12}, 1)$	$2.719894658150835 \times 10^{12}$ (3)	14	$2.718267221656422 \times 10^{12}$ (5)	15
35	$(10, 10^{-12}, 10)$	170 terms computed	N/A	$2.050732524592003 \times 10^{24}$ (0)	32
36	$(1, -1 + 10^{-12}i, 1)$	$9.811147843567357 \times 10^6$ $+2.722943556625816 \times 10^{12}i$ (0)	17	$9.992870000444725 \times 10^6$ $+2.718281828848585 \times 10^{12}i$	15
37	$(1000, 1, -1000)$	170 terms computed	N/A	500 terms computed	N/A
38	$(-1000, 1, 1000)$	170 terms computed	N/A	500 terms computed	N/A
39	$(-10 + 500i, 5i, 10)$	$6.199417984891287 \times 10^{42}$ $+3.820721389070740 \times 10^{43}i$ (0)	15	$3.982062330684256 \times 10^{42}$ $+3.783409778766555 \times 10^{43}i$ (0)	28
40	$(20, 10 + 1000i, -5)$	170 terms computed	N/A	170 terms computed	N/A

Table 22: Numerical results for ${}_1F_1(a; b; z)$ part 3/6; shown are the results from methods 1 and 2 from Section 3.4, and the number of terms computed for each method.

Case	(a,b,z)	Buchholz 3 ($tol = 10^{-15}$)	N	Beta series ($tol = 10^{-15}$)	N
1	(0.1,0.2,0.5)	1.317839415371721 (4)	11	1.317627177923377 (9)	6
2	$(-0.1, 0.2, 0.5)$	0.695742258430129 (4)	11	0.695536565102261 (16)	6
3	$(0.1, 0.2, -0.5 + i)$	0.665061834948175 $+0.277048273017027i$ (2)	15	0.667236705173041 $+0.277048273017027i$ (3)	6
4	$(1 + i, 1 + i, 1 - i)$	1.469314879177899 $-2.286400529446476i$ (3)	14	500 terms computed $-2.286400529446476i$ (3)	N/A
5	$(10^{-8}, 10^{-8}, 10^{-10})$	1.000000000100000 (9)	3	1.000000000100000 (16)	3
6	$(10^{-8}, 10^{-12}, -10^{-10} + 10^{-12}i)$	0.999999000088888 $+0.000000009999111i$ (9)	3	0.999999000088888 $+0.000000009999111i$ (9)	3
7	$(1, 1, 10 + 10^{-9}i)$	170 terms computed	N/A	500 terms computed	N/A
8	(1,3,10)	170 terms computed	N/A	$4.403093158961343 \times 10^{2}$ (16)	39
9	(500,511,10)	170 terms computed	N/A	$1.779668553337395 \times 10^{-4}$ (15)	12
10	(8.1,10.1,100)	170 terms computed	N/A	500 terms computed	N/A
11	(1,2,600)	170 terms computed	N/A	$2.913833835501408 \times 10^{134}$ (0)	2
12	(100,1.5,2.5)	$2.748923904028461 \times 10^{12}$ (4)	10	500 terms computed	N/A
13	$(-60, 1, 10)$	$1.413226048215916 \times 10^{5}$ (0)	19	500 terms computed	N/A
14	(60,1,10)	170 terms computed	N/A	500 terms computed	N/A
15	$(60, 1, -10)$	2.771191610071790 (0)	18	500 terms computed	N/A
16	$(-60, 1, -10)$	170 terms computed	N/A	500 terms computed	N/A
17	$(1000, 1, 10^{-3})$	2.279929853828663 (16)	3	2.279929853828663 (16)	25
18	$(10^{-3}, 1, 700)$	170 terms computed	N/A	500 terms computed	N/A
19	$(500, 1, -5)$	$1.905228957818582 \times 10^{24}$ (0)	9	500 terms computed	N/A
20	$(-500, 1, 5)$	$7.330744651005754 \times 10^{24}$ (0)	10	500 terms computed	N/A
21	$(20, -10 + 10^{-9}, -2.5)$	170 terms computed	N/A	$8.857765424923950 \times 10^{9}$ (4)	83
22	$(20, 10 - 10^{-9}, 2.5)$	98.355163835291577 (4)	15	98.353133058093931 (13)	21
23	$(-20, -10 + 10^{-12}, 2.5)$	170 terms computed	N/A	$-1.051351364505603 \times 10^{14}$ (7)	69
24	$(50, 10, 200i)$	170 terms computed	N/A	500 terms computed	N/A
25	$(-5, (-5 + 10^{-9}) + (-5 + 10^{-9})i, -1)$	0.507423641026765 $+0.298580648127604i$ (4)	7	0.507421537454510 $+0.298577267504408i$ (16)	21
26	(4,80,200)	170 terms computed	N/A	$-1.051283623213817 \times 10^{33}$ (0)	60
27	$(-4, 500, 300)$	170 terms computed	N/A	0.153016595479951 (0)	114
28	$(5, 0.1, -2 + 300i)$	170 terms computed	N/A	500 terms computed	N/A
29	$(-5, 0.1, 2 + 300i)$	170 terms computed	N/A	500 terms computed	N/A
30	$(2 + 8i, -150 + i, 150)$	170 terms computed	N/A	500 terms computed	N/A
31	$(5, 2, 100 - 1000i)$	170 terms computed	N/A	500 terms computed	N/A
32	$(-5, 2, -100 + 1000i)$	170 terms computed	N/A	500 terms computed	N/A
33	$(-5, -2 - i, 1 + (2 - 10^{-10})i)$	170 terms computed	N/A	61.699999992550538 $+9.89999999709625i$ (10)	45
34	$(1, 10^{-12}, 1)$	$2.719652879427196 \times 10^{12}$ (3)	14	500 terms computed	N/A
35	$(10, 10^{-12}, 10)$	170 terms computed	N/A	500 terms computed	N/A
36	$(1, -1 + 10^{-12}i, 1)$	170 terms computed	N/A	$-5.528996131321100 \times 10^{-1}$ $+2.718281828459045 \times 10^{12}i$ (13)	38
37	$(1000, 1, -1000)$	170 terms computed	N/A	500 terms computed	N/A
38	$(-1000, 1, 1000)$	170 terms computed	N/A	500 terms computed	N/A
39	$(-10 + 500i, 5i, 10)$	$6.199417984891350 \times 10^{42}$ $+3.820721389070737 \times 10^{43}i$ (0)	15	500 terms computed	N/A
40	$(20, 10 + 1000i, -5)$	0.993763703523084 $+0.099687803000057i$ (8)	11	0.993763703678828 $+0.099687801957356i$ (16)	9

Table 23: Numerical results for ${}_1F_1(a;b;z)$ part 4/6; shown are the results from method 3 of Section 3.4, the beta series method of Appendix G.1, and the number of terms computed for each.

Case	(a,b,z)	Asymptotic (a) ($tol = 10^{-15}$)	N	Asymptotic (b) ($tol = 10^{-15}$)	N
1	(0.1,0.2,0.5)	500 terms computed	N/A	500 terms computed	N/A
2	(-0.1,0.2,0.5)	500 terms computed	N/A	500 terms computed	N/A
3	$(0.1, 0.2, -0.5 + i)$	500 terms computed	N/A	500 terms computed	N/A
4	$(1 + i, 1 + i, 1 - i)$	500 terms computed	N/A	500 terms computed	N/A
5	$(10^{-8}, 10^{-8}, 10^{-10})$	500 terms computed	N/A	500 terms computed	N/A
6	$(10^{-8}, 10^{-12}, -10^{-10} + 10^{-12}i)$	500 terms computed	N/A	500 terms computed	N/A
7	$(1, 1, 10 + 10^{-9}i)$	500 terms computed	N/A	500 terms computed	N/A
8	(1,3,10)	$4.403093158961344 \times 10^{2}$ (15)	3/3	$4.403093158961344 \times 10^{2}$ (15)	3/4
9	(500,511,10)	500 terms computed	N/A	500 terms computed	N/A
10	(8.1,10.1,100)	$1.724131075992683 \times 10^{41}$ (15)	10/2	$1.724131075992683 \times 10^{41}$ (2)	11/3
11	(1,2,600)	$6.288367168216566 \times 10^{257}$ (16)	3/3	$6.288367168216566 \times 10^{257}$ (16)	3/3
12	(100,1.5,2.5)	500 terms computed	N/A	500 terms computed	N/A
13	$(-60, 1, 10)$	500 terms computed	N/A	500 terms computed	N/A
14	(60,1,10)	500 terms computed	N/A	500 terms computed	N/A
15	$(60, 1, -10)$	500 terms computed	N/A	500 terms computed	N/A
16	$(-60, 1, -10)$	500 terms computed	N/A	500 terms computed	N/A
17	$(1000, 1, 10^{-3})$	500 terms computed	N/A	500 terms computed	N/A
18	$(10^{-3}, 1, 700)$	$1.461353307199288 \times 10^{298}$ (15)	8/5	$1.461353307199288 \times 10^{298}$ (15)	9/6
19	$(500, 1, -5)$	500 terms computed	N/A	500 terms computed	N/A
20	$(-500, 1, 5)$	500 terms computed	N/A	500 terms computed	N/A
21	$(20, -10 + 10^{-9}, -2.5)$	500 terms computed	N/A	500 terms computed	N/A
22	$(20, 10 - 10^{-9}, 2.5)$	500 terms computed	N/A	500 terms computed	N/A
23	$(-20, -10 + 10^{-12}, 2.5)$	500 terms computed	N/A	500 terms computed	N/A
24	$(50, 10, 200i)$	500 terms computed	N/A	500 terms computed	N/A
25	$(-5, (-5 + 10^{-9}) + (-5 + 10^{-9})i, -1)$	500 terms computed	N/A	500 terms computed	N/A
26	(4,80,200)	$3.448551506216226 \times 10^{27}$ (13)	5/35	$3.448551506216225 \times 10^{27}$ (13)	6/36
27	$(-4, 500, 300)$	500 terms computed	N/A	500 terms computed	N/A
28	$(5, 0.1, -2 + 300i)$	$7.208553632163907 \times 10^{10}$ $-1.550289119122399 \times 10^{10}i$ (13)	6/14	$7.208553632163907 \times 10^{10}$ $-1.550289119122399 \times 10^{10}i$ (13)	7/15
29	$(-5, 0.1, 2 + 300i)$	NaN+NaNi (0)	14/7	NaN+NaNi (0)	15/8
30	$(2 + 8i, -150 + i, 150)$	NaN+NaNi (0)	5/10	NaN+NaNi (0)	6/11
31	$(5, 2, 100 - 1000i)$	NaN+NaNi (0)	11/7	NaN+NaNi (0)	12/8
32	$(-5, 2, -100 + 1000i)$	500 terms computed	N/A	500 terms computed	N/A
33	$(-5, -2 - i, 1 + (2 - 10^{-10})i)$	500 terms computed	N/A	500 terms computed	N/A
34	$(1, 10^{-12}, 1)$	500 terms computed	N/A	500 terms computed	N/A
35	$(10, 10^{-12}, 10)$	500 terms computed	N/A	500 terms computed	N/A
36	$(1, -1 + 10^{-12}i, 1)$	500 terms computed	N/A	500 terms computed	N/A
37	$(1000, 1, -1000)$	500 terms computed	N/A	500 terms computed	N/A
38	$(-1000, 1, 1000)$	500 terms computed	N/A	500 terms computed	N/A
39	$(-10 + 500i, 5i, 10)$	500 terms computed	N/A	500 terms computed	N/A
40	$(20, 10 + 1000i, -5)$	500 terms computed	N/A	500 terms computed	N/A

Table 24: Numerical results for ${}_1F_1(a; b; z)$ part 5/6; shown are the results from asymptotic series methods (a) and (b) of Section 3.5, and the number of terms computed to compute the series (3.23) and (3.24).

Case	(a,b,z)	Gauss-Jacobi ($N_{mesh} = 200$)	N	RK4 ($N_{mesh} = 500$)
1	(0.1,0.2,0.5)	1.317627178278510 (16)	10	1.317627178272184 (12)
2	$(-0.1, 0.2, 0.5)$			0.695536565117007 (11)
3	$(0.1, 0.2, -0.5 + i)$	0.667236640109151	10	0.667236640069050
		$+0.274769720129335i$ (14)		$+0.274769720118543i$ (10)
4	$(1 + i, 1 + i, 1 - i)$			1.468693939915453
				$-2.287355287178543i$ (13)
5	$(10^{-8}, 10^{-8}, 10^{-10})$			1.000000000100030 (14)
6	$(10^{-8}, 10^{-12}, -10^{-10} + 10^{-12}i)$			0.999999999900024
				$+0.000000000001000i$ (7)
7	$(1, 1, 10 + 10^{-9}i)$			$2.202646550655129 \times 10^{-4}$
				$+2.202646550655129 \times 10^{-5}i$ (8)
8	(1,3,10)	$4.403093158961341 \times 10^{2}$ (15)	10	$4.403093148049358 \times 10^{2}$ (8)
9	(500,511,10)	NaN (0)		NaN (0)
10	(8.1,10.1,100)	$1.724131075992687 \times 10^{41}$ (15)	30	$1.722537193851716 \times 10^{41}$ (3)
11	(1,2,600)	$6.288367168215225 \times 10^{257}$ (12)	70	$1.479432114195077 \times 10^{256}$ (0)
12	(100,1.5,2.5)			$2.748885788836260 \times 10^{12}$ (5)
13	$(-60, 1, 10)$			-10.188448906836700 (2)
14	(60,1,10)			$1.816295490482157 \times 10^{22}$ (3)
15	$(60, 1, -10)$			$-6.793293948029929 \times 10^{-4}$ (2)
16	$(-60, 1, -10)$			$1.232168820022416 \times 10^{18}$ (3)
17	$(1000, 1, 10^{-3})$			2.279929853883460 (11)
18	$(10^{-3}, 1, 700)$	$1.461353307199045 \times 10^{298}$ (13)	70	$1.404571654438778 \times 10^{295}$ (0)
19	$(500, 1, -5)$			$-7.627275304977973 \times 10^{-4}$ (0)
20	$(-500, 1, 5)$			-0.032291191622581 (0)
21	$(20, -10 + 10^{-9}, -2.5)$			$5.483601858393873 \times 10^{14}$ (0)
22	$(20, 10 - 10^{-9}, 2.5)$			98.353133200849229 (8)
23	$(-20, -10 + 10^{-12}, 2.5)$			$-6.201407443966957 \times 10^{17}$ (15)
24	$(50, 10, 200i)$			$-2.793635422283713 \times 10^{35}$
				$-2.775425472898587 \times 10^{35}i$ (0)
25	$(-5, (-5 + 10^{-9}) + (-5 + 10^{-9})i, -1)$			$-4.285233435708474 \times 10^{3}$
				$-4.445607978729533 \times 10^{4}i$ (0)
26	(4,80,200)	$6.470060431231330 \times 10^{28}$ (0)	N/A	$-3.322885799839796 \times 10^{59}$ (0)
27	$(-4, 500, 300)$			$8.361936470287937 \times 10^{280}$ (0)
28	$(5, 0.1, -2 + 300i)$			$5.863939931340572 \times 10^{10}$
				$-2.167792798974293 \times 10^{10}i$ (0)
29	$(-5, 0.1, 2 + 300i)$			$3.370368728276094 \times 10^{10}$
				$-8.381696309624608 \times 10^{11}i$ (0)
30	$(2 + 8i, -150 + i, 150)$			NaN+NaNi (0)
31	$(5, 2, 100 - 1000i)$			$-1.311578113933106 \times 10^{13}$
				$+1.010042944762634 \times 10^{13}i$ (0)
32	$(-5, 2, -100 + 1000i)$			$7.842056930954999 \times 10^{11}$
				$-1.255786004291030 \times 10^{12}i$ (1)
33	$(-5, -2 - i, 1 + (2 - 10^{-10})i)$			61.630758240468424
				$+10.102788220752368i$ (0)
34	$(1, 10^{-12}, 1)$			$2.718279502014370 \times 10^{12}$ (5)
35	$(10, 10^{-12}, 10)$			$1.332433731584012 \times 10^{23}$ (4)
36	$(1, -1 + 10^{-12}i, 1)$			$-5.533356053310802 \times 10^{-1}$
				$+2.718218662119980 \times 10^{12}i$ (2)
37	$(1000, 1, -1000)$			$1.148741372573041 \times 10^{248}$ (0)
38	$(-1000, 1, 1000)$			NaN (0)
39	$(-10 + 500i, 5i, 10)$			$7.539964771333821 \times 10^{43}$
				$+1.951376092802496 \times 10^{43}i$ (0)
40	$(20, 10 + 1000i, -5)$			NaN+NaNi (0)

Table 25: Numerical results for ${}_1F_1(a; b; z)$ part 6/6; shown are the results from the Gauss-Jacobi quadrature method of Section 3.6 with 200 mesh points (when $\mathrm{Re}(b) > \mathrm{Re}(a) > 0$), the number of mesh points N_{crit} required to generate 10 digit accuracy with the Gauss-Jacobi quadrature method (where the number of mesh points was increased in increments of 10 until 10 digit accuracy was obtained), and the results from applying the RK4 method of Section 3.7 with 500 mesh points.

F Numerical results for ${}_2F_1(a, b; c; z)$

Case	(a,b,c,z)	Correct ${}_2F_1(a, b; c; z)$	Taylor (a) ($tol = 10^{-15}$)	N
1	(0.1,0.2,0.3,0.5)	1.046432811217352	1.046432811217352 (16)	41
2	$(-0.1,0.2,0.3,0.5)$	0.956434210968214	0.956434210968214 (16)	40
3	$(0.1,0.2,-0.3,-0.5+0.5i)$	1.027216624114001 $-0.013577157567418i$	1.027216624114002 $-0.013577157567418i$ (15)	88
4	$(10^{-8},10^{-8},10^{-8},10^{-6})$	1.000000000000010	1.000000000000010 (16)	3
5	$(10^{-8},-10^{-6},10^{-12},-10^{-10}+10^{-12}i)$	1.000000000001000 $-0.000000000000010i$	1.000000000001000 $+0.000000010000000i$ (16)	3
6	$(1,10,1,0.5+10^{-9}i)$	$1.024000000000000 \times 10^3$ $+2.048000000000000 \times 10^{-5}i$	$1.023999999999999 \times 10^3$ $+2.047999999999990 \times 10^{-5}i$ (3)	79
7	$(1,-1+10^{-12}i,1,-0.8)$	1.800000000000000 $-0.000000000001058i$	1.800000000000000 $-0.000000000001058i$ (16)	10
8	$(2+8i,3-5i,\sqrt{2}-\pi i,0.75)$	$6.882463762011614 \times 10^3$ $-6.596555778724484 \times 10^3 i$	$6.882463762011581 \times 10^3$ $-6.596555778724495 \times 10^3 i$ (13)	163
9	$(100,200,350,i)$	$5.686708048303445 \times 10^{155}$ $+4.471204020179333 \times 10^{155}i$	500 terms computed	N/A
10	$(2+10^{-9},3,5,-0.75)$	0.492238858852651	0.492238858852651 (16)	115
11	$(-2,-3,-5+10^{-9},0.5)$	0.474999999913750	0.474999999913750 (16)	3
12	$(-1,-1.5,-2-10^{-15},0.5)$	0.625000000000000	0.625000000000000 (16)	2
13	$(500,-500,500,0.75)$	$9.332636185032189 \times 10^{-302}$	$3.047459035998018 \times 10^{102}$ (0)	351
14	$(500,500,500,-0.6)$	$8.709809816217217 \times 10^{-103}$	500 terms computed	N/A
15	$(-1000,-2000,-4000.1,-0.5)$	$5.233580403196932 \times 10^{94}$	$5.233580403196953 \times 10^{94}$ (14)	293
16	$(-100,-200,-300+10^{-9},0.5\sqrt{2})$	$2.653635302903707 \times 10^{-31}$	0.078395353116161 (0)	87
17	$(300,10,5,0.5)$	$3.912238919961547 \times 10^{98}$	500 terms computed	N/A
18	$(5,-300,10,0.5)$	$1.661006238211309 \times 10^{-7}$	$4.628142177960427 \times 10^{29}$ (0)	199
19	$(10,5,-300.5,0.5)$	$-3.852027081523919 \times 10^{32}$	0.921182716632848 (0)	12
20	$(2+200i,5,10,0.6)$	$1.499739394713933 \times 10^{-7}$ $+5.771450716812297 \times 10^{-7}i$	$-8.206946157063342 \times 10^{31}$ $+8.961768586845500 \times 10^{31}i$ (0)	399
21	$(2+200i,5-100i,10+500i,0.8)$	-4.103442641430799 $+6.013632243569482i$	-4.102898902166944 $+6.016364243229925i$ (3)	146
22	$(2,5,10-500i,-0.8)$	0.999450314116122 $-0.015980509652011i$	0.999450314116122 $-0.015980509652011i$ (16)	9
23	$(2.25, 3.75, -0.5, -1)$	-0.631220676949703	500 terms computed	N/A
24	$(1,2,4+3i,0.6-0.8i)$	0.834550347995121 $-0.316176129469793i$	500 terms computed	22
25	$(1,0.9,2,e^{i\pi/3})$	0.932633569241998 $+0.475200538581623i$	500 terms computed	N/A
26	$(1,2.5,5,e^{i\pi/3})$	0.950417228136049 $+0.548723642506806i$	500 terms computed	N/A
27	$(-1,0.9,2,e^{-i\pi/3})$	0.775000000000000 $+0.389711431702997i$	500 terms computed	N/A
28	$(4,1.1,2,0.5+(0.5\sqrt{3}-0.01)i)$	-0.470097672835090 $+0.500986178581549i$	500 terms computed	N/A
29	$(5,2.2,-2.5,0.49+0.5\sqrt{3}i)$	$1.084589030597151 \times 10^3$ $-5.115786480028586i$	500 terms computed	N/A
30	$(\frac{2}{3},1,\frac{4}{3},e^{i\pi/3})$	0.883319375142725 $+0.509984679019064i$	500 terms computed	N/A

Table 26: Numerical results for ${}_2F_1(a, b; c; z)$ part 1/4; shown is the correct value using MATLAB and verified with Mathematica, the results from Taylor series method (a) from Section 4.2, and the number of terms computed using this method. The number of digits of accuracy this method is placed in brackets; this notation will continue for the remainder the rest of this appendix.

Case	(a,b,c,z)	Taylor (b) ($tol = 10^{-15}$)	N	Single fraction ($tol = 10^{-15}$)	N
1	(0.1,0.2,0.3,0.5)	1.046432811217351 (15)	41	1.046432811217352 (16)	42
2	(−0.1,0.2,0.3,0.5)	0.956434210968214 (16)	40	0.956434210968215 (15)	41
3	(0.1,0.2,−0.3,−0.5 + 0.5i)	1.027216624114002 −0.013577157567418i (15)	88	1.027216624114002 −0.013577157567418i (15)	90
4	(10^{-8},10^{-8},10^{-8},10^{-6})	1.000000000000010 (16)	3	1.000000000000010 (16)	3
5	(10^{-8},-10^{-6},10^{-12},$-10^{-10} + 10^{-12}i$)	1.000000000001000 +0.000000010000000i (16)	3	1.000000000001000 −0.000000000000010i (16)	3
6	(1,10,1,0.5 + $10^{-9}i$)	$1.023999999999999 \times 10^{-4}$ $+2.047999999999989 \times 10^{-5}i$ (3)	79	NaN+NaNi (0)	N/A
7	(1,$-1 + 10^{-12}i$,1,−0.8)	1.800000000000000 −0.000000000001058i (16)	10	1.800000000000000 −0.000000000001058i (16)	10
8	(2 + 8i,3 − 5i,$\sqrt{2} - \pi i$,0.75)	$6.882463762011582 \times 10^3$ $-6.596555778724483 \times 10^3 i$ (13)	163	500 terms computed	N/A
9	(100,200,350,i)	500 terms computed	N/A	500 terms computed	N/A
10	(2 + 10^{-9},3,5,−0.75)	0.492238858852651 (16)	115	0.492238858852701 (13)	97
11	(−2,−3,−5 + 10^{-9},0.5)	0.474999999913750 (16)	3	0.474999999913750 (16)	4
12	(−1,−1.5,−2 − 10^{-15},0.5)	0.625000000000000 (16)	2	0.625000000000000 (16)	3
13	(500,−500,500,0.75)	$-1.308965312275652 \times 10^{104}$ (0)	347	$2.282231327371186 \times 10^{69}$ (0)	59
14	(500,500,500,−0.6)	500 terms computed	N/A	−Inf (0)	60
15	(−1000,−2000,−4000.1,−0.5)	$5.233580403196921 \times 10^{94}$ (14)	293	$3.206764029878514 \times 10^{55}$ (0)	53
16	(−100,−200,−300 + 10^{-9},0.5$\sqrt{2}$)	−0.117989577492215 (0)	87	0.026445004024236 (0)	80
17	(300,10,5,0.5)	500 terms computed	N/A	$6.070771369227341 \times 10^{64}$ (0)	82
18	(5,−300,10,0.5)	$3.698084043503173 \times 10^{28}$ (0)	201	$-5.218499333377090 \times 10^{44}$ (0)	86
19	(10,5,−300.5,0.5)	0.921182716632848 (0)	12	0.921182716632848 (0)	12
20	(2 + 200i,5,10,0.6)	$3.168834584274869 \times 10^{32}$ $+5.250473854631711 \times 10^{31} i$ (0)	396	NaN+NaNi (0)	N/A
21	(2 + 200i,5 − 100i,10 + 500i,0.8)	−4.100998516155065 +6.017586881185369i (3)	146	500 terms computed	N/A
22	(2,5,10 − 500i,−0.8)	0.999450314116122 −0.015980509652011i (16)	9	0.999450314116122 −0.015980509652011i (16)	10
23	(2.25, 3.75, −0.5, −1)	500 terms computed	N/A	$-3.096946800128161 \times 10^{10}$ (0)	98
24	(1,2,4 + 3i,0.6 − 0.8i)	500 terms computed	N/A	500 terms computed	N/A
25	(1,0.9,2,$e^{i\pi/3}$)	500 terms computed	N/A	500 terms computed	N/A
26	(1,1,4,$e^{i\pi/3}$)	500 terms computed	N/A	500 terms computed	N/A
27	(−1,0.9,2,$e^{-i\pi/3}$)	500 terms computed	N/A	500 terms computed	N/A
28	(4,1.1,2,0.5 + (0.5$\sqrt{3}$ − 0.01)i)	500 terms computed	N/A	500 terms computed	N/A
29	(5,2.2,−2.5,0.49 + 0.5$\sqrt{3}i$)	500 terms computed	N/A	500 terms computed	N/A
30	($\frac{2}{3}$,1,$\frac{4}{3}$,$e^{i\pi/3}$)	500 terms computed	N/A	500 terms computed	N/A

Table 27: Numerical results for ${}_2F_1(a, b; c; z)$ part 2/4; shown are results from Taylor series method (b) from Section 4.2, the number of terms computed using this method, the results from the single fraction method of Section 4.3, and the number of terms computed using that method.

Case	(a,b,c,z)	Gauss-Jacobi ($N_{mesh}=200$)	N_{crit}	RK4 ($N_{mesh}=500$)
1	$(0.1,0.2,0.3,0.5)$	1.046432811217352 (16)	10	1.046432811211848 (12)
2	$(-0.1,0.2,0.3,0.5)$			0.956434210972278 (11)
3	$(0.1,0.2,-0.3,-0.5+0.5i)$			1.027216624331061 $-0.013577157137817i$ (10)
4	$(10^{-8},10^{-8},10^{-8},10^{-6})$			1.000000000000000 (14)
5	$(10^{-8},-10^{-6},10^{-12},-10^{-10}+10^{-12}i)$			1.000000000000999 $-0.000000000000010i$ (12)
6	$(1,10,1,0.5+10^{-9}i)$			$1.023999993125049\times10^{3}$ $+2.047999977031734\times10^{3}i$ (3)
7	$(1,-1+10^{-12}i,1,-0.8)$			1.800000000000023 $-0.000000000001058i$ (14)
8	$(2+8i,3-5i,\sqrt{2}-\pi i,0.75)$			$6.882465637979511\times10^{3}$ $-6.596556746271624\times10^{3}i$ (6)
9	$(100,200,350,i)$	NaN+NaNi (0)	N/A	$-1.464729971468833\times10^{210}$ $+5.133950185570621\times10^{209}i$ (0)
10	$(2+10^{-9},3,5,-0.75)$	0.492238858852651 (16)	10	0.492238858850299 (12)
11	$(-2,-3,-5+10^{-9},0.5)$			$-5.809677420327498\times10^{4}$ (0)
12	$(-1,-1.5,-2-10^{-15},0.5)$			0.624999999997437 (3)
13	$(500,-500,500,0.75)$			NaN (0)
14	$(500,500,500,-0.6)$			NaN (0)
15	$(-1000,-2000,-4000.1,-0.5)$			NaN(0)
16	$(-100,-200,-300+10^{-9},0.5\sqrt{2})$			$-$Inf (0)
17	$(300,10,5,0.5)$			$3.659136729623440\times10^{298}$ (1)
18	$(5,-300,10,0.5)$	$1.661006238211367\times10^{-7}$ (14)	40	$1.660425565716309\times10^{-7}$ (3)
19	$(10,5,-300.5,0.5)$			Inf (0)
20	$(2+200i,5,10,0.6)$	$1.499739529309672\times10^{-7}$ $+5.771450811496187\times10^{-7}i$ (7)	460	$1.486356299813440\times10^{-7}$ $+5.756173666779157\times10^{-7}i$ (2)
21	$(2+200i,5-100i,10+500i,0.8)$	NaN+NaNi (0)	N/A	NaN+NaNi (0)
22	$(2,5,10-500i,-0.8)$	NaN+NaNi (0)	N/A	$4.508369694146940\times10^{280}$ $+8.641388250534234\times10^{280}i$ (0)
23	$(2.25,3.75,-0.5,-1)$			$-1.379795759846573\times10^{22}$ (0)
24	$(1,2,4+3i,0.6-0.8i)$	0.23026616844809 $+0.134028428957673i$ (0)	N/A	0.834550347993301 $-0.316176129470047i$ (11)
25	$(1,0.9,2,e^{i\pi/3})$	0.932633569241997 $+0.475200538581622i$ (14)	10	500 terms computed
26	$(1,2.5,5,e^{i\pi/3})$	0.950417228135954 $+0.548723642506928i$ (11)	10	0.950417228134638 $+0.548723642510377i$ (10)
27	$(-1,0.9,2,e^{-i\pi/3})$	0.775000000000000 $+0.389711431702997i$ (16)	10	0.775000000000025 $+0.389711431702995i$ (14)
28	$(4,1.1,2,0.5+(0.5\sqrt{3}-0.01)i)$	-0.470097672835091 $+0.500986178581549i$ (16)	20	-0.470097673031055 $+0.500986178535211i$ (10)
29	$(5,2.2,-2.5,0.49+0.5\sqrt{3}i)$			$1.141887557446073\times10^{3}$ $-4.761996862215397\times10^{3}i$ (1)
30	$(\frac{2}{3},1,\frac{4}{3},e^{i\pi/3})$	0.883319375142726 $+0.509984679019065i$ (15)	10	0.883319375176185 $+0.509984679041032i$ (11)

Table 28: Numerical results for ${}_2F_1(a,b;c;z)$ part 3/4; shown are the results from the Gauss-Jacobi quadrature method of Section 4.4 with 200 mesh points (when $\mathrm{Re}(c)>\mathrm{Re}(b)>0$ or $\mathrm{Re}(c)>\mathrm{Re}(a)>0$), the number of mesh points N_{crit} required to generate 10 digit accuracy with the Gauss-Jacobi quadrature method (where the number of mesh points was increased in increments of 10 until 10 digit accuracy was obtained), and the results from applying the RK4 method of Section 4.5 with 500 mesh points.

Case	a,b,c,z)	Analytic cont. ($tol = 10^{-15}$)	N	z_0
1	(0.1,0.2,0.3,0.5)	500 terms computed	N/A	N/A
2	$(-0.1$,0.2,0.3,0.5)	500 terms computed	N/A	N/A
3	$(0.1$,0.2,$-0.3,-0.5+0.5i)$	1.027216624114002 $-0.013577157567418i$ (16)	39	0.5
4	$(10^{-8},10^{-8},10^{-8},10^{-6})$	500 terms computed	N/A	N/A
5	$(10^{-8},-10^{-6},10^{-12},-10^{-10}+10^{-12}i)$	500 terms computed	N/A	N/A
6	$(1,10,1,0.5+10^{-9}i)$	500 terms computed	N/A	N/A
7	$(1,-1+10^{-12}i,1,-0.8)$	500 terms computed	N/A	N/A
8	$(2+8i,3-5i,\sqrt{2}-\pi i,0.75)$	500 terms computed	N/A	N/A
9	$(100,200,350,i)$	500 terms computed	N/A	N/A
10	$(2+10^{-9}$,3,5,$-0.75)$	500 terms computed	N/A	N/A
11	$(-2,-3,-5+10^{-9}$,0.5)	500 terms computed	N/A	N/A
12	$(-1,-1.5,-2-10^{-15}$,0.5)	500 terms computed	N/A	N/A
13	(500,-500,500,0.75)	500 terms computed	N/A	N/A
14	(500,500,500,-0.6)	500 terms computed	N/A	N/A
15	$(-1000,-2000,-4000.1,-0.5)$	500 terms computed	N/A	N/A
16	$(-100,-200,-300+10^{-9},0.5\sqrt{2})$	500 terms computed	N/A	N/A
17	(300,10,5,0.5)	500 terms computed	N/A	N/A
18	(5,-300,10,0.5)	500 terms computed	N/A	N/A
19	(10,5,-300.5,0.5)	500 terms computed	N/A	N/A
20	$(2+200i$,5,10,0.6)	500 terms computed	N/A	N/A
21	$(2+200i,5-100i,10+500i,0.8)$	500 terms computed	N/A	N/A
22	$(2,5,10-500i,-0.8)$	500 terms computed	N/A	N/A
23	$(2.25, 3.75, -0.5, 1)$	-0.631220676949703 (16)	54	0.5
24	$(1,2,4+3i,0.6-0.8i)$	500 terms computed	N/A	N/A
25	$(1,0.9,2,e^{i\pi/3})$	0.932633569242002 $+0.475200538581619i$ (11)	63	0.5
26	$(1,1,4,e^{i\pi/3})$	0.950417228135954 $+0.548723642506928i$ (11)	89	0.5
27	$(-1,0.9,2,e^{-i\pi/3})$	0.775000000000000 $+0.389711431702997i$ (16)	3	0.5
28	$(4,1.1,2,0.5+(0.5\sqrt{3}-0.01)i)$	-0.470097672835090 $+0.500986178581549i$ (16)	79	0.5
29	$(5,2.2,-2.5,0.49+0.5\sqrt{3}i)$	$1.084589030597452\times10^3$ $-5.115786480030682\times10^3 i$ (11)	125	0.5
30	$(\frac{2}{3},1,\frac{4}{3},e^{i\pi/3})$	0.883319375142725 $+0.509984679019064i$ (16)	58	0.49

Table 29: Numerical results for ${}_2F_1(a,b;c;z)$ part 4/4; shown are the results using the analytic continuation theory of Section 4.7, the number of terms computed using this method, and the value of z_0 used to carry out the computation.

G Other methods considered for evaluating ${}_1F_1(a;b;z)$

In this appendix, we discuss other methods that we considered for computing the confluent hypergeometric function ${}_1F_1(a;b;z)$. We form a judgement of these methods either by researching them and deeming them unsuitable or inferior to other methods, or by implementing them and obtaining numerical results that were not as accurate or fast as other methods discussed in Section 3. Details are set out of the background to each method, and a brief analysis of its effectiveness is given.

G.1 Series in terms of beta random variables

The method described below is based on the theory of a beta random variable $\beta(\alpha, \delta)$, for $\alpha, \delta > 0$. In [44], μ_j is defined as

$$\mu_j = \mathbb{E}[\beta(\alpha,\delta) - \mathbb{E}\beta(\alpha,\delta)]^j,$$

and the beta random variable is stated to have the following **moment generating function** (which is defined as $\phi(z) = \mathbb{E}[zX],\ z \in \mathbb{R}$, where X is the random variable being considered):

$$\phi(z) = M(\alpha;\alpha+\delta;z) = {}_1F_1(\alpha;\alpha+\delta;z).$$

As shown in [44], the following series expression for ${}_1F_1(a;b;z)$ holds:

$$M(a;b;z) = e^{az/b}\left[1 + z^2\sum_{j=0}^{\infty}\frac{\mu_{j+2}}{(j+2)!}z^j\right], \tag{G.1}$$

where

$$\begin{aligned} &\mu_0 = 1, \quad \mu_1 = 0 \\ &\mu_{j+1} = a\sum_{k=1}^{j}\frac{\mu_{j-k}}{b+j-k}\prod_{l=0}^{k-1}\frac{(b-a)(j-l)}{b(b+j-l)} - \frac{aj\mu_j}{b(b+j)}, \quad j = 1,2,\ldots\,. \end{aligned} \tag{G.2}$$

Verdict: The results shown in Table 30 and Appendix E indicate that although the beta series method does not generally give as accurate results as, for example, the Taylor series methods, it is fairly competitive in terms of the accuracy it generates, especially for $\mathrm{Re}(b) > \mathrm{Re}(a) > 0$. There is an issue in terms of accuracy when $b = 2a$ and z is real, because this case is the threshold between the terms in the series (G.1) having alternating signs and

Case	(a,b,z)	Correct $M(a;b;z)$	Acc.	Time taken	Beta series ($tol = 10^{-15}$)	Acc.	N	Time taken
1	(0.1,0.2,0.5)	1.317627178278510	16	11.164293s	1.317627177923377	9	6	0.178753s
2	$(-0.1,0.2,0.5)$	0.695536565102261	16	11.048696s	0.695536565102261	16	6	0.180622s
8	(1,3,10)	$4.403093158961343\times10^{-2}$	16	11.513977s	$4.403093158961343\times10^{-2}$	16	39	0.196009s
11	(1,2,600)	$6.288367168216566 \times 10^{257}$	0	11.892762s	$2.913833835501408 \times 10^{134}$	0	2	0.100343s
21	$(20,-10+10^{-9},-2.5)$	$8.857934344815256 \times 10^{9}$	9	10.971249s	$8.857765424923950 \times 10^{9}$	4	83	0.214479s
25	$(-5,(-5+10^{-9})$ $+(-5+10^{-9})i,-1)$	0.507421537454510 $+0.298577267504408i$	16	11.231796s	0.507421537454510 $+0.298577267504408i$	16	21	0.195662s

Table 30: Table showing the correct solution using MATLAB for a variety of test cases from Appendix B and their computation times, the accuracy of Taylor series method (a) of Section 3.2 (fourth column), and the results, accuracy and computation times of computing the series of beta random variables, as detailed in this appendix. Full results are shown in Appendix E.

all the terms being positive. Otherwise, however, 10 digit accuracy is usually obtained for $|a|, |b|, |z| \lesssim 15$, although it is outperformed by each of the methods described in Sections 3.2–3.5 for most parameter regimes, and the time taken to achieve the results using the beta series method is very long compared to these other methods, because the computation of each term of the series in (G.1) itself requires a computation of a product of terms.

G.2 Expansion in terms of incomplete gamma functions

The starting point of this method, recommended in [44], is that if $b > a > 0$ and $z > 0$ for real a, b, z, then

$$M(a;b;-z) = \frac{\Gamma(b)}{\Gamma(a)\Gamma(b-a)} \int_0^1 e^{-zt} t^{a-1} (1-t)^{b-a-1} \mathrm{d}t.$$

Then, using the binomial expansion for $f(t) = (1-t)^{b-a-1}$,

$$f(t) = \sum_{j=0}^{\infty} \frac{(t-t_0)^j}{j!} f^{(j)}(t_0) = \sum_{j=0}^{\infty} (a-b+1)_j \frac{t^j}{j!},$$

about the point $t_0 = 0$, and transforming the integral, we obtain

$$\begin{aligned} M(a;b;-z) &= \frac{\Gamma(b)}{\Gamma(b-a)\Gamma(a)} z^{-a} \sum_{j=0}^{\infty} (a-b+1)_j \frac{1}{j! z^j} \int_0^z e^{-u} u^{j+a-1} \mathrm{d}u \\ &= \frac{\Gamma(b)}{\Gamma(b-a)\Gamma(a)} z^{-a} \sum_{j=0}^{\infty} (a-b+1)_j \frac{1}{j! z^j} \gamma(z, j+a), \end{aligned} \tag{G.3}$$

giving a computation for $-z < 0$.

It is added in [44] that a second transformation gives the expression

$$M(a;b;-z) = \frac{\Gamma(b)}{\Gamma(b-a)} z^{-a} \sum_{j=0}^{\infty} \frac{(a-b+1)_j (a)_j}{j! z^j} F_\gamma(z, j+a), \tag{G.4}$$

where F_γ is the **cumulative distribution function** of the γ-distribution, which is defined as its integral over the entire real line. Using the transformation (3.7), the method can be extended to approximate $M(a;b;z)$ for $z > 0$ as follows:

$$M(a;b;z) = e^z z^{b-a} \frac{\Gamma(b)}{\Gamma(a)\Gamma(b-a)} \sum_{j=0}^{\infty} \frac{(1-a)_j}{j! z^j} \gamma(z, j+b-a),$$

$$M(a;b;z) = e^z z^{a-b} \frac{\Gamma(b)}{\Gamma(a)} \sum_{j=0}^{\infty} \frac{(b-a)_j (1-a)_j}{j! z^j} F_\gamma(z, j+b-a).$$

This method can also be extended to the case $\mathrm{Re}(b) > \mathrm{Re}(a) > 0$ for complex parameters.

Verdict: We implemented this method, and found the accuracy it generated to vary greatly for different parameter regimes. For example, for $(a, b, z) = (1, 5, -20)$, the method generated the correct answer to 16 digit accuracy, and for $(a, b, z) = (0.1, 50, -20)$, 9 digit accuracy was obtained, but with $(a, b, z) = (10, 50, -20)$, the method did not give a single digit of accuracy. Whereas the method seemed to work well for some parameter values, due to its unreliability and the computation time (roughly 5 times as long as the Taylor series methods of Section 3.2), this method was not considered one of the best tested.

G.3 Asymptotic expansion for large $|b|$ and $|z|$

For large $|z|$, the basic asymptotic expansion is a series involving ${}_2F_0$ functions, which unlike the function ${}_1F_1$ involve the Pochhammer symbol of a term that includes the parameter b in both the numerator and the denominator. Therefore, when computing the asymptotic expansion for large z, we no longer have the advantage of cases with large $|b|$ requiring few terms to generate an accurate approximation; in fact, our experiments show that it becomes very difficult. One method of resolving this issue is to use the asymptotic expansions for large $|b|$ and $|z|$, which are stated in [34], and noted below.

$\underline{b-z-a-1<0 \text{ (for this expansion, } a, b \text{ and } z \text{ must be real):}}$

$$M(a;b;z) \sim \frac{\Gamma(b)}{\Gamma(a)\Gamma(b-a)} e^z (1-\alpha)^{a-1} \left(\frac{\alpha}{e}\right)^{\alpha z} \sqrt{\frac{2\alpha\pi}{z}} \times \left[1 + \sum_{n=1}^{\infty} \sum_{k=0}^{2} \frac{a_{6n+2k-4}(z)}{\sqrt{\pi}} \left(\frac{2\alpha}{z}\right)^{3n+k-2} \Gamma\left(3n+k-\frac{3}{2}\right)\right],$$

$$a_n(z) = \sum_{k=0}^{\lfloor \frac{n}{2} \rfloor} \sum_{j=0}^{n-2k} \sum_{l=0}^{j} \frac{(-1)^j (-\alpha z)_l (1-a)_{n-2k-j} z^{j+k-l}}{2^k (j-l)! l! k! (n-2k-j)! \alpha^{k+l} (1-\alpha)^{n-2k-j}},$$

$$0 < \alpha = \frac{b-a-1}{z} < 1.$$

$\underline{\mathrm{Re}(b-z-a-1)>0:}$

$$M(a;b;z) \sim \frac{\Gamma(b)}{\Gamma(a)\Gamma(b-a)} \times \sum_{n=0}^{\infty} \frac{\Gamma(2n+a-1)[(2n+a-1)a_{2n}(z) + (b-a-z-1)a_{2n-1}(z)]}{(b-a-z-1)^{2n+a}},$$

$$a_n(z) = \sum_{k=0}^{n} \frac{(a+1-b)_k (b-a-1)^{n-k}}{k!(n-k)!}.$$

$\underline{\mathrm{Re}(b-z-a-1)=0:}$

$$M(a;b;z) \sim \frac{\Gamma(b)}{2\Gamma(a)\Gamma(b-a)} \left(\frac{2}{b-a-1}\right)^{\frac{a}{2}} \times \left[\Gamma\left(\frac{a}{2}\right) + \sum_{k=0}^{1} a_{2k+1}(z) \Gamma\left(k+\frac{a+1}{2}\right) \left(\frac{2}{b-a-1}\right)^{k+\frac{1}{2}} + \sum_{n=2}^{\infty} \sum_{k=0}^{2} a_{3n+2k-4}(z) \Gamma\left(k+\frac{3n+a}{2}-2\right) \left(\frac{2}{b-a-1}\right)^{k+\frac{3n}{2}-2}\right],$$

$$a_n(z) = \sum_{k=0}^{\lfloor \frac{n}{2} \rfloor} \sum_{j=0}^{n-2k} \frac{(a+1-b)_k (b-a-1)^{n-k-j}}{j!k!(n-3k)!2^k}.$$

Verdict: A major drawback of this method is that an expansion for the case $\mathrm{Re}(b-z-a-1)<0$ for complex a, b, z is not stated in [34], thereby restricting the potential robustness of the method significantly. Furthermore, the series computations involve many more computations than other methods, taking roughly twice as long as the Taylor series methods of Section 3.2. The method is also generally out-performed by applying recurrence relation

techniques on the parameter b for large $|b|$, as in Section 3.8. Additionally, for moderate values of $|b|$, the method was not very robust; for example with $(a, b, z) = (0.1, 50, 20)$, only 6 digit accuracy was obtained, and with $(a, b, z) = (4, 50, 20)$, not even one digit of accuracy was obtained.

G.4 Hyperasymptotic expansions

Applying certain expansions, called **hyperasymptotic expansions**, results in the computation error being exponentially small as opposed to algebraically small, which means that small terms are added to the expansion to obtain an exponential term in the error bound for the computation. The subject of hyperasymptotics is explained in more detail in [11, 49, 50].

- **First hyperasymptotic expansion:** One such expansion, discussed in [48, 52, 53], for $U(a; b; z)$ is

$$U(a; b; z) = z^{-a} \sum_{j=0}^{N-1} \frac{(a)_j (a-b+1)_j}{j!} (-z)^{-j} + R_N(a; b; z), \tag{G.5}$$

where

$$\begin{aligned} R_N(a; b; z) = {} & \frac{2\pi(-1)^N z^{a-b}}{\Gamma(a)\Gamma(a-b+1)} \\ & \times \left(\sum_{j=0}^{M-1} \frac{(1-a)_j (b-a)_j}{(-z)^j j!} G_{N+2a-b-j}(z) + (1-a)_M (b-a)_M R_{M,N}(a; b; z) \right), \\ G_\eta(z) = {} & \frac{e^z}{2\pi} \Gamma(\eta)\Gamma(1-\eta, z), \end{aligned} \tag{G.6}$$

and $\Gamma(\varpi, z)$ is the incomplete gamma function defined by

$$\Gamma(\varpi, z) = \int_z^\infty t^{\varpi-1} e^{-t} \mathrm{d}t = \Gamma(\varpi) - \gamma(\varpi, z),$$

where $\gamma(\varpi, z)$ is the incomplete gamma function introduced in (3.11).

This is shown in [53] by using the integral expression for $U(a; b; z)$,

$$U(a; b; z) = \frac{1}{\Gamma(a)} \int_0^\infty e^{-zt} t^{a-1} (1+t)^{-b} \mathrm{d}t, \quad \mathrm{Re}(a) > 0, \quad |\arg z| < \frac{\pi}{2},$$

to express $R_N(a;b;z)$ as an integral, and then applying Cauchy's integral theorem.

Then, as stated in [53], if δ is taken to be an arbitrary small parameter, and a, b, m are fixed, as $|z| \to \infty$,

$$R_{M,N}(a;b;z) = \begin{cases} O(e^{-|z|}z^{-M}), & \text{if } |\arg\ z| < \pi, \\ O(e^{z}z^{-M}), & \text{if } \pi \leq |\arg\ z| \leq \frac{5}{2}\pi - \delta, \end{cases} \tag{G.7}$$

which is shown by deriving an integral expression for $R_{M,N}(a;b;z)$ and applying Taylor's theorem, Cauchy's integral theorem and Stirling's formula. Note that the expression (G.7) for $R_{M,N}(a;b;z)$ incorporates its values in two separate branches.

Verdict: As MATLAB is unable to compute the incomplete gamma function $\Gamma(\varpi, z)$ for all ϖ or z in the complex plane, we were not able to generate accurate results using this method. One piece of further work on this project would be to write a routine that computes the incomplete gamma function for all complex ϖ and z, and thereby makes use of the expansion (G.5) for the purposes of computing $U(a;b;z)$, subsequently using this to compute $M(a;b;z)$ with (3.6). Ideas of how we might write such a routine are outlined in Appendix I.

- **Second hyperasymptotic expansion:** For large $|z|$, as discussed in [48], the following is another exponentially-improved expansion for $U(a;b;z)$:

$$U(a;b;z) \sim z^{-a} \sum_{j=0}^{N_0-1} (-1)^j \frac{(a)_j (b)_j}{j!\ z^j} + \frac{1}{\Gamma(a)} R_{N_0}(z), \tag{G.8}$$

where N_0 is a function of $|z|$ and

$$\begin{aligned} R_{N_0}(z) &= e^{-ia\theta} \int_0^\infty e^{-\rho\tau} \tau^{a-1+N_0} f_1(\tau) \mathrm{d}\tau, \\ f_1(\tau) &= \frac{1}{2\pi i} \int_{\Omega_0(0,\tau)} \frac{\mathrm{d}w}{(1+we^{i\theta})^b w^{N_0}(w-\tau)}, \\ \Omega_0(\tau) &= \{w \in \mathbb{C} : |w| = 1-\epsilon\} \cup \{w \in \mathbb{C} : |w-\tau| = \frac{1}{2}\epsilon\}, \\ z &= \rho e^{i\theta},\ t = \tau e^{i\theta},\ -\pi + \delta \leq \theta \leq \pi - \delta,\ 0 < \delta \ll 1,\ 0 < \epsilon < \frac{1}{2}. \end{aligned}$$

By residue calculations, we obtain

$$f_1(\tau) = \frac{1}{(1+\tau e^{-i\theta})\tau^{N_0}} + e^{i\theta}(-1)^{N_0-1}(N_0-1)! \sum_{j=0}^{N_0-1} \frac{(b)_j}{(1)_j} \frac{1}{e^{i\theta(b+j)}(-\tau)^{N_0-j}}.$$

Now, as shown in [48], if $f_1(\tau)$ can be written as

$$f_1(\tau) = a_{0,1} + a_{1,1}(\tau - \gamma_1) + ... + a_{N_1,1}(\tau - \gamma_1)^{N_1 - 1} + (\tau - \gamma_1)^{N_1},$$

where $\gamma_1 = \frac{a-1+N_0}{\rho}$,

$$f_2(\tau) = \frac{1}{2\pi i} \int_{\Omega_1(\gamma_1,\tau)} \frac{f_1(w)}{(w-\gamma_1)^{N_1}(w-\tau)} \mathrm{d}w,$$

and $\Omega_1(\gamma_1, \tau)$ is a contour that encircles γ_1 and τ, then we obtain

$$R_{N_0}(z) = \Gamma(a+N_0) z^{-a} \sum_{j=0}^{N_1-1} a_{j,1} P_j^1(\gamma_1) \rho^{-(N_0+j)} + R_{N_1}(z). \tag{G.9}$$

Here, P_j^1 is given by

$$P_j^1(\gamma_1) = \frac{\rho^{a+N_0+j}}{\Gamma(a+N_0)} \int_0^\infty e^{-\rho\tau} \tau^{a-1+N_0} (\tau - \gamma_1)^j \mathrm{d}\tau,$$

which can be computed using the known recurrence relation:

$$P_0^1(\gamma_1) = P_1^1(\gamma_1) = 1,\ P_{j+1}^1(\gamma_1) = (j+1) P_j^1(\gamma_1) + \gamma_1 P_{j-1}^1(\gamma_1).$$

Verdict: Due to time constraints, we were not able to test this method. However, the error bounds given in [48] imply that this method could marginally improve on the performance of the asymptotic series methods of Section 3.5. Hence, this method seems to be worth testing as an aspect of further work, in order to evaluate it.

G.5 Other quadrature methods

- **Splitting the integral in** (3.25)**:** The method discussed in Section 3.6 was that of applying Gauss-Jacobi quadrature to the integral in (3.25). An alternative method involves splitting the integral into different segments, applying substitutions to each segment, and using these to repose the problem as a sum of several integrals.

  For example, splitting the integral in (3.25) into two equal segments yields

$$\int_0^1 e^{zt} t^{a-1} (1-t)^{b-a-1} \mathrm{d}t = \int_0^{1/2} e^{zt} t^{a-1} (1-t)^{b-a-1} \mathrm{d}t + \int_{1/2}^1 e^{zt} t^{a-1} (1-t)^{b-a-1} \mathrm{d}t$$

for $\mathrm{Re}(b) > \mathrm{Re}(a) > 0$. Applying the substitution $t \mapsto \frac{1}{4}(t_1+1)$ to the first integral and $t \mapsto \frac{1}{4}(t_2+3)$ to the second yields

$$\int_0^1 e^{zt}t^{a-1}(1-t)^{b-a-1}\mathrm{d}t = \frac{1}{4}\left[\int_{-1}^1 e^{\frac{1}{4}z(t_1+1)}\left(1-\frac{t_1+1}{4}\right)^{b-a-1}\left(\frac{t_1+1}{4}\right)^{a-1}\mathrm{d}t_1 + \int_{-1}^1 e^{\frac{1}{4}z(t_2+3)}\left(1-\frac{t_2+3}{4}\right)^{b-a-1}\left(\frac{t_2+3}{4}\right)^{a-1}\mathrm{d}t_2\right].$$

Extending this to splitting the interval of integration into n intervals of integration, and applying the substitutions $t \mapsto \frac{1}{2j}(s+\{2j-1\}),\ j=1,...,n$ to the j-th integral and then replacing all of the variables with t again after the substitutions have been applied, we obtain, for $\mathrm{Re}(b) > \mathrm{Re}(a) > 0$, the expression

$$M(a;b;z) = \frac{\Gamma(b)}{\Gamma(a)\Gamma(b-a)} \times \frac{1}{(2n)^{b-1}}\sum_{j=1}^{n}\int_{-1}^1 e^{\frac{1}{2n}z(t+\{2j-1\})}(\{2n+1-2j\}-t)^{b-a-1}(\{2j-1\}+t)^{a-1}\mathrm{d}t,$$

at which point Gauss-Jacobi quadrature can be applied to each of the n integrals.

An extension to this method, and one which is found to be more useful, is to split off two small intervals on $[0,1]$, one on each side of the integration interval, therefore splitting the interval $[0,1]$ into $[0,\lambda]$, $[\lambda,1-\lambda]$ and $[1-\lambda,1]$, where $0 < \lambda \leq \frac{1}{2}$. Then, one makes the substitution $t \mapsto \frac{1}{2}\lambda(t_1+1)$ to the integral on $[0,\lambda]$ and the substitution $t \mapsto \frac{1}{2}\lambda(t_2-1)+1$ to the integral on $[1-\lambda,1]$. This yields the following integral expression when we re-write the integration variables t_1 and t_2 as t:

$$M(a;b;z) = \frac{\Gamma(b)}{\Gamma(a)\Gamma(b-a)}\left[\left(\frac{\lambda}{2}\right)^{b-1}e^{\frac{\lambda z}{2}}\int_{-1}^1 e^{\frac{1}{2}\lambda zt}\left(\frac{2-\lambda}{\lambda}-t\right)^{b-a-1}(1+t)^{a-1}\mathrm{d}t + \int_{\lambda}^{1-\lambda} e^{zt}(1-t)^{b-a-1}t^{a-1}\mathrm{d}t + \left(\frac{\lambda}{2}\right)^{b-1}e^{z\left(1-\frac{\lambda}{2}\right)}\int_{-1}^1 e^{\frac{1}{2}\lambda zt}(1-t)^{b-a-1}\left(\frac{2-\lambda}{\lambda}+t\right)^{a-1}\mathrm{d}t\right]. \tag{G.10}$$

The motivation behind this method is that two relatively small intervals at either end of $[0,1]$ can be separated, transformed, and have Gauss-Jacobi quadrature applied to

them; but the middle integral no longer has infinite values of the integrand at the end-points, so a variety of different methods can be applied to this integral (such as the composite trapezoidal or composite Simpson's rules, or the built-in MATLAB routine 'quad'), with precision as high as the user desires, as opposed to having to apply Gauss-Jacobi quadrature to the entire interval. This way, any error from the Gauss-Jacobi quadrature computations will have a reduced effect if the integral over the interval $[\lambda, 1-\lambda]$ is computed accurately.

Verdict: One major disadvantage of using this method is that computation takes at least 3 seconds, which is much longer than using Gauss-Jacobi quadrature. The accuracy we obtained was reasonable; 6 digit accuracy was obtained for computing $M(20; 40; 10+5i)$, a relatively difficult case, when the composite trapezoidal rule was applied for computing the integral on $[\lambda, 1-\lambda]$ when $\lambda = 0.1$, and 300 mesh points were used to compute each of the three integrals in (G.10). However, results obtained were similar to those obtained using Gauss-Jacobi quadrature directly, so this method is not as powerful because of the computation time.

- **Adaptive quadrature:** The idea of adaptive quadrature is to adjust the step-size at which the numerical integration of (3.25) is being carried out, depending on the accuracy that is being generated; if the solution generated is deemed not to be accurate enough, the step-size is reduced, and if the solution is very accurate, a larger step-size can be taken. In our computations, this was done by taking an initial step-size h and using two numerical methods to compute the value of the integral taken between the start point of the integral $\hat{a}$ and the next point $\hat{a}+h$. If the results generated by these two methods differ by a number greater than a specified tolerance tol_1, the step-size h is halved, and if the results differ by a number less than another specified tolerance tol_2, the step-size is doubled. Once one finds a step-size that results in the two methods differing by a number less than tol_1 but greater than tol_2, the result obtained by integrating in $[\hat{a}, \hat{a}+h]$ using the theoretically more accurate quadrature method is recorded, the integration is then computed in a similar way starting from $\hat{a}+h$, and so on until the end point $\hat{b}$ is reached (the last step-size is restricted so that we never pass the point $\hat{b}$ in the numerical integration).

Two routines were written for this method. The first used the trapezoidal rule,

$$\int_{\hat{a}}^{\hat{b}} f(x)\mathrm{d}x \approx \frac{\hat{b}-\hat{a}}{2}[f(\hat{a})+f(\hat{b})], \tag{G.11}$$

with known error bound (i.e. bound of the numerical solution subtracted from the exact solution) $-\frac{(\hat{b}-\hat{a})^3}{12}f''(x_*)$, for some $x_* \in [a,b]$ and Simpson's rule,

$$\int_{\hat{a}}^{\hat{b}} f(x)\mathrm{d}x \approx \frac{\hat{b}-\hat{a}}{6}\left[f(\hat{a})+4f\left(\hat{a}+\frac{1}{2}(\hat{b}-\hat{a})\right)+f(\hat{b})\right], \tag{G.12}$$

with known error bound $-\frac{(\hat{b}-\hat{a})^5}{2880}f^{(\mathrm{iv})}(x_*)$. The second routine used the trapezoidal rule along with Boole's law,

$$\begin{aligned}\int_{\hat{a}}^{\hat{b}} f(x)\mathrm{d}x \approx \frac{\hat{b}-\hat{a}}{90}\Bigg[&7f(\hat{a})+32f\left(\hat{a}+\frac{1}{4}(\hat{b}-\hat{a})\right)\\ &+12f\left(\hat{a}+\frac{1}{2}(\hat{b}-\hat{a})\right)+32f\left(\hat{a}+\frac{3}{4}(\hat{b}-\hat{a})\right)+7f(\hat{b})\Bigg],\end{aligned} \tag{G.13}$$

with known error bound $-\frac{(\hat{b}-\hat{a})^7}{1935360}f^{(\mathrm{vi})}(x_*)$.

Verdict: This method worked fairly well for cases with reasonably small $|a|$, $|b|$ and $|z|$. For example $M(10;15;0.5)$ was computed with 12 digit accuracy when $tol_1 = 10^{-10}$ and $tol_2 = 10^{-15}$, and the initial step-size was set to 0.001. However in cases where $|a|$, $|b|$ and $|z|$ are larger, this method was not as effective; for example $M(20;40;0.5)$ was only computed to 5 digit accuracy, as opposed to 12 digit accuracy when Gauss-Jacobi quadrature was used with 500 mesh points. This, coupled with the restriction that the method can only be applied if a and b are such that there is no blow up of the integrand of (3.25) at the end-points, renders this method unsuitable for our purposes.

- **Romberg integration:** This is a quadrature method that involves repeatedly applying Richardson extrapolation, which is a procedure designed to result in faster convergence of the quadrature method to which it is being applied, to the trapezoidal rule. It is defined as

$$R(0,0) = \frac{1}{2}(b-a)[f(a)+f(b)], \tag{G.14}$$

$$R(n,0) = \frac{1}{2}R(n-1,0) + h_n\sum_{k=1}^{2^n-1} f(a+(2k-1)h_n), \quad n\geq 1,\ h_n = \frac{b-a}{2^n}, \tag{G.15}$$

$$R(n,m) = \frac{1}{4^m-1}[4^m R(n,m-1) - R(n-1,m-1)],\ n\geq 1,\ m\geq 1. \tag{G.16}$$

The method has the known error property

$$E(n,m) = O(h_n^{2^{m+1}}), \tag{G.17}$$

where $E(n,m)$ denotes the value of the difference between the numerical approximation and the exact solution.

We note that $R(n,0)$ corresponds to the composite trapezoidal rule with $2^{n-1}+1$ mesh points, and $R(n,1)$ corresponds to the composite Simpson's rule with the same number of mesh points.

Verdict: This method was seen to be fairly effective in comparison to other integral methods described in this appendix; for example, 14 digit accuracy was generated for the computation of $M(20;40;0.5)$, $M(20;40;5)$, $M(20;40;50)$ and $M(30;70;50)$ when $m = n = 10$. Furthermore, the computation times were approximately 0.2 seconds. However, the effectiveness of this method was restricted by the constraint that the integrand needed to be effectively computed by the trapezoidal rule initially, in other words, it did not blow up at the end-points. As this could only be guaranteed for a small range of parameters, we do not recommend this as a sufficiently robust method.

- **Method for oscillatory integrals:** We obtain from [55] the following expression for an oscillatory integral:

$$\int_{-1}^{1} r(t)e^{i\omega t}\mathrm{d}x \approx \sum_{k=1}^{N_{mesh}} \Big[i^{k-1}(2k-1)\sqrt{\frac{\pi}{2\omega}}J_{k-\frac{1}{2}}(\omega) \times \sum_{j=1}^{N_{mesh}} w_{j-1}P_{k-1}(x_{j-1})r(x_{j-1})\Big], \quad \omega \neq 0, \tag{G.18}$$

where J_ν is the Bessel function of first order as defined in (2.4), $P_k(x)$ are the Legendre polynomials as defined in Appendix D, w_k are the weights for Gauss quadrature defined as $w_k = \frac{2}{(1-x_k)^2[P'_{N_{mesh}}(x_k)]^2}$, where x_k are the nodes for Gauss quadrature defined as the k-th root of $P_k(x)$, and N_{mesh} denotes the number of mesh points used for evaluating the integral in (G.18).

We now consider the integral of (3.25), and rewrite it as follows:

$$\int_{-1}^{1} e^{z\left(\frac{1}{2}t+\frac{1}{2}\right)}(1-t)^{b-a-1}t^{a-1}\mathrm{d}t = e^{z/2}\int_{-1}^{1} \underbrace{e^{\mathrm{Re}(z)t/2}(1-t)^{b-a-1}t^{a-1}}_{r(t)} \underbrace{e^{i\mathrm{Im}(z)t/2}}_{e^{i\omega t},\ \omega=\frac{\mathrm{Im}(z)}{2}} \mathrm{d}t, \tag{G.19}$$

so that the formula (G.18) can now be applied.

Verdict: We found that using the built-in MATLAB routine `legendre.m` to compute the Legendre polynomials resulted in the computation time being prohibitively large before N_{mesh} was raised to a number high enough to generate accurate results. However, if a faster routine can be written for this method, it would be worth exploring due to the problems, noted in Sections 3.9 and 5, associated with large imaginary parts of the variable z in the confluent hypergeometric function ${}_1F_1(a;b;z)$.

- **Other integral representations for $M(a;b;z)$ or $U(a;b;z)$:** Further work on computation of the confluent hypergeometric function using quadrature methods could involve other less widely examined and more complicated integrals for $M(a;b;z)$ and $U(a;b;z)$. Some examples of such integrals, taken from [3, 63, 64], include the following line integrals:

$$M(a;b;-z) = \frac{\Gamma(b)}{\Gamma(a)} z^{\frac{1}{2}-\frac{1}{2}b} \int_0^\infty e^{-t} t^{a-\frac{1}{2}b-\frac{1}{2}} J_{b-1}(2\sqrt{zt})\mathrm{d}t, \quad \mathrm{Re}(a) > 0,$$
$$U(a;b;z) = \frac{1}{\Gamma(a)} \int_0^\infty e^{-zt} t^{a-1}(1+t)^{b-a-1}\mathrm{d}t, \quad \mathrm{Re}(a) > 0, \quad |\arg z| < \frac{1}{2}\pi,$$

the contour integral valid for $b - a \neq 1, 2, 3, ..., \ \mathrm{Re}(a) > 0$,

$$M(a;b;z) = \frac{1}{2\pi i} \frac{\Gamma(1+a-b)}{\Gamma(a)\Gamma(b)} \int_0^{(1+)} e^{zt} t^{a-1}(t-1)^{b-a-1}\mathrm{d}t,$$

which starts at 0, traverses anti-clockwise around 1 and returns to 0, and the **Mellin-Barnes integral**,

$$M(a;b;-z) = \frac{1}{2\pi i \Gamma(a)\Gamma(b)} \int_{-i\infty}^{+i\infty} \frac{\Gamma(a+t)\Gamma(-t)}{\Gamma(b+t)} z^t \mathrm{d}t, \quad |\arg z| < \frac{1}{2}\pi,$$

where $a \neq 0, -1, -2, ...$, and the poles of $\Gamma(a+t)$ and $\Gamma(-t)$ must be separated by the contour of integration.

Further information on computing $U(a;b;z)$ using the trapezoidal rule is given in [4].

G.6 Other differential equation methods

- **Dormand-Prince method:** A linear multistep method is said to be **consistent** if

$$\sum_{j=1}^{i-1} a_{ij} = c_i, \quad i = 2, ..., s,$$

where, for $y' = f(t, y)$, the method reads

$$\begin{aligned} y_{n+1} &= y_n + h\sum_{i=1}^{s} b_i k_i, \\ k_1 &= f(t_n, y_n), \\ k_2 &= f(t_n + c_2 h, y_n + a_{21} h k_1), \\ k_3 &= f(t_n + c_3 h, y_n + a_{31} h k_1 + a_{32} h k_2), \\ &\vdots \\ k_s &= f(t_n + c_s h, y_n + a_{s1} h k_1 + a_{s2} h k_2 + ... + a_{s,s-1} h k_{s-1}). \end{aligned}$$

One consistent method that we studied, the Dormand-Prince 5th order method [19], reads as follows:

$$\begin{aligned} y_{n+1} &= y_n + h\left(\frac{35}{384}k_1 + \frac{500}{1113}k_3 + \frac{125}{192}k_4 - \frac{2187}{6784}k_5 + \frac{11}{84}k_6\right), \qquad \text{(G.20)} \\ k_1 &= f(t_n, y_n), \\ k_2 &= f\left(t_n + \frac{1}{5}h, y_n + \frac{1}{5}hk_1\right), \\ k_3 &= f\left(t_n + \frac{3}{10}h, y_n + \frac{3}{40}hk_1 + \frac{9}{40}hk_2\right), \\ k_4 &= f\left(t_n + \frac{4}{5}h, y_n + \frac{44}{45}hk_1 - \frac{56}{15}hk_2 + \frac{32}{9}hk_3\right), \\ k_5 &= f\left(t_n + \frac{8}{9}h, y_n + \frac{19372}{6561}hk_1 - \frac{25360}{2187}hk_2 + \frac{64448}{6561}hk_3 - \frac{212}{729}hk_4\right), \\ k_6 &= f\left(t_n + h, y_n + \frac{9017}{3168}hk_1 - \frac{355}{33}hk_2 + \frac{46732}{5247}hk_3 + \frac{49}{176}hk_4 - \frac{5103}{18656}hk_5\right). \end{aligned}$$

We program this method into MATLAB for the differential equation (3.2), with initial conditions given in (3.28), and values at $z = h$, where h is the first mesh point past the initial point $z = 0$, obtained by taking Taylor series expansions of ${}_1F_1(a; b; z)$ and its derivative about $z = 0$.

Verdict: After implementation, this method was not found to be as effective as the RK4 method as described in Section 3.7; across most parameter regimes, the method generated less accurate solutions when a relatively small number of mesh points was used in both methods. For example, when both methods were applied with 500 mesh

points to calculate $M(2;5;4)$, they only yielded 3 digit accuracy when applying the Dormand-Prince method, as opposed to 10 digit accuracy for the RK4 method. For this reason, the RK4 method was preferred to this method.

- **More general differential equation:** As stated in [36], the differential equation

$$hy'' + \left(\frac{2\alpha h}{z} + 2f'h - \frac{hh''}{h'} - hh' + bh'y' + \left(h'\left(\frac{\alpha}{z} + f'\right)(b-h)\right.\right. \tag{G.21}$$
$$\left.\left. + h\left[\frac{\alpha(\alpha-1)}{z^2} + \frac{2\alpha f'}{z} + f'' + (f')^2 - \frac{h''}{h'}\left(\frac{\alpha}{z} + f'\right)\right] - a(h')^2\right)y = 0,$$

is satisfied by

$$y = z^{-a}e^{-f(z)}\ {}_1F_1(a;b;h(z)).$$

Using this relation could provide a more general approach to finding a larger class of hypergeometric functions.

Verdict: Although this differential equation covers a more general case, it is not required for the purposes of this investigation, and the equation will result in far more complex numerical schemes for computation than those for solving the differential equation (3.2). As a possible extension to the work carried out in this project, we could examine a variety of methods for solving this differential equation, whose solutions represent a more general class of functions.

G.7 Padé approximants and rational approximation

- **Padé approximants:** For this method, we write $M(a;b;z)$ as:

$$M(a;b;z) = \sum_{j=0}^{\infty} \frac{(a)_j}{(b)_j}\frac{z^j}{j!} = \sum_{j=0}^{\infty} A_j z^j. \tag{G.22}$$

We approximate (G.22) as $M(a;b;z) \approx \frac{n_0+n_1 z+\ldots+n_{\tilde{p}} z^{\tilde{p}}}{1+d_1 z+\ldots+d_{\tilde{q}} z^{\tilde{q}}}$, where:

$$\left[\begin{array}{cccccc|ccccc} 1 & 0 & 0 & \cdots & \cdots & 0 & 0 & \cdots & \cdots & 0 \\ 0 & 1 & 0 & \cdots & \cdots & 0 & -A_0 & \ddots & & 0 \\ 0 & 0 & 1 & \cdots & \cdots & 0 & -A_1 & -A_0 & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & & \vdots & -A_2 & -A_1 & \ddots & 0 \\ \vdots & \vdots & \vdots & & \ddots & 0 & \vdots & -A_2 & \ddots & -A_0 \\ 0 & 0 & 0 & \cdots & 0 & 1 & \vdots & \vdots & \ddots & -A_1 \\ \hline 0 & 0 & 0 & \cdots & \cdots & 0 & \vdots & \vdots & & -A_2 \\ \vdots & \vdots & \vdots & \ddots & & \vdots & \vdots & \vdots & & \vdots \\ \vdots & \vdots & \vdots & & \ddots & \vdots & -A_{n+d} & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & \cdots & 0 & -A_{\tilde{p}+\tilde{q}+1} & \cdots & \cdots & -A_{\tilde{p}} \end{array}\right] \left[\begin{array}{c} n_0 \\ n_1 \\ \vdots \\ \vdots \\ \vdots \\ n_{\tilde{p}} \\ d_1 \\ d_2 \\ \vdots \\ \vdots \\ d_{\tilde{q}} \end{array}\right] = \left[\begin{array}{c} A_0 \\ A_1 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ A_{\tilde{p}+\tilde{q}} \end{array}\right]. \quad \text{(G.23)}$$

As pointed out in [57] and as verified by our testing, the matrix in (G.23) often has a very low condition number (frequently less than 10^{-20}), especially for large $\tilde{p}$ and $\tilde{q}$. It is recommended in [57] therefore that the above system be solved by performing an LU decomposition.

Verdict: There is not an easy method of error control with this method without solving a large number of matrix systems. Unless we took large $\tilde{p}$ and $\tilde{q}$, we found that the method generated accurate solutions only for moderate parameter regimes (with, as a guide, $|a|, |z| \lesssim 15$). For example, $M(1;3;2)$ is computed to 15 digit accuracy, but $M(10;30;20)$ is only computed to 3 digit accuracy when $\tilde{p} = 15$, $\tilde{q} = 3$. The computation time was reasonable (about 0.25 seconds), but the lack of possibilities for error control means that we do not recommend this approach for use in its own right.

- **Rational approximation for $U(a;1+a-b;z)$:** In [37], the following expression is given for $U(a,1+a-b;z)$:

$$U(a;1+a-b;z) = z^{-a} \lim_{m\to\infty} \frac{N_m(z)}{D_m(z)}, \quad \text{(G.24)}$$

where

$$N_m(z) = \sum_{j=0}^{m} \Big[\frac{(-m)_j (m+1)_j (a)_j (b)_j}{(a+1)_j (b+1)_j (m!)^2} \times {}_3F_3(-m+j, m+1+j, 1; 1+j, a+1+j, b+1+j; -z) \Big],$$

$$D_m(z) = {}_2F_2(-m, m+1; a+1, b+1; -z).$$

This was computed in MATLAB using Taylor series methods for evaluating the hypergeometric functions ${}_3F_3$ and ${}_2F_2$.

Verdict: When we coded and tested this method, we found that, due to the fact that a sum of more complex hypergeometric functions (by which we mean hypergeometric functions with larger p and q in the notation (2.1)) than ${}_1F_1(a;b;z)$ is being computed, the computation time was much longer than for other series methods used to compute the confluent hypergeometric function. The method was also found to be less accurate than many others investigated; for example, when we tried to compute $U(10;20;30)$ using Taylor series methods for ${}_2F_2$ and ${}_3F_3$, not a single digit of accuracy was obtained. We conclude that this method is not a viable one for computing $U(a;b;z)$ unless a substantial amount of research into computing ${}_2F_2$ and ${}_3F_3$ is carried out first.

- **Rational approximation for** ${}_1F_1(a;c;-z)$: In [35], $E(z)$ is set to be ${}_1F_1(a;b;-z)$, which is then written as

$$
\begin{aligned}
E(z) &= \frac{A_n(z)}{B_n(z)} + R_n(z), \qquad \text{(G.25)}\\
A_n(z) &= L_n z^n \sum_{k=0}^{n} \left[\frac{(-n)_k (n+1)_k (a)_k}{(a+1)_k (k!)^2} \times \hat{A}_n^k \right],\\
\hat{A}_n^k &= {}_4F_2\left(-n+k, n+1+k, b+k, 1; 1+k, a+1+k; -\frac{1}{z} \right),\\
B_n(z) &= L_n z^n {}_3F_1\left(-n, n+1, b; a+1; -\frac{1}{z} \right),\\
L_n &= \frac{(a+1)_n}{(n+1)_n (b)_n},
\end{aligned}
$$

which can then be programmed using series methods.

Verdict: This method involves computing a sum of more complex hypergeometric functions than ${}_1F_1(a;b;z)$. There is therefore too much computational work involved for this method to be efficient in comparison to some of the methods discussed in the main body of this dissertation.

G.8 Other expansions for ${}_1F_1(a;b;z)$

- **Chebyshev expansion:** In [38], $M(a;b;z)$ is represented as

$$M(a;b;z) = \sum_{j=0}^{\infty} C_j(\omega)\hat{T}_j\left(\frac{z}{\omega}\right),\ 0 \le \frac{z}{\omega} \le 1, \tag{G.26}$$

where

$$\begin{aligned} C_j(\omega) &= \frac{\epsilon_j (a)_j \omega^j}{2^{2j}(b)_j j!}\ {}_2F_2\left(a+j, \frac{1}{2}+j; b+j, 1+2j; \omega\right), \\ \epsilon_0 &= 1,\ \epsilon_j = 2, \quad j = 1, 2, \ldots\ , \\ \frac{2C_j(\omega)}{\epsilon_j} &= \frac{4(j+1)}{\omega} C_{j+1}(\omega) + C_{j+2}(\omega), \quad j = 1, 2, \ldots\ . \end{aligned}$$

Verdict: As this method of computing this expansion involves evaluating the sum of more complex hypergeometric functions, time constraints prevented us from implementing this method. As discussed in Section 5, an avenue of future work could be to carry out research on ${}_2F_2$ in order to subsequently attempt this computation.

- **Expansion in terms of Bessel functions:** The function ${}_1F_1(a;b;z)$ has a number of expansions as series of Bessel functions. Two of the most useful, detailed in [36], are

$$\begin{aligned} {}_1F_1(a;b;z) = \Gamma\left(a+\frac{1}{2}\right) e^{z/2} \left(\frac{4}{z}\right)^{a-\frac{1}{2}} \\ \times \sum_{j=0}^{\infty} \frac{(j+a-\frac{1}{2})(2a-1)_j (2a-b)_j}{j!(b)_j (a-\frac{1}{2})} I_{j+a-\frac{1}{2}}\left(\frac{z}{2}\right) \end{aligned} \tag{G.27}$$

and

$$\begin{aligned} {}_1F_1(a;b;z) = \Gamma\left(b-a+\frac{1}{2}\right) e^{z/2} \left(\frac{4}{z}\right)^{c-a-\frac{1}{2}} \\ \times \sum_{j=0}^{\infty} \frac{(-1)^j (j+b-a-\frac{1}{2})(2b-2a-1)_j (b-2a)_j}{j!(b)_j (b-a-\frac{1}{2})} I_{j+b-a-\frac{1}{2}}\left(\frac{z}{2}\right), \end{aligned} \tag{G.28}$$

where

$$\begin{aligned} I_\nu(z) &= \frac{\left(\frac{z}{2}\right)^\nu}{\Gamma(\nu+1)}\ {}_0F_1\left(\ ;1+\nu; \frac{z^2}{4}\right) = e^{-i\nu\pi/2} J_\nu(ze^{i\pi/2}), \quad -\pi < \arg\ z \le \frac{\pi}{2}, \\ J_\nu(z) &= \frac{\left(\frac{z}{2}\right)^\nu}{\Gamma(\nu+1)}\ {}_0F_1\left(\ ;1+\nu; \frac{-z^2}{4}\right). \end{aligned}$$

Verdict: Although (G.27) and (G.28) were not computed due to time constraints, doing so could provide another potential method for computing ${}_1F_1$, as Bessel functions are known to be easier to compute than the confluent hypergeometric function itself.

G.9 Multiplication formula

A known formula, stated in [36], can be used to find the value of the confluent hypergeometric function in terms of another confluent hypergeometric function with the same parameters but with the variable of opposite sign. The formula is as follows:

$$ {}_1F_1(a;b;z) \times {}_1F_1(a;b;-z) = {}_2F_3\left(a, b-a; b, \frac{1}{2}b, \frac{1}{2}b+\frac{1}{2}; \frac{z^2}{4}\right). \tag{G.29} $$

Verdict: The study in this project of ${}_0F_1$, ${}_1F_1$ and ${}_2F_1$ suggests that computing the function ${}_2F_3$ will only be relatively simple if all a_p in (2.1) are fairly small, in other words if a and $b-a$ are small. In this case, computing ${}_1F_1(a;b;-z)$ directly should be more straightforward. However, this formula could prove useful for checking solutions obtained by using other methods.

H Other methods considered for evaluating ${}_2F_1(a,b;c;z)$

In this appendix, we discuss other methods that we considered for computing the Gauss hypergeometric function ${}_2F_1(a,b;c;z)$. We state an opinion as to the effectiveness of these methods either by researching them and deeming them unsuitable or inferior to other methods, or by finding by numerical testing that they were not as accurate or efficient as other methods detailed in Section 4. Details are set out of the background to each method, and a brief analysis of its effectiveness is stated.

H.1 Other quadrature methods

- **Splitting the integral in** (4.8)**:** We discussed in Section 4.4 the method involving applying Gauss-Jacobi quadrature to the integral in (4.8). An alternative method involved splitting the integral as discussed for ${}_1F_1$ in Appendix G.5.

  We consider the integral (4.8) for $\mathrm{Re}(c) > \mathrm{Re}(b) > 0$, and split off two small intervals on $[0,1]$, one on each side of the integration interval, hence splitting the interval $[0,1]$ into $[0,\lambda]$, $[\lambda, 1-\lambda]$ and $[1-\lambda, 1]$, where $0 < \lambda \leq \frac{1}{2}$. Then, we make the substitution $t \mapsto \frac{1}{2}\lambda(t_1+1)$ to the integral on $[0,\lambda]$ and the substitution $t \mapsto \frac{1}{2}\lambda(t_2-1)+1$ to the integral on $[1-\lambda,1]$. This yields the following integral expression when the integration variables t_1 and t_2 were re-labelled as t:

$$
\begin{aligned}
{}_2F_1(a,b;c;z) = \frac{\Gamma(c)}{\Gamma(b)\Gamma(c-b)} \qquad\qquad (\mathrm{H.1})\\
\times \Bigg[\left(\frac{\lambda}{2}\right)^{c-1} \int_{-1}^{1} \left(1 - \frac{1}{2}\lambda z\{t+1\}\right)^{-a} \left(\frac{2-\lambda}{\lambda} - t\right)^{c-b-1} (1+t)^{b-1} \mathrm{d}t \\
+ \int_{\lambda}^{1-\lambda} e^{zt}(1-t)^{c-b-1} t^{b-1} \mathrm{d}t \\
+ \left(\frac{\lambda}{2}\right)^{c-1} \int_{-1}^{1} \left(1 - z\{\frac{1}{2}\lambda(t-1)+1\}\right)^{-a} (1-t)^{c-b-1} \left(\frac{2-\lambda}{\lambda} + t\right)^{b-1} \mathrm{d}t \Bigg],
\end{aligned}
$$

  similar to the expression when we applied this same method in Appendix G.5.

  Verdict: The accuracy of this method was reasonable for many cases. For instance, ${}_2F_1(2.25, 4; 5.5; 0.5)$ was computed to 15 digit accuracy, with ${}_2F_1(20.25, 40.5; 50; 0.5)$ to 7 digit accuracy and ${}_2F_1(20.25, 4; 50.5; 0.98)$ (where here z is very close to the unit

disc) to 10 digit accuracy when 300 mesh points were used to compute each of the three integrals in (H.1), λ was taken to be 0.1, and the composite trapezoidal rule was used to compute the integral on $[\lambda, 1-\lambda]$. However, similarly to when we applied this method to compute ${}_1F_1$, we found that the time taken was very large compared to the time taken for Gauss-Jacobi quadrature as described in Section 4.4 (the computation time was usually over 3 seconds the first time we ran this code after loading MATLAB), and for this reason, we preferred the method of Gauss-Jacobi quadrature.

- **Romberg integration:** We implement Romberg integration, the theory of which was introduced in Appendix G.5, and apply it to the integral (4.8).

  Verdict: The results generated using this method were very accurate in many cases. For example, when $m = n = 10$ was taken in (G.15) and (G.16), we obtained 15 digit accuracy when computing ${}_2F_1(2, 4.5; 10.25; 0.5)$, and 12 digit accuracy for ${}_2F_1(2, 4.5; 10.25; 0.9)$ and ${}_2F_1(20, 4.5; 10.25; 0.5)$. On the other hand, when we computed ${}_2F_1(20, 40.5; 10.25; 0.5)$, we did not obtain any digits of accuracy, which underlines the limitations of the method. This, and the restriction that this method cannot be applied when the integrand in (4.8) blows up at the end-points of the integral, are quite serious drawbacks.

- **Adaptive quadrature:** We apply the theory of adaptive quadrature introduced in Appendix G.5, but apply it this time to the integral (4.8).

  Verdict: For some cases, this method was very effective. For example, when tol_1 and tol_2 were taken to be 10^{-10} and 10^{-15} respectively, in the notation of G.5, we obtained 13 digit accuracy when we computed ${}_2F_1(1, 2; 5.5; 0.5)$ and ${}_2F_1(10, 2; 5.5; 0.5)$. However, some computations such as ${}_2F_1(10, 20; 5.5; 0.5)$ took a prohibitively long time (over 2 minutes) to carry out using this method, which is a significant disadvantage to using adaptive quadrature for computing ${}_2F_1$. Also, as for ${}_1F_1$, we require that the integrand in (4.8) is not infinitely large at the end-points of the integral. These two reasons justified our decision to rule out this method in favour of Gauss-Jacobi quadrature, which is detailed in Section 4.4.

- **Other integral representations for ${}_2F_1(a,b;c;z)$:** Further methods for numerical integration could be considered by applying them to other integrals for ${}_2F_1(a,b;c;z)$. Some examples of contour integrals which could be computed numerically, introduced in [3, 65], are stated below:

$$
\begin{aligned}
{}_2F_1(a,b;c;z) &= \frac{\Gamma(c)\Gamma(1+b-c)}{2\pi i\Gamma(b)}\int_0^{(1+)}\frac{t^{b-1}(t-1)^{c-b-1}}{(1-zt)^a}\mathrm{d}t, \\
&\qquad\qquad c-b\neq 1,2,3,..., \quad \mathrm{Re}(b)>0, \\
{}_2F_1(a,b;c;z) &= \frac{e^{-b\pi i}}{2\pi i}\frac{\Gamma(c)\Gamma(1-b)}{\Gamma(c-b)}\int_\infty^{(0+)}\frac{t^{b-1}(t+1)^{a-c}}{(t-zt+1)^a}\mathrm{d}t, \\
&\qquad\qquad b\neq 1,2,3,..., \quad \mathrm{Re}(c-b)>0, \\
{}_2F_1(a,b;c;z) &= \frac{\Gamma(c)}{2\pi i\Gamma(a)\Gamma(b)}\int_{-i\infty}^{+i\infty}\frac{\Gamma(a+t)\Gamma(b+t)\Gamma(-t)}{\Gamma(c+t)}(-z)^t\mathrm{d}t, \\
&\qquad\qquad a,b\neq 0,-1,-2,... \ , \\
{}_2F_1(a,b;c;z) &= \frac{\Gamma(c)}{2\pi i\Gamma(a)\Gamma(b)\Gamma(c-a)\Gamma(c-b)} \\
&\quad \times\int_{-i\infty}^{+i\infty}\Gamma(a+t)\Gamma(b+t)\Gamma(c-a-b-t)\Gamma(-t)(1-z)^t\mathrm{d}t, \\
&\qquad\qquad a,b,c-a,c-b\neq 0,-1,-2,... \ .
\end{aligned}
$$

  The first contour integral above represents beginning at 0, traversing anti-clockwise around 1 and then back to 0, and the second represents starting at ∞, traversing anti-clockwise around 0 and then moving back towards ∞.

H.2 Other differential equation methods

- **Dormand-Prince:** We apply the Dormand-Prince method introduced in Appendix G.6 to the differential equation (4.2).

  Verdict: We found that this method was not as accurate as the RK4 method described in Section 4.5 when we used a relatively small number of mesh points. For example, when we tried to compute ${}_2F_1(1.75+3i,2+4i;5.5+2i;0.5)$ using both methods with 500 mesh points, we obtained 2 digit accuracy with the Dormand-Prince method as opposed to 10 digit accuracy with the RK4 method. Therefore, we concluded that for a number of mesh points that was feasible without resulting in a very large computation

time, the RK4 method was the more effective for solving the differential equation (4.2) numerically.

H.3 Padé approximants and rational approximation

- **Padé approximants:** We write

$$ {}_2F_1(a,b;c;z) = \sum_{j=0}^{\infty} \frac{(a)_j (b)_j}{(c)_j} = \sum_{j=0}^{\infty} A_j z^j, \tag{H.2}$$

and then solve the matrix system (G.23) as in Appendix G.7.

Verdict: As for ${}_1F_1(a;b;z)$ in Appendix G.5, we found that this Padé approximation method for computing ${}_2F_1(a,b;c;z)$ was effective for some parameter values. For example, we computed ${}_2F_1(1,3;4.5;0.5)$ to 11 digit accuracy in roughly a quarter of a second, when $\tilde{p} = 15$ and $\tilde{q} = 3$. However, when we tried to compute ${}_2F_1(5,15;4.5;0.5)$ for instance, no digits of accuracy were obtained, although 4 digit accuracy was obtained when $\tilde{p}$ was increased to 30. Similarly to when we computed ${}_1F_1$ using this method, we refrained from recommending this method due to the lack of possibilities for altering the values of $\tilde{p}$ and $\tilde{q}$ to control the error it generates.

- **Rational approximation for ${}_2F_1(a,b;c;-z)$:** We computed the representation stated in [35],

$$ E(z) = \frac{A_n(z)}{B_n(z)} + R_n(z), \tag{H.3}$$

where

$$\begin{aligned} A_n(z) &= L_n z^n \sum_{k=0}^{n} \frac{(-n)_k (n+1)_k (a)_k (b)_k k}{(a+1)_k (b+1)_k (k!)^2} \\ &\qquad \times {}_4F_3\left(-n+k, n+1+k, c+k, 1; 1+k, a+1+k, b+1+k; -\frac{1}{z}\right), \\ B_n(z) &= L_n z^n \, {}_3F_2\left(-n, n+1, c; a+1, b+1; -\frac{1}{z}\right), \\ L_n &= \frac{(a+1)_n (b+1)_n}{(n+1)_n (c)_n}. \end{aligned}$$

Verdict: We concluded that, on the basis of our studies for computing ${}_1F_1$ detailed in Appendix G.7, computing the Gauss hypergeometric function ${}_2F_1$ using (H.3) might

not be the most efficient method due to the fact that more complex hypergeometric functions are involved in this representation. Therefore this method was not implemented due to time constraints. If research is carried out on the computation of ${}_4F_3$ and ${}_3F_2$, this could be a viable method for calculating ${}_2F_1$, however it is unlikely that more accurate results could be generated for a sum of more complex hypergeometric functions than a single simpler hypergeometric function.

H.4 Chebyshev expansion for ${}_2F_1(a, b; c; z)$

As in [36], we write

$$ {}_2F_1(a, b; c; zx) = \sum_{j=0}^{\infty} C_j(z)\hat{T}_j(x), \quad 0 \leq x \leq 1, \quad z \neq 1, \quad |\arg(1-z)| < \pi, \tag{H.4} $$

where $\hat{T}_j(x)$ denote the scaled Chebyshev polynomials, and $C_j(z)$ are defined as

$$
\begin{aligned}
C_j(z) &= \frac{\epsilon_j (a)_j (b)_j z^j}{2^{2j}(c)_j j!}\ {}_3F_2\left(a+j, b+j, \frac{1}{2}+j; c+j, 1+2j; z\right), \\
\epsilon_0 &= 1,\ \epsilon_j = 2, \quad j = 1, 2, \ldots, \\
\frac{2C_j(z)}{\epsilon_j} &= (j+1)\left(2 - \frac{(2j+3)(j+a+1)(j+b+1)}{(j+2)(j+a)(j+b)} + \frac{4(j+c)}{z(j+a)(j+b)}\right) C_{j+1}(z) \\
&\quad + \frac{2}{(j+a)(j+b)}\left([(j+2-a)(j+2-b)\left(j+\frac{3}{2}\right)\right. \\
&\quad \left. -(j+3-a)(j+3-b)(j+1)] + \frac{2(j+1)(j+3-b)}{z}\right) C_{j+2}(z) \\
&\quad - \frac{(j+1)(j+3-a)(j+3-b)}{(j+2)(j+a)(j+b)} C_{j+3}(z).
\end{aligned}
$$

Verdict: Carrying out this computation would involve computing the hypergeometric function ${}_3F_2$, which we did not research in this project. We concluded that this method could potentially be a successful one, but only if a body of knowledge was built-up about the function ${}_3F_2$ first. However, it is unlikely that computing more complex hypergeometric functions will be easier than calculating simpler ones.

I Evaluation of ${}_0F_1(\ ;a;z)$ and other special functions required for this project

This appendix details key facts and methods concerning the computation of other special functions that were relevant to this project. When these functions were evaluated using built-in MATLAB routines, code from the MathWorks website `http://www.mathworks.co.uk/` or the NAG Toolbox, of the Numerical Algorithms Group, these are mentioned.

- **Confluent hypergeometric limit function:** The **confluent hypergeometric limit function** ${}_0F_1(\ ;a;z)$, required for computation for method 3 of Section 3.4 (that of computing (3.22)), is defined as the series

$$ {}_0F_1(\ ;a;z) = \sum_{j=0}^{\infty} \frac{1}{(a)_j} \frac{z^j}{j!}. \tag{I.1} $$

  The differential equation satisfied by ${}_0F_1(\ ;a;z)$, which is obtained from (2.2) when $p = 0$, $q = 1$, is given by

$$ z\frac{\mathrm{d}^2 w}{\mathrm{d}z^2} + a\frac{\mathrm{d}w}{\mathrm{d}z} - w = 0, \tag{I.2} $$

  and, as illustrated in (2.4), the Bessel function, another special function, can be defined in terms of the confluent hypergeometric limit function.

  One way to compute the function ${}_0F_1$ is by using the basic Taylor series definition. Methods (a) and (b), first discussed in Section 3.2, can be used as follows:

  Method (a): Similarly to the method (a) in Sections 3.2 and 4.2, compute

$$ A_0 = 1, \quad S_0 = A_0, $$
$$ A_{j+1} = A_j \times \frac{1}{a+j} \times \frac{z}{j+1}, \quad S_{j+1} = S_j + A_{j+1}, \quad j = 0, 1, 2, ..., $$

  terminate the summation of the series when $\frac{|A_{N+1}|}{|S_N|} < tol$ for some tol and some N, and return S_N as the solution.

Method (b): Following the method of [44] as in Sections 3.2 and 4.2, compute

$$\begin{aligned} S_{-1} = S_0 = 1, \quad S_1 = \frac{1}{a}z, \\ r_j = \frac{1}{j(a+j-1)}, \quad j = 2, 3, \dots , \\ S_j = S_{j-1} + (S_{j-1} - S_{j-2})r_j z, \quad j = 2, 3, \dots , \end{aligned}$$

terminate the summation of the series when $\frac{|S_{N+1}-S_N|}{|S_N|} < tol$ for some tol and some N, and return S_N as the solution.

Alternatively, ${}_0F_1$ has the following expression as a Bessel function [60]:

$${}_0F_1(\ ; a; z) = \Gamma(a) z^{\frac{1-a}{2}} I_{a-1}(2\sqrt{z}), \tag{I.3}$$

where $I_\nu(z)$ is defined in (3.10). The function ${}_0F_1$ can therefore also be computed using the Bessel function routines explained later in this appendix.

It was found that Taylor series method (a) generated at least 12 digit accuracy for almost all values tested, provided the condition $|z| \lesssim 1000$ held. This was therefore used rather than the Bessel function expansion, due to the fact that the built-in MATLAB function `besselj.m` will not compute the Bessel function with complex or negative real argument. However, the Bessel function expression would be useful if software were created to allow its computation for negative and complex argument, and we recommend that the Numerical Algorithms Group develop software for computing the Bessel function $J_\nu(z)$ for all $\nu, z \in \mathbb{C}$. Details on the software used for computing the Bessel function are given later in this appendix.

- **Gamma function:** We use the Gamma function, as defined in (2.5) and (2.6), in many of the methods for computing the hypergeometric functions discussed.

  It should be noted that, by (2.6), the Gamma function is singular when its argument is equal to a negative integer. This causes a problem in Section 4.6 for example, when applying transformation formulae can involve attempting to compute a finite value by adding up two infinitely large terms of opposite sign, when summing the real or imaginary part of the individual terms.

The Gamma function $\Gamma(z)$ satisfies

$$\Gamma(z+1) = z\Gamma(z), \tag{I.4}$$

which gives a useful recurrence relation for computations.

Effective code for computation of the Gamma function will take account of the reflection formula (2.6) and the recursion formula (I.4). Such code could also take account of the polynomial expansion in [3]

$$\frac{1}{\Gamma(z)} = \sum_{j=1}^{\infty} c_j z^j,$$

where the coefficients c_j are stated in [3], or the following asymptotic formula given in the same text:

$$\Gamma(z) \sim e^z z^{z-\frac{1}{2}} (2\pi)^{\frac{1}{2}} \left[1 + \frac{1}{12z} + \frac{1}{288z^2} - \frac{139}{51840z^3} - \frac{571}{2488320z^4} + ...\right], \tag{I.5}$$

as $z \to \infty$, for $|\arg z| < \pi$.

Alternatively, the following expansion, stated in [57], can be used, for certain d_j, θ and N:

$$\begin{aligned}\Gamma(z+1) = &\left(z + \theta + \frac{1}{2}\right)^{z+\frac{1}{2}} e^{-\left(z+\theta+\frac{1}{2}\right)} \\ &\times \sqrt{2\pi}\left[d_0 + \frac{d_1}{z+1} + \frac{d_2}{z+2} + ... + \frac{d_N}{z+N} + \epsilon_N\right],\end{aligned} \tag{I.6}$$

for $\mathrm{Re}(z) > 0$, where ϵ_N denotes the error in the computation for the chosen value of N. For $N = 14$ and $d_0, d_1, ..., d_{14}$ and θ stated in [57], the authors note that the error $|\epsilon_{14}|$ is less than 10^{-15} for real z and almost as small for complex z.

It is also recommended in [57] that $\log \Gamma(z)$, rather than $\Gamma(z)$ directly, should be computed, due to the greater possibility of overflow when z is increased if direct evaluation of $\Gamma(z)$ is used.

For this project, we use the built-in MATLAB function `gamma.m` for real z, and the code `cgama.m` from [71], based on ideas from [70], for complex z. The NAG Toolbox code `s14aa.m` can be used, but only for a real variable. We recommend to the Numerical Algorithms Group that code for the Gamma function for complex variable z be

designed to feed into routines for the evaluation of hypergeometric functions for complex parameters, and that such code might benefit by using some of the ideas discussed in this appendix.

One reason why the methods explained in this project that involve Gamma functions are restricted by the capabilities of MATLAB is that the Gamma function is read out as infinity for sufficiently large z (for example, according to MATLAB, $\Gamma(171)$ is finite, but $\Gamma(172)$ is infinite).

- **Incomplete gamma function:** The two incomplete gamma functions used in this project, which were required in Appendix G.2 and G.4, are:

$$\gamma(a,z) = \int_0^z e^{-t}t^{a-1}\mathrm{d}t, \quad \Gamma(a,z) = \int_z^\infty e^{-t}t^{a-1}\mathrm{d}t, \tag{I.7}$$

from which we deduce that:

$$\gamma(a,z) + \Gamma(a,z) = \int_0^\infty e^{-t}t^{a-1}\mathrm{d}t = \Gamma(a). \tag{I.8}$$

It is stated in [3] that $\gamma(a,z)$ and $\Gamma(a,z)$ satisfy the following recurrence relations:

$$\begin{aligned} \gamma(a+1,z) &= a\gamma(a,z) - z^a e^{-z}, \\ \Gamma(a+1,z) &= a\Gamma(a,z) + z^{-a}e^{-z}. \end{aligned} \tag{I.9}$$

As noted in [3],the following is a series definition for $\gamma(a,z)$:

$$\gamma(a,z) = e^{-z}z^a \sum_{j=0}^{\infty} \frac{\Gamma(a)}{\Gamma(a+1+j)} z^j. \tag{I.10}$$

We note that, as first indicated in Section 3.2, the $(k+1)$-st term can be computed from the k-th term using multiplication by an appropriate factor in terms of k, so this series is relatively simple to evaluate.

Apart from determining the function $\Gamma(a,z)$ from $\gamma(a,z)$ using (I.8), the following continued fraction representations stated in [57] can be used for $z > 0$:

$$\begin{aligned} \Gamma(a,z) &= e^{-z}z^a \left(\frac{1}{z+} \frac{1-a}{1+} \frac{1}{z+} \frac{2-a}{1+} \frac{2}{z+} \ldots \right) \equiv e^{-z}z^a \times \frac{1}{z + \frac{1-a}{1+\frac{1}{z+\ldots}}}, \\ \Gamma(a,z) &= e^{-z}z^a \left(\frac{1}{z+1-a-} \frac{1\cdot(1-a)}{z+3-a-} \frac{2\cdot(2-a)}{z+5-a-} \cdots \right). \end{aligned} \tag{I.11}$$

A good routine for computing the incomplete gamma functions should contain some of the ideas above. The built-in MATLAB function used for computation of the function $\gamma(a, z)$ is `gammainc.m`, in terms of variable z and parameter a. This needs to be multiplied by $\Gamma(a)$ to obtain the function as defined in (I.7), and in turn subtracting this from $\Gamma(a)$ gives $\Gamma(a, z)$ as defined in (I.7). The NAG Library has the routine `s14ba.m` for computing the incomplete gamma function in normalised form [i.e. divided by $\Gamma(a)$]. However this along with the MATLAB routine only works for real a. For complex a, we could use the known relation

$$\gamma(a, z) = a^{-1} z^a e^{-z} M(1; a + 1; z) = a^{-1} z^a M(a; a + 1; -z). \tag{I.12}$$

We recommend that a routine for computing the two incomplete gamma functions $\gamma(a, z)$ and $\Gamma(a, z)$ for complex a be devised for the NAG Toolbox as a supplement to the code for evaluating hypergeometric functions.

- **Polygamma function:** The polygamma function, $\psi(z)$, is defined as in (4.26), or alternatively, for $n = 0, 1, 2, ...$

$$\psi^{(n)}(z) = \frac{\mathrm{d}^n}{\mathrm{d}z^n}[\log \Gamma(z)]. \tag{I.13}$$

  The routines used for the computation of this function are the NAG Toolbox routines `s14ae.m` and `s14af.m`, for real and complex z, respectively.

- **Bessel functions:** The built-in MATLAB function `besselj.m`, in terms of parameter ν and variable z, was used to compute the Bessel functions $J_\nu(z)$ and $I_\nu(z)$, as defined in (2.4) and (3.9) respectively, which arose in this project. Alternatively, the NAG Toolbox routines `s17ae.m`, `s17de.m` or `s18gk.m`, could be used, but these only work for specific regimes of ν.

J Some examples of hypergeometric functions from practical applications

There are a large number of practical applications for the hypergeometric functions ${}_1F_1(a;b;z)$ and ${}_2F_1(a,b;c;z)$. One key application that we have previously indicated is the expression of other elementary and special functions as a special case of these hypergeometric functions. A selection of other practical applications of the confluent and Gauss hypergeometric functions is listed below. In all examples, the notation from the relevant literature is used.

- The confluent hypergeometric function ${}_1F_1$ can be used to find exact solutions of the wave equation. As stated in [31], in paraboloidal coordinates,

$$x = 2\sqrt{\xi\eta}\cos\phi,\ y = 2\sqrt{\xi\eta}\sin\phi,\ z = \xi - \eta,$$

separating the variables of the Helmholtz equation $\nabla^2 u + k^2 u = 0$ as follows:

$$u = f_1(\xi)f_2(\eta)e^{ip\phi} = \underbrace{\xi^{-\frac{1}{2}}W^{(1)}_{k,\frac{1}{2}p}(2ik\xi)}_{f_1(\xi)}\underbrace{\eta^{-\frac{1}{2}}W^{(2)}_{k,\frac{1}{2}p}(-2ik\eta)}_{f_2(\eta)}e^{ip\phi} \tag{J.1}$$

will enable its solution to be found. Here, $W^{(1)}_{\kappa,\mu}(z)$, $W^{(2)}_{\kappa,\mu}(z)$ denote two solutions to **Whittaker's equation**

$$\frac{\mathrm{d}^2W}{\mathrm{d}z^2} + \left(-\frac{1}{4} + \frac{\kappa}{z} + \frac{\frac{1}{4}-\mu^2}{z^2}\right)W = 0, \tag{J.2}$$

whose two standard solutions are

$$\begin{aligned} M_{\kappa,\mu}(z) &= e^{-\frac{1}{2}z}z^{\frac{1}{2}+\mu}M\left(\frac{1}{2}+\mu-\kappa;1+2\mu;z\right),\ 2\mu \neq -1,-2,..., \\ W_{\kappa,\mu}(z) &= e^{-\frac{1}{2}z}z^{\frac{1}{2}+\mu}U\left(\frac{1}{2}+\mu-\kappa;1+2\mu;z\right), \end{aligned}$$

details of which are described in [3, 14].

- In [10], the authors discuss writing a program designed to calculate cross sections for the scattering of charged particles, and note the applicability of these calculations to

problems in atomic and molecular physics. The discussion is based around seeking the solution to the following differential equation, which is also discussed in [47]:

$$\frac{\mathrm{d}^2u}{\mathrm{d}r^2} + \left[k^2 + \frac{2Z}{r} - \frac{L(L+1)}{r^2}\right] u = 0, \tag{J.3}$$

where r is the distance between the locations of the projectile and the target, Z is the charge product, k^2 denotes the energy of the charged particle, and L is an integer which is implicit in the definition of the Coulomb function as the solution u of (J.3).

By writing $c = \frac{ik}{Z}$ and $z = Zr$ as in [10], the equation (J.3) can be written as a form of (J.2),

$$\frac{\mathrm{d}^2u}{\mathrm{d}z^2} + \left[-c^2 + \frac{2}{z} - \frac{L(L+1)}{z^2}\right] u = 0,$$

which has a solution

$$u = e^{-cz}(2cz)^{L+1} v\left(L + 1 - \frac{1}{c}; 2L+2; 2cz\right), \tag{J.4}$$

$$\begin{aligned} v(q; n+1; z) = \frac{1}{n!\Gamma(q-n)}\{M(q; n+1; z)\log z + \sum_{j=0}^{\infty} \frac{(q)_j z^j}{(n+1)_j j!}\left[\psi(q+j)\right. \\ \left. -\psi(1+j) - \psi(1+j+n)\right]\} + \frac{(n-1)!}{\Gamma(q)}\frac{1}{z^n}\sum_{j=0}^{n-1}\frac{(q-n)_j}{(1-n)_j}\frac{z^j}{j!}. \end{aligned} \tag{J.5}$$

- In [13], the pricing of Asian options, a problem in financial mathematics, is considered, for all choices of market parameters. The Black-Scholes model

$$\mathrm{d}S_t = rS_t + \sigma S_t \mathrm{d}W_t, \quad t \geq 0,$$

is considered, where S_t denotes the price of the asset being considered at time t with initial price S_0, $r \geq 0$ denotes the constant risk-free rate, $\sigma > 0$ denotes constant volatility, and W_t represents standard Brownian motion. Of interest in this paper is the normalised price

$$C_A(\nu, \tau, \kappa) = \mathbb{E}[\max(A_\tau^\nu - \kappa, 0)],$$

where $A_\tau^\nu = \int_0^\tau e^{2W_2 + \nu s}\mathrm{d}s$ is Yor's process, which is related to the average price of the underlying asset over the time period $[0, \tau]$, and $\kappa > 0$ is the strike price of the Asian call option.

The Laplace transform of $C_A(\nu, \tau, \kappa)$,

$$\hat{C}_A \equiv \int_0^\infty C_A(\nu, \tau, \kappa)(t) e^{-st} \mathrm{d}t,$$

is sought; this is found to be related to the confluent hypergeometric function as follows:

$$U(s) = \frac{z^{b-a} e^{-s} \Gamma(a)}{s(s-2c)\Gamma(b)} M(a; b; z), \quad a = \frac{\mu+\nu}{2} + 2, \quad b = \mu + 1, \quad c = \nu + 1, \quad z = \frac{1}{2\kappa}, \tag{J.6}$$

for any μ.

- The Gauss hypergeometric function ${}_2F_1$ can be used to describe transonic adiabatic flow over a smooth bump, involving an ideal compressible fluid, as described in [16]. The model is described by the following equation:

$$\psi_{ww} + \frac{1}{w}\left(1 + \frac{w^2}{c^2}\right)\psi_w + \frac{1}{w^2}\left(1 - \frac{w^2}{c^2}\right)\psi_{\theta\theta} = 0,$$

where ψ is the stream function of the flow, w (the magnitude of the velocity) and θ (the angle between the flow and a reference direction) are the plane coordinates, and c is the speed of sound, which is given by

$$c^2 = \frac{1}{2\beta}(1 - w^2),$$

where $\beta = \frac{1}{\gamma - 1}$, and γ defines the ratio of the specific heat coefficients of fluid and air. One finds that

$$\psi(w, \theta) = \sum_{j=0}^{\infty} C_j w^j \; {}_2F_1(a_j, b_j; j+1; w^2) \sin(j\theta), \tag{J.7}$$

where

$$C_0 = 1, \quad C_j = \frac{2j-1}{2j} \times C_{j-1},$$
$$a_j + b_j = j - \beta, \quad b_j = \frac{1}{2}\beta j(j+1).$$

This needs to be extended for large j, which results in problems for computing the function ${}_2F_1(a, b; c; z)$ for large or small, real $|a|$, $|b|$ and $|c|$. This motivates the use of recurrence relations as detailed in Section 4.8.

- In [39], a plasma dispersion function $Z_\kappa(\xi)$ is considered. This is defined as

$$Z_\kappa(\xi) = \frac{1}{\pi^{1/2}\kappa^{3/2}} \frac{\Gamma(\kappa+1)}{\Gamma(\kappa-\frac{1}{2})} Q(\xi),$$

where

$$Q(\xi) = \int_{-\infty}^{\infty} \frac{\mathrm{d}s}{(s-\xi)(1+\frac{s^2}{\kappa})^{\kappa+1}}, \quad \mathrm{Im}(\xi) > 0.$$

with variable ξ and spectral index κ.

It is found that

$$Q(\xi) = \frac{2\pi i}{2^{2\kappa+2}} \frac{\Gamma(2\kappa+2)}{\Gamma(\kappa+1)\Gamma(\kappa+2)} \, {}_2F_1\left(1, 2\kappa+2; \kappa+2; \frac{1}{2}\left[1 - \frac{\xi}{i\sqrt{\kappa}}\right]\right), \tag{J.8}$$

and hence that

$$Z_\kappa(\xi) = \frac{i(\kappa+\frac{1}{2})(\kappa-\frac{1}{2})}{\kappa^{3/2}(\kappa+1)} \, {}_2F_1\left(1, 2\kappa+2; \kappa+2; \frac{1}{2}\left[1 - \frac{\xi}{i\sqrt{\kappa}}\right]\right). \tag{J.9}$$

It is noted in [39] that, when $\xi = 0$, the following relation can be used:

$${}_2F_1\left(a, b; \frac{1}{2}(a+b+1); \frac{1}{2}\right) = \frac{\Gamma\left(\frac{1}{2}\right)\Gamma\left(\frac{1}{2}+\frac{1}{2}a+\frac{1}{2}b\right)}{\Gamma\left(\frac{1}{2}+\frac{1}{2}a\right)\Gamma\left(\frac{1}{2}+\frac{1}{2}b\right)}.$$

- In [21], the effect of penetration by electrons of a potential barrier is investigated. The wave equation related to this problem is

$$\xi^2 u'' + \xi u' + \frac{2ml^2}{h^2}\left[\frac{A\xi}{1-\xi} + \frac{B\xi}{(1-\xi)^2} + W\right] u = 0,$$

where u denotes displacement, h is Planck's constant, the de Broglie wavelength of the electron is $2l$, A, B, and W are other constants, ξ is the transformed length coordinate, and m denotes mass.

The solution is

$$\begin{aligned} u = {} & a_1 \left(\frac{\xi}{\xi-1}\right)^{i\alpha} (1-\xi)^{i\beta} \, {}_2F_1\left(\frac{1}{2} + i(\alpha-\beta+\delta), -\frac{1}{2} + i(\alpha-\beta-\delta); 1+2i\alpha; \frac{\xi}{\xi-1}\right) \\ & + a_2 \left(\frac{\xi}{\xi-1}\right)^{-i\alpha} (1-\xi)^{i\beta} \\ & \times {}_2F_1\left(\frac{1}{2} + i(-\alpha-\beta+\delta), -\frac{1}{2} + i(-\alpha-\beta-\delta); 1-2i\alpha; \frac{\xi}{\xi-1}\right), \end{aligned} \tag{J.10}$$

which converges when $\left|\frac{\xi}{\xi-1}\right| < 1$, where

$$\begin{aligned}
a_1 &= \frac{\Gamma(1-2i\beta)\Gamma(-2i\alpha)}{\Gamma\left(\frac{1}{2}+i(-\alpha-\beta-\delta)\right)\Gamma\left(\frac{1}{2}+i(-\alpha-\beta+\delta)\right)}, \\
a_2 &= \frac{\Gamma(1-2i\beta)\Gamma(2i\alpha)}{\Gamma\left(\frac{1}{2}+i(\alpha-\beta-\delta)\right)\Gamma\left(\frac{1}{2}+i(\alpha-\beta+\delta)\right)}, \\
\alpha &= \frac{1}{2}\left(\frac{W}{C}\right)^{\frac{1}{2}}, \\
\beta &= \frac{1}{2}\left(\frac{W-A}{C}\right)^{\frac{1}{2}}, \\
\delta &= \frac{1}{2}\left(\frac{B-C}{C}\right)^{\frac{1}{2}}, \\
C &= \frac{h^2}{8ml^2},
\end{aligned}$$

where the electron has energy C.

- In [54], the susceptible-infected-susceptible (SIS) epidemic model is applied to complex networks. The network studied is represented by an exponential network, meaning that the probability that a node is connected to k others, $P(k)$, is exponentially bounded. The network obeys the law

$$P(k) \sim k^{2-\gamma},$$

where the parameter $\gamma > 0$.

In this paper, the density of infected nodes, ρ, is found to satisfy

$$\rho = {}_2F_1(1, 1+\gamma; 2+\gamma; -[m\lambda\Theta(\lambda)]^{-1}), \tag{J.11}$$

where m is the minimum number of connections at each node, λ denotes the spreading rate, and $\Theta(\lambda)$ denotes the probability that a link points to an infected node with this spreading rate.

K List of code written for this project

Below is a list of routines written for this project. All the code listed carries out computations for complex parameters and variable unless otherwise specified. These routines will be hosted on the website

`http://people.maths.ox.ac.uk/~porterm/research/hypergeometricpackage.zip`

by 12 September 2009.

Code for computing ${}_1F_1(a;b;z)$:

- `taylora1f1.m` – Computes ${}_1F_1(a;b;z)$ using Taylor series method (a) in Section 3.2
- `taylorb1f1.m` – Computes ${}_1F_1(a;b;z)$ using Taylor series method (b) in Section 3.2
- `singlefraction1f1.m` – Computes ${}_1F_1(a;b;z)$ using the single fraction method in Section 3.3
- `buchholza1f1.m` – Computes ${}_1F_1(a;b;z)$ using method 1 in Section 3.4 for complex a, b
- `buchholzbreal1f1.m` – Computes ${}_1F_1(a;b;z)$ using method 2 in Section 3.4 for real a, b
- `buchholzbcomplex1f1.m` – Computes ${}_1F_1(a;b;z)$ using method 2 in Section 3.4 for complex a, b
- `buchholzc1f1.m` – Computes ${}_1F_1(a;b;z)$ using method 3 in Section 3.4 for complex a, b
- `buchholzpoly.m` – Computes the Buchholz polynomials $p_n(b,z)$ for use in the routines `buchholzb1f1.m` and `buchholzc1f1.m`
- `betaseries1f1.m` – Computes ${}_1F_1(a;b;z)$ using the series of beta random variables, as described in Appendix G.1
- `asymptotica1f1.m` – Computes ${}_1F_1(a;b;z)$ using method (a) for asymptotic series in Section 3.5
- `asymptoticb1f1.m` – Computes ${}_1F_1(a;b;z)$ using method (b) for asymptotic series in Section 3.5
- `hyperasymptoticu1f1.m` – Model code for computing $U(a;b;z)$ using the hyperasymptotic expansion given in Appendix G.4

- `gjquadreal1f1.m` – Computes ${}_1F_1(a;b;z)$ using Gauss-Jacobi quadrature as in Section 3.6 for real a, b
- `gjquadcomplex1f1.m` – Computes ${}_1F_1(a;b;z)$ using Gauss-Jacobi quadrature as in Section 3.6 for complex a, b
- `deivprk1f1.m` – Computes ${}_1F_1(a;b;z)$ using the RK4 method as discussed in Section 3.7
- `recurrencea1f1.m` – Computes $U(a+n;b;z)$ using the computation of $U(a;b;z)$, as discussed in Section 3.8
- `recurrenceb1f1.m` – Computes ${}_1F_1(a;b+n;z)$ using the computation of ${}_1F_1(a;b;z)$, as discussed in Section 3.8
- `recurrencec1f1.m` – Computes ${}_1F_1(a+n;b+n;z)$ using the computation of ${}_1F_1(a;b;z)$, as discussed in Section 3.8
- `incgammaexpansion1f1.m` – Computes ${}_1F_1(a;b;z)$ using the incomplete gamma function expansion given in Appendix G.2
- `asymptoticexpansionbz1f1.m` – Computes ${}_1F_1(a;b;z)$ using the asymptotic expansion for large $|b|$, $|z|$ given in Appendix G.3
- `intsplit1f1.m` – Computes ${}_1F_1(a;b;z)$ using the method of splitting the integral in (3.25), as discussed in Appendix G.5
- `romberg1f1.m` – Computes ${}_1F_1(a;b;z)$ using Romberg integration, as discussed in Appendix G.5
- `adquad1f1.m` – Computes ${}_1F_1(a;b;z)$ using adaptive quadrature, as discussed in Appendix G.5, with the trapezoidal rule and Simpson's rule
- `adquadboole1f1.m` – Computes ${}_1F_1(a;b;z)$ using adaptive quadrature, as discussed in Appendix G.5, with the trapezoidal rule and Boole's rule
- `deivpdp1f1.m` – Computes ${}_1F_1(a;b;z)$ using the Dormand-Prince method, as discussed in Appendix G.6
- `pade1f1.m` – Computes ${}_1F_1(a;b;z)$ using Padé approximants, as discussed in Appendix G.7
- `rational1f1.m` – Computes ${}_1F_1(a;b;z)$ using rational approximation, as discussed in Appendix G.7
- `oscint1f1.m` – Computes ${}_1F_1(a;b;z)$ using the method for oscillatory integrals, as discussed in Appendix G.5

- `debvpfd1f1.m` – Computes the profile of ${}_1F_1(a;b;z)$ over a range of z using the finite difference method, as discussed in [73]
- `debvpshooting1f1.m` – Computes the profile of ${}_1F_1(a;b;z)$ over a range of z using the shooting method, as discussed in [73]
- `debvpchebyshev1f1.m` – Computes the profile of ${}_1F_1(a;b;z)$ over a range of z using Chebyshev differentiation matrices, as discussed in [73]

Code for computing ${}_2F_1(a,b;c;z)$:

- `taylora2f1.m` – Computes ${}_2F_1(a,b;c;z)$ using Taylor series method (a) in Section 4.2
- `taylorb2f1.m` – Computes ${}_2F_1(a,b;c;z)$ using Taylor series method (b) in Section 4.2
- `singlefraction2f1.m` – Computes ${}_2F_1(a,b;c;z)$ using the single fraction method in Section 4.3
- `transformations2f1.m` – Code containing transformations detailed in Section 4.6
- `buhringa2f1.m` – Computes ${}_2F_1(a,b;c;z)$ using the method of Section 4.7
- `buhringb2f1.m` – Computes ${}_2F_1(a,b;c;z)$ using the method of Section 4.7 with $a = b$
- `gjquadreal2f1.m` – Computes ${}_2F_1(a,b;c;z)$ using Gauss-Jacobi quadrature as in Section 4.4 for real a, b, c
- `gjquadcomplex2f1.m` – Computes ${}_2F_1(a,b;c;z)$ using Gauss-Jacobi quadrature as in Section 4.4 for complex a, b, c
- `deivprk2f1.m` – Computes ${}_2F_1(a,b;c;z)$ using the RK4 method as discussed in Section 4.5
- `recurrencea2f1.m` – Code that implements recurrence relation 1 of Section 4.8
- `recurrenceb2f1.m` – Code that implements recurrence relation 2 of Section 4.8
- `recurrencec2f1.m` – Code that implements recurrence relation 3 of Section 4.8
- `recurrenced2f1.m` – Code that implements recurrence relation 4 of Section 4.8
- `intsplit2f1.m` – Computes ${}_2F_1(a,b;c;z)$ using the method of splitting the integral in (4.8), as discussed in Appendix H.1
- `romberg2f1.m` – Computes ${}_2F_1(a,b;c;z)$ using Romberg integration, as discussed in Appendix H.1

- `adquad2f1.m` – Computes ${}_2F_1(a,b;c;z)$ using adaptive quadrature, as discussed in Appendix H.1, with the trapezoidal rule and Simpson's rule
- `adquadboole2f1.m` – Computes ${}_2F_1(a,b;c;z)$ using adaptive quadrature, as discussed in Appendix H.1, with the trapezoidal rule and Boole's rule
- `deivpdp2f1.m` – Computes ${}_2F_1(a,b;c;z)$ using the Dormand-Prince method, as discussed in Appendix H.2
- `pade2f1.m` – Computes ${}_2F_1(a,b;c;z)$ using Padé approximants, as discussed in Appendix H.3
- `debvpfd2f1.m` – Computes the profile of ${}_2F_1(a,b;c;z)$ over a range of z using the finite difference method, as discussed in [73]
- `debvpshooting2f1.m` – Computes the profile of ${}_2F_1(a,b;c;z)$ over a range of z using the shooting method, as discussed in [73]
- `debvpchebyshev2f1.m` – Computes the profile of ${}_2F_1(a,b;c;z)$ over a range of z using Chebyshev differentiation matrices, as discussed in [73]

Code for computing other hypergeometric functions:

- `taylora0f1.m` – Computes ${}_0F_1(\ ;a;z)$ using Taylor series method (a), as detailed in Appendix I
- `taylorb0f1.m` – Computes ${}_0F_1(\ ;a;z)$ using Taylor series method (b), as detailed in Appendix I
- `bessel0f1.m` – Computes ${}_0F_1(\ ;a;z)$ using its Bessel function representation, as detailed in Appendix I
- `taylor2f2.m` – Computes ${}_2F_2(a,b;c,d;z)$ using a Taylor series method
- `taylor3f3.m` – Computes ${}_3F_3(a,b,c;d,e,f;z)$ using a Taylor series method

References

[1] J. Abad, J. Sesma: *Buchholz Polynomials: A Family of Polynomials Relating Solutions of Confluent Hypergeometric and Bessel Equations*, Journal of Computational and Applied Mathematics, **101**, pp.237–41, 1999

[2] J. Abad, J. Sesma: *Computation of the Regular Confluent Hypergeometric Function*, The Mathematica Journal, **5**, pp.74–6, 1995

[3] M. Abramowitz, I. Stegun (Editors): *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, National Bureau of Standards, 1970

[4] G. Allasia, R. Besenghi: *Numerical Computation of Tricomi's Psi Function by the Trapezoidal Rule*, Computing, **39**, pp.271–9, 1987

[5] L. U. Ancarani, G. Gasaneo: *Derivatives of any Order of the Confluent Hypergeometric Function ${}_1F_1(a,b,z)$ with Respect to the Parameter a or b*, Journal of Mathematical Physics, **49**, DOI: 10.1063/1.2939395, 2008

[6] G. E. Andrews, R. Askey, R. Roy: *Special Functions*, Volume 71 of *Mathematics and its Applications*, Cambridge University Press, 1999

[7] R. Badralexe, P. Marksteiner, Y. Oh, A. J. Freeman: *Computation of the Kummer functions and Whittaker functions by using Neumann Type Series Expansions*, Computer Physics Communications, **71**, pp.47–55, 1992

[8] W. N. Bailey: *Generalized Hypergeometric Series*, Cambridge Tract No.32, Cambridge University Press, 1935

[9] W. Becken, P. Schmelcher: *The Analytic Continuation of the Gaussian Hypergeometric Function ${}_2F_1(a,b;c;z)$ for Arbitrary Parameters*, Journal of Computational and Applied Mathematics, **126**, pp.449–78, 2000

[10] K. L. Bell, N. S. Scott: *Coulomb Functions (Negative Energies)*, Computer Physics Communications, **20**, pp.447–58, 1980

[11] M. Berry: *Asymptotics, Superasymptotics, Hyperasymptotics...*, in *Asymptotics Beyond All Orders*, ed. H. Segur, S. Tanveer (Plenum, New York, 1991), pp.1–14

[12] H. Bornemann, D. Laurie, S. Wagon, J. Waldvogel: *The SIAM 100-Digit Challenge: A Study in High-Accuracy Numerical Computing*, SIAM, 2004

[13] P. Boyle, A. Potapchik: *Application of High-Precision Computing for Pricing Arithmetic Asian Options*, International Conference on Symbolic and Algebraic Computation, pp.39–46, 2006

[14] W. Bühring: *An Analytic Continuation of the Hypergeometric Series*, SIAM Journal of Mathematical Analysis, **18**, pp.884–9, 1987

[15] J. L. Cardoso, C. M. Fernandes, R. Álvarez-Nodarse: *Structural and Recurrence Relations for Hypergeometric-Type Functions by Nikiforov-Uvarov Method*, Electronic Transactions on Numerical Analysis, **35**, pp.17–39, 2009

[16] G. Chiocchia, B. Gabutti: *A New Transformation for Computing Hypergeometric Series and the Exact Evaluation of the Transonic Adiabatic Flow Over a Smooth Bump*, Computers and Fluids, **17**, pp.13–23, 1989

[17] A. Deaño, J. Segura: *Transitory Minimal Solutions of Hypergeometric Recursions and Pseudoconvergence of Associated Continued Fractions*, Mathematics of Computation, **76**, pp.879–901, 2007

[18] A. Deaño, N. M. Temme: *On Modified Asymptotic Series Involving Confluent Hypergeometric Functions*, University of Cambridge Department of Applied Mathematics and Theoretical Physics Technical Report NA2008/07, to appear in Electronic Transactions on Numerical Analysis, 2008

[19] J. R. Dormand, P. J. Prince: *A Family of Embedded Runge-Kutta Formulae*, Journal of Computational and Applied Mathematics, **6**, 1980

[20] T. M. Dunster: *Uniform Asymptotic Expansions for Whittaker's Confluent Hypergeometric Function*, SIAM Journal of Mathematical Analysis, **20**, pp.744–60, 1989

[21] C. Eckart: *The Penetration of a Potential Barrier by Electrons*, Physical Review, **35**, pp.1303–09, 1930

[22] A. Erdélyi, W. Magnus, F. Oberhettinger, F. G. Tricomi: *Higher Transcendental Functions, Volume I*, McGraw-Hill, 1953

[23] C. Ferreira, J. L. López, E. P. Sinusía: *The Gauss Hypergeometric Function $F(a, b; c; z)$ for Large c*, Journal of Computational and Applied Mathematics, **197**, pp.568–77, 2006

[24] R. C. Forrey: *Computing the Hypergeometric Function*, Journal of Computational Physics, **137**, pp.79–100, 1997

[25] W. Gautschi: *Computational Aspects of Three-Term Recurrence Relations*, SIAM Review, **9**, pp.24–82, 1967

[26] W. Gautschi: *Gauss Quadrature Approximations to Hypergeometric and Confluent Hypergeometric Functions*, Journal of Computational and Applied Mathematics, **139**, pp.173–87, 2002

[27] M. Gavrila: *Elastic Scattering of Photons by a Hydrogen Atom*, Physical Review, **163**, pp.147–55, 1967

[28] A. Gil, J. Segura, N. M. Temme: *The ABC of Hyper Recursions*, Journal of Computational and Applied Mathematics, **190**, pp.270–86, 2006

[29] A. Gil, J. Segura, N. M. Temme: *Numerically Satisfactory Solutions of Hypergeometric Recursions*, Mathematics of Computation, **76**, pp.1449–68, 2007

[30] A. Gil, J. Segura, N. M. Temme: *Numerical Methods for Special Functions*, SIAM, 2007

[31] H. Hochstadt: *The Functions of Mathematical Physics*, John Wiley & Sons, 1971

[32] A. K. Ibrahim, M. A. Rakha: *Contiguous Relations and Their Computations for ${}_2F_1$ Hypergeometric Series*, Computers and Mathematics with Applications, **56**, pp.1918–26, 2008

[33] P. Koev, A. Edelman: *The Efficient Evaluation of the Hypergeometric Function of a Matrix Argument*, Mathematics of Computation, **75**, pp.833–46, 2006

[34] J. L. López, P. J. Pagola: *The Confluent Hypergeometric Functions $M(a, b; z)$ and $U(a, b; z)$ for Large b and z*, Journal of Computational and Applied Mathematics, DOI: 10.1016/j.cam.2009.02.072, 2009

[35] Y. L. Luke: *Algorithms for the Computation of Mathematical Functions*, Academic Press, 1977

[36] Y. L. Luke: *Mathematical Functions and their Approximations*, Academic Press, 1975

[37] Y. L. Luke: *The Special Functions and Their Approximations, Volume I*, Academic Press, 1969

[38] Y. L. Luke: *The Special Functions and Their Approximations, Volume II*, Academic Press, 1969

[39] R. L. Mace, M. A. Hellberg: *A Dispersion Function for Plasmas Containing Superthermal Particles*, Physics of Plasmas, **2**, pp.2098–2109, 1995

[40] R. J. Mathar: *Numerical Representations of the Incomplete Gamma Function of Complex-Valued Argument*, Numerical Algorithms, **36**, pp.247–64, 2004

[41] N. Michel, M. V. Stoitsov: *Fast Computation of the Gauss Hypergeometric Function with all its Parameters Complex with Application to the Pöschl-Teller-Ginocchio Potential Wave Functions*, Computer Physics Communications, **178**, pp.535–51, 2008

[42] T. Morita: *Use of the Gauss Contiguous Relations in Computing the Hypergeometric Functions* $F(n+1/2, n+1/2; m; z)$, Interdisciplinary Information Sciences, **2**, pp.63–74, 1996

[43] S. L. Moshier: *Methods and Programs for Mathematical Functions*, Ellis Horwood, 1989

[44] K. E. Muller: *Computing the Confluent Hypergeometric Function,* $M(a, b, x)$, Numerische Mathematik, **90**, pp.179–96, 2001

[45] M. Nardin, W. F. Perger, A. Bhalla: *Algorithm 707: CONHYP: A Numerical Evaluator of the Confluent Hypergeometric Function for Complex Arguments of Large Magnitudes*, ACM Transactions on Mathematical Software, **18**, pp.345–9, 1992

[46] M. Nardin, W. F. Perger, A. Bhalla: *Numerical Evaluation of the Confluent Hypergeometric Function for Complex Arguments of Large Magnitudes*, Journal of Computational and Applied Mathematics, **39**, pp.193–200, 1992

[47] C. J. Noble, I. J. Thompson: *COULN, a Program for Evaluating Negative Energy Coulomb Functions*, Computer Physics Communications, **33**, pp.413–9, 1984

[48] A. B. Olde Daalhuis: *Hyperasymptotic Expansions of Confluent Hypergeometric Functions*, IMA Journal of Applied Mathematics, **49**, pp.203–16, 1992

[49] A. B. Olde Daalhuis, F. W. J. Olver: *Hyperasymptotic Solutions of Second-Order Linear Differential Equations I*, Methods and Applications of Analysis, **2**, pp.173–97, 1995

[50] A. B. Olde Daalhuis, F. W. J. Olver: *On the Asymptotic and Numerical Solution of Linear Ordinary Differential Equations*, SIAM Review, **40**, pp.463–95, 1998

[51] F. W, Olver: *Asymptotics and Special Functions*, Academic Press, 1974

[52] F. W. Olver: *Exponentially-Improved Asymptotic Solutions of Ordinary Differential Equations, I: The Confluent Hypergeometric Function*, SIAM Journal of Mathematical Analysis, **24**, pp.756–67, 1993

[53] F. W. Olver: *Uniform, Exponentially Improved, Asymptotic Expansions for the Confluent Hypergeometric Function and Other Integral Transforms*, SIAM Journal of Mathematical Analysis, **22**, pp.1475–89, 1991

[54] R. Pastor-Satorras, A. Vespignani: *Epidemic Dynamics and Endemic States in Complex Networks*, Physical Review E, **63**, DOI: 10.1103, 066117, 2001

[55] T. N. L. Patterson: *On High Precision Methods for the Evaluation of Fourier Integrals with Finite and Infinite Limits*, Numerische Mathematik, **27**, pp.41–52, 1976

[56] V. Pierro, I. M. Pinto, A. D. A. M. Spallicci di F. : *Computation of Hypergeometric Functions for Gravitationally Radiating Binary Stars*, Monthly Notices of the Royal Astronomical Society, **334**, pp.855–8, 2002

[57] W. A. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery: *Numerical Recipes: The Art of Scientific Computing*, Third Edition, Cambridge University Press, 2007

[58] M. A. Rakha, E. S. El-Sedy: *Application of Basic Hypergeometric Series*, Applied Mathematics and Computation, **148**, pp.717–23, 2004

[59] H. T. Rathod, B. Venkatesudu, K. V. Nagaraja, Md. Shafiqul Islam: *Gauss Legendre–Gauss Jacobi Quadrature Rules Over a Tetrahedral Region*, Applied Mathematics and Computation, **190**, pp.186–94, 2007

[60] K. Roach: *Hypergeometric Function Representations*, Proceedings of the 1996 International Symposium on Symbolic and Algebraic Computation, pp.301–8, 1996

[61] J. B. Seaborn: *Hypergeometric Functions and their Applications*, Springer-Verlag, 1991

[62] J. Segura, N. M. Temme: *Numerically Satisfactory Solutions of Kummer Recurrence Relations*, Numerische Mathematik, **111**, pp.109–19, 2008

[63] J. Sesma: *The Temme's Sum Rule for Confluent Hypergeometric Functions Revisited*, Journal of Computational and Applied Mathematics, **163**, pp.429–31, 2004

[64] L. J. Slater: *Confluent Hypergeometric Functions*, Cambridge University Press, 1960

[65] L. J. Slater: *Generalized Hypergeometric Functions*, Cambridge University Press, 1966

[66] N. M. Temme: *Large Parameter Cases of the Gauss Hypergeometric Function*, Journal of Computational and Applied Mathematics, **2003**, pp.441–62, 2003

[67] N. M. Temme: *The Numerical Computation of the Confluent Hypergeometric Function* $U(a, b, z)$, Numerisch Mathematik, **41**, pp.43–82, 1983

[68] N. M. Temme: *Uniform Asymptotic Expansions of Confluent Hypergeometric Functions*, Journal of the Institute of Mathematical Applications, **22**, pp.215–23, 1978

[69] J. Wimp: *Computation with Recurrence Relations*, Pitman, 1984

[70] S. Zhang, J. Jin: *Computation of Special Functions*, John Wiley & Sons, 1966

[71] `http://www.mathworks.com/matlabcentral/fileexchange/6218`, file `cgama.m` from package 'Computation of Special Functions', by B. Barrowes, last updated 14 Apr 2005, downloaded 26 Apr 2009

[72] `http://www.mathworks.com/matlabcentral/fileexchange/32`, files `gaussq.m` and `qrule.m`, by P. A. Brodtkorb, last updated 28 Nov 2005, downloaded 4 May 2009

[73] J. Pearson: *An Analysis of Five Numerical Methods for Approximating Certain Hypergeometric Functions in Domains Within Their Radii of Convergence*, MSc in Mathematical Modelling and Scientific Computing Special Topic, 2009, hosted at `http://people.maths.ox.ac.uk/~porterm/research/hypergeometricsptop.pdf`