

Fast and accurate evaluation of a generalized incomplete gamma function

Rémy Abergel, Lionel Moisan

▶ **To cite this version:**

Rémy Abergel, Lionel Moisan. Fast and accurate evaluation of a generalized incomplete gamma function. 2016. hal-01329669v1

HAL Id: hal-01329669

https://hal.science/hal-01329669v1

Preprint submitted on 9 Jun 2016 (v1), last revised 7 Nov 2019 (v2)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Fast and accurate evaluation of a generalized incomplete gamma function

Rémy Abergel and Lionel Moisan,
Université Paris Descartes, MAP5 (CNRS UMR 8145), France.

June 7, 2016

Abstract

We propose a computational procedure to evaluate the generalized incomplete gamma function $\int_x^y s^{p-1}\,e^{-\mu s}\,ds$ for $0 \leq x < y \leq +\infty$, a real number $\mu \neq 0$ and a positive integer p. Our approach consists in selecting, according to the value of the parameters x, y, μ, p, the fastest and most accurate estimate among series expansions, continued fractions, recursive integration by parts, or, when $x \approx y$, a first order trapezoidal rule. We show that the accuracy reached by our algorithm is nearly optimal for a large range of parameters.

Keywords: differences of generalized incomplete gamma function, special functions, integral calculus.

1 Introduction

In this work, we focus on the computation of a generalized incomplete gamma function that will be defined below. Let us first recall the definition of the gamma function,

$$\forall a > 0, \quad \Gamma(a) = \int_0^{+\infty} s^{a-1}\,e^{-s}\,ds\,. \tag{1}$$

The lower and upper incomplete gamma functions are respectively obtained by allowing the integration domain to vary in (1),

$$\forall a > 0, \quad \forall x \geq 0, \quad \gamma(a,x) = \int_0^{x} s^{a-1}\,e^{-s}\,ds \quad \text{and} \quad \Gamma(a,x) = \int_x^{+\infty} s^{a-1}\,e^{-s}\,ds\,. \tag{2}$$

The gamma function is usually viewed as an extension of the factorial function since it satisfies $\Gamma(a) = (a-1)!$ for any positive integer a. Note that the gamma function can also be defined for all complex numbers a with positive real part, using the same convergent improper integral as in (1), and can even be extended by analytic continuation to all complex numbers except the nonpositive integers, that is, to $a \in \mathbb{C} \setminus \{0, -1, -2, -3, \dots\}$. Some similar extensions are also available for their incomplete variants $\gamma_\mu(a,x)$ and $\Gamma_\mu(a,x)$. These special functions arise in many areas, such as astronomy and astrophysics [Cannon and Vardavas, 1974, Hills, 1975], Rayleigh scattering [Kissel et al., 1980], quantum gravity [Bleicher and Nicolini, 2010], networks [Moreno et al., 2002], financial mathematics [Linetsky, 2006], image analysis [Robin et al., 2010], etc. (see [Chaudhry and Zubair, 2001] for more examples). From the mathematical viewpoint, the computation of incomplete gamma functions

is typically required in applications involving the evaluation of χ^2 distribution functions, exponential integrals, error functions (erf), cumulative Poisson or Erlang distributions, etc. Their practical numerical evaluation is still subject to some flourishing research in the modern litterature. The first practical algorithm dedicated to the numerical evaluation of the incomplete gamma functions was, to the best of our knowledge, proposed in [Bhattacharjee, 1970]. It consists in evaluating the ratio $\gamma(a,x)/\Gamma(a)$ using a series expansion when $0 < a \leq x < 1$ or $0 \leq x < a$, or the ratio $\Gamma(a,x)/\Gamma(a)$ using a continued fraction in the remaining part of the domain $\{x \geq 0, a > 0\}$. The same strategy is also used in [Press et al., 1992]. Gautschi [1979] proposed another computational procedure, based on Taylor's series and continued fractions, to evaluate those two functions in the region $\{x \geq 0, a \in \mathbb{R}\}$ (in fact, for $a \leq 0$, Tricomi's version [Tricomi, 1950, Gautschi, 1998] of the lower incomplete gamma function, which remains real for any real numbers x, a, is considered). The criterion proposed in [Bhattacharjee, 1970] to decide which one of the two integrals should be computed according to the value of (x,a) is refined, and a more suitable normalization is employed, which extends the range over which those two functions can be represented within standard double precision arithmetics. More recently, Winitzki [2003] focused on the computation of the upper incomplete gamma function and used some series expansions, a continued fraction (due to Legendre), some recurrence relations, or, for large values of x, an asymptotic series. The precision of the approximation is controlled by estimating the number of terms required to reach a given absolute precision according to the values of x and a. However, the study is not considered from the practical point of view, and no algorithm or experimental validation are provided to assess the numerical stability of the proposed method. In [Guseinov and Mamedov, 2004], the lower and upper incomplete gamma functions are computed using backward and forward recurrence relations. The experimental validation is done for the range $0.001 \leq x \leq 100$ and $0 < a \leq 100$, which is relatively large in comparison to the numerical validations usually proposed in the litterature. We will also use, for a particular region of the quarter plane $\{(a,x),\ a > 0,\ x < 0\}$, a recurrence relation to compute the lower incomplete gamma function. However, we shall see that in the region $x > 0$, we experimentally achieve a faster convergence by using some continued fractions.

In the present paper, we consider the more general case of the generalized incomplete gamma function, defined by

$$I_{x,y}^{\mu,p} = \int_x^y s^{p-1}\, e^{-\mu s}\, ds, \quad \text{for} \quad 0 \leq x < y \leq +\infty,\ p > 0,\ \mu \in \mathbb{R} \setminus \{0\}\,, \tag{3}$$

and we restrict the study to integer values of p (even though all the algorithms we propose also work for non-integer values of p when $\mu x > 0$). Notice that $y = +\infty$ is only allowed when $\mu > 0$, otherwise the integral is equal to $+\infty$. Note also that thanks to the rescaling relation

$$I_{x,y}^{\mu,p} = |\mu|^{-p}\, I_{|\mu|x,|\mu|y}^{\varepsilon,p}, \quad \text{where} \quad \varepsilon = \frac{\mu}{|\mu|}\,, \tag{4}$$

we could restrict the study, without any loss of generality, to $\mu \in \{-1, 1\}$. We will however not adopt this restriction since it does not simplify the study, even though the numerical evaluations that we propose are limited to $\mu = \pm 1$, which simplifies the experimental validation. The computation of $I_{x,y}^{\mu,p}$ will be closely related to that of the generalized lower (γ_μ) and upper (Γ_μ) incomplete gamma functions, which we naturally define by

$$\forall \mu \in \mathbb{R}, \quad \gamma_\mu(p,x) = \int_0^x s^{p-1}\, e^{-\mu s}\, ds\,, \quad \forall \mu > 0, \quad \Gamma_\mu(p,x) = \int_x^{+\infty} s^{p-1}\, e^{-\mu s}\, ds\,. \tag{5}$$

Note that when $\mu > 0$ in (3) or (5), the change of variable $t = \mu s$ would lead us back to the standard definitions of the incomplete gamma functions (up to the multiplicative factor

μ^{-p}), but this is not the case when $\mu < 0$. The possibility to evaluate the lower incomplete gamma function $\gamma(p,x)$ with a negative argument x (which amounts to compute $\gamma_\mu(p,|x|)$ with $\mu=-1$) is explored in [Thompson, 2013], but in another situation than ours, since he focused on the case $p = n + \frac{1}{2},\ n \in \mathbb{Z}$.

The generalized incomplete gamma function (5) was actually previously introduced in [Fullerton, 1972], under the slightly different form

$$J^a_{x_1,x_2} = e^{x_1} \int_{x_1}^{x_2} |s|^{a-1}\, e^{-s}\, ds\,, \quad \text{for any } (x_1,x_2) \in \mathbb{R}^2, \text{ and } a > 0\,. \tag{6}$$

The integrals I and J are closely related since one easily checks that

$$\forall x, y,\ 0 < x < y, \quad \forall p > 0, \quad I^{\mu,p}_{x,y} = \begin{cases} \mu^{-p}\, e^{-\mu x} J^p_{\mu x,\mu y} & \text{if } \mu > 0\,, \\ |\mu|^{-p}\, e^{-\mu y} J^p_{\mu y,\mu x} & \text{if } \mu < 0\,. \end{cases} \tag{7}$$

Our parametrization of the integral I using the scale parameter μ will be helpful to avoid the absolute values in the integral, which would inevitably have involved the distinction of the cases $x_1 > x_2$ and $x_1 \leq x_2$ in our study. The numerical evaluation of the generalized incomplete gamma function $I^{\mu,p}_{x,y}$ has found some applications in the field of astronomy, for instance in [Hills, 1975], where its computation was needed to model the dynamical evolution of stellar clusters. It was more recently needed in the field of image processing, in [Abergel et al., 2015], where the accurate computation of $I^{\mu,p}_{x,y}$ for a large range of parameters was at the heart of a denoising algorithm for the restoration of images corrupted with Poisson noise. Unfortunately, Fullerton's algorithm, which was not validated for a large range of parameters, presents several weaknesses. As pointed out in [Schoene, 1978], for some values of the parameters, the algorithm suffers from numerical instabilities, yielding for instance, a computed integral with incorrect sign, or zero digit of precision. We also observed some overflow issues when we tested the algorithm on a higher range of parameters (typically when $p \approx 10^2$ or higher, but also for many other parameter settings).

Note also that a numerical procedure specific to the evaluation of $I^{\mu,p}_{x,y}$ is available into the scientific computing software *Mathematica* (see [Wolfram Research Inc, 1988], but also [Wolfram Research Inc, 1998] for the online evaluation of $I^{\mu,p}_{x,y}$). Unfortunately, *Mathematica*'s algorithms are not currently disclosed to the public.

Let us now consider the numerical evaluation of $I^{\mu,p}_{x,y}$. This integral can be computed as a difference of generalized lower (γ_μ) and upper (Γ_μ) incomplete gamma functions, since for any $\mu \in \mathbb{R}$, we have

$$I^{\mu,p}_{x,y} = \gamma_\mu(p,y) - \gamma_\mu(p,x)\,, \tag{8}$$

and for $\mu > 0$, we have

$$I^{\mu,p}_{x,y} = \Gamma_\mu(p,x) - \Gamma_\mu(p,y) = \frac{\Gamma(p)}{\mu^p} - \gamma_\mu(p,x) - \Gamma_\mu(p,y)\,. \tag{9}$$

The effective computation of $I^{\mu,p}_{x,y}$ using (8) or (9) raises several numerical issues:

1. For some values of the parameters, the generalized incomplete gamma functions γ_μ and Γ_μ cannot be represented in the computer floating point arithmetic (for example when they exceed $1.9 \cdot 10^{308}$, the largest double precision number). To solve that issue, we will represent all integrals in (5) under the form $\rho \cdot e^\sigma$, where ρ and σ are floating point numbers with double precision;

2. The possibility to efficiently compute $\gamma_\mu(p,x)$ and $\Gamma_\mu(p,x)$ depends on the values of the parameters μ, p, x, or more precisely of μx and p because of the scaling relation (4). We derived a division of the plane $(\mu x, p)$ allowing an efficient computation of these two functions for each parameter set (μ, p, x);

3. When $I_{x,y}^{\mu,p}$ is computed as the difference $A-B$, the result may be inaccurate if A and B are close to each other (the well-known *cancellation* effect in floating-point arithmetic), which typically happens in (8)-(9) when x and y are very close to each other. In that case, the integral $I_{x,y}^{\mu,p}$ is well approximated by a first order approximation of the integral. We found a good criterion, to decide when this approximation should be used.

In particular, the issue (1) detailed above is of great importance when some integrals of the kind $I_{x,y}^{\mu,p}$ appear into more complicated mathematical expressions, such as in [Abergel et al., 2015], where the computation of a ratio of sums of generalized incomplete gamma functions is involved, with a numerator and a denominator that may both exceed the highest representable double floating point number, although the ratio itself is representable in the standard computer floating-point arithmetic.

This paper is organized as follows. In Section 2, we recall some mathematical methods based on series expansion, fraction continuation, or recursive integration by parts, that can be used for the numerical evaluation with a mantissa-exponent representation of the generalized lower (Section 2.1) and upper (Section 2.2) incomplete gamma functions γ_μ and Γ_μ. In Section 2.3, we derive theoritical accuracy bounds achievable with such a mantissa-exponent representation, and check experimentally in Section 2.4 that we can achieve these bounds by selecting the appropriate method (continued fraction or integration by parts) depending on the values of the parameters μ, p, x. In Section 3, we focus on the practical evaluation of the generalized incomplete gamma function $I_{x,y}^{\mu,p}$ (that is, with arbitrary finite values of x and y). The numerical evaluation of this integral is done by means of a difference (8)-(9), or, when $x \approx y$, using the first order trapezoidal rule. In Section 5, our algorithm is compared with the one of Fullerton, and is shown to exhibit a much greater accuracy for a large range of parameters. We finally conclude in Section 6 and discuss some perspectives.

2 Numerical computation of the generalized lower and upper incomplete gamma functions

2.1 Evaluation of the generalized lower incomplete gamma function

Given $p \geq 1$, $x > 0$ and $\mu \neq 0$, we detail below how the generalized lower incomplete gamma $\gamma_\mu(p,x)$ can be evaluated with a mantissa-exponent representation of the kind

$$\gamma_\mu(p,x) = m(\mu x, p) \cdot e^{n(\mu,x,p)}, \quad \text{where} \quad n(\mu,x,p) = -\mu x + p \log x\,. \tag{10}$$

The mantissa $m(\mu x, p)$ will be determined using either a series expansion, or a continued fraction, or, in the case $\mu < 0$, a recursive integration by parts. This yields three different computational methods for the evaluation of $\gamma_\mu(p,x)$. Notice that in the following, we will need to extend the representation (10) to the particular cases $x = 0$ and (only when $\mu > 0$) $x = +\infty$. For that purpose we set $m(0,p) = 0$, $n(\mu,0,p) = -\infty$ (taking the obvious convention that $0 \cdot e^{-\infty} = 0$), and in the case $\mu > 0$, we set $m(+\infty,p) = 1$ and $n(\mu,+\infty,p) = \log\Gamma(p) - p\log\mu$. The practical computation of $\log\Gamma(p)$ will be discussed in Section 4.

2.1.1 Series expansion

Writing the Taylor series expansion with order $p-1$ and integral remainder of the exponential function near zero we get

$$e^{\mu x} = \sum_{k=0}^{p-1} \frac{(\mu x)^k}{k!} + \int_0^{\mu x} \frac{(\mu x - t)^{p-1}}{(p-1)!} e^t \, dt \underset{s=x-t/\mu}{=} e^{\mu x} - \sum_{k=p}^{+\infty} \frac{(\mu x)^k}{k!} + \frac{\mu^p \, e^{\mu x}}{(p-1)!} \int_0^x s^{p-1} \, e^{-\mu s} \, ds \, .$$

This yields a series expansion of the generalized lower incomplete gamma function under the form $\gamma_\mu(p,x) = \gamma_\mu^{\text{ser}}(p,x)$ where

$$\gamma_\mu^{\text{ser}}(p,x) = m^{\text{ser}}(\mu x, p) \cdot e^{-\mu x + p \log x} \, , \quad \text{and} \quad m^{\text{ser}}(\mu x, p) = \sum_{k=0}^{+\infty} \frac{(p-1)!}{(k+p)!} (\mu x)^k \, . \tag{11}$$

Although the power series $m^{\text{ser}}(\mu x, p)$ defined above has an infinite radius of convergence, its convergence can be quite slow and numerically unstable according to the values of p and μx. It is suggested in [Press et al., 1992] to evaluate $\gamma_\mu(p,x)$ using (11) as soon as $\frac{|\mu x|}{p+1} < 1$; however, according to our experiments, a better convergence rate can be obtained by using a continued fraction development. Thus, we shall not use (11) in the algorithm we propose.

2.1.2 Continued fraction

Let us consider the confluent hypergeometric function M, defined by

$$M(a,b,z) = \sum_{n=0}^{+\infty} \frac{a^{(n)}}{b^{(n)}} \frac{z^n}{n!} \, , \quad \text{where} \quad \forall \alpha, \ \alpha^{(0)} = 1 \ \text{ and } \ \alpha^{(n)} = \alpha(\alpha+1)\cdots(\alpha+n-1) \, .$$

Since for any (b,z) we have $M(0,b,z) = 1$, Equation (11) rewrites as

$$\gamma_\mu(p,x) = \frac{M(1,p+1,\mu x)}{p \cdot M(0,p,\mu x)} \cdot e^{-\mu x + p \log x} \, . \tag{12}$$

As detailed in [Olver et al., 2010, DLMF, Cuyt et al., 2008, Jones and Thron, 1980], the ratio $\frac{M(a,b,z)}{M(a+1,b+1,z)}$ can be continued for any $z \in \mathbb{C}$, as soon as $a \notin \mathbb{Z} \setminus \mathbb{N}$ and $a - b \notin \mathbb{N}$. Under this assumption (which will be satisfied here, since we will consider the setting $a = 0$, $b = p$), and using the usual notation for continued fractions,

$$\frac{\alpha_1}{\beta_1 +} \, \frac{\alpha_2}{\beta_2 +} \, \frac{\alpha_3}{\beta_3 +} \, \cdots = \frac{\alpha_1}{\beta_1 + \frac{\alpha_2}{\beta_2 + \frac{\alpha_3}{\beta_3 + \cdots}}} \, ,$$

we get

$$\frac{M(a,b,z)}{M(a+1,b+1,z)} = 1 + \frac{u_1}{1+} \, \frac{u_2}{1+} \, \frac{u_3}{1+} \, \cdots \, ,$$

where $\forall n \geq 0$, $u_{2n+1} = \frac{(a-b-n)z}{(b+2n)(b+2n+1)}$, $u_{2n} = \frac{(a+n)z}{(b+2n-1)(b+2n)}$. Writing the inverse ratio (with $a = 0$ and $b = p$), and after basic manipulations of the continued fraction, we obtain

$$\frac{M(1,p+1,\mu x)}{p \cdot M(0,p,\mu x)} = \frac{a_1}{b_1 +} \, \frac{a_2}{b_2 +} \, \frac{a_3}{b_3 +} \, \cdots \, ,$$

where $a_1 = 1$ and $\forall n \geq 1$, $a_{2n} = -(p-1+n)\cdot\mu x$, $a_{2n+1} = n\cdot\mu x$ and $b_n = p-1+n$. Therefore, Equation (12) rewrites as $\gamma_\mu(p,x) = \gamma_\mu^{\text{cfrac}}(p,x)$, where

$$\gamma_\mu^{\text{cfrac}}(p,x) = m^{\text{cfrac}}(\mu x,p)\cdot e^{-\mu x + p\log x}\,, \quad \text{and} \quad m^{\text{cfrac}}(\mu x,p) = \frac{a_1}{b_1+}\,\frac{a_2}{b_2+}\,\frac{a_3}{b_3+}\,\ldots \tag{13}$$

The above defined continued fraction $m^{\text{cfrac}}(\mu x,p)$ can be evaluated thanks to the modified Lentz's method [Lentz, 1976, Thompson and Barnett, 1986] which is also described in [Press et al., 1992] and that we recall in Algorithm 1 for the reader's convenience, with however a slight adaptation of the initialization process since we observed some instabilities when using that described in [Press et al., 1992] (see comment in Algorithm 1). This continued fraction converges for any value of μx and the convergence is fast as it requires in general less than 20 approximants to converge, except when $\mu > 0$ and $\mu x \approx p$ (where it takes around p approximants) or when $\mu < 0$ and p is small (several hundred of approximants needed for $p \leq 20$ and $|\mu x| \leq 1000$). Note that $m^{\text{cfrac}}(\mu, px)$ becomes huge when μx is chosen too large compared to p, and numerical instabilities can appear. For that reason, we will restrict the use of (13) to a subdomain of the plane $(\mu x, p)$, as discussed in section 3.

Algorithm 1: Modified Lentz's method for continued fractions evaluation.

Input: Two real-valued sequences $\{a_n\}_{n\geq 1}$ and $\{b_n\}_{n\geq 1}$, with $b_1 \neq 0$.

Output: Accurate estimate f of the continued fraction $\frac{a_1}{b_1+}\,\frac{a_2}{b_2+}\,\frac{a_3}{b_3+}\,\cdots$

Initialization:
$d_m \leftarrow 10^{-300}$ `// Number near the minimal floating-point value`

$f \leftarrow \frac{a_1}{b_1}$; $C \leftarrow \frac{a_1}{d_m}$; $D \leftarrow \frac{1}{b_1}$; $n \leftarrow 2$ `// see the algorithm footnote`

repeat
- $D \leftarrow D\cdot a_n + b_n$
- **if** $D = 0$ **then** $D \leftarrow d_m$
- $C \leftarrow b_n + \frac{a_n}{C}$
- **if** $C = 0$ **then** $C \leftarrow d_m$
- $D \leftarrow \frac{1}{D}$
- $\Delta \leftarrow C\cdot D$
- $f \leftarrow f\cdot\Delta$
- $n \leftarrow n+1$

until $|\Delta - 1| < \varepsilon_{machine}$

return f

In the initialization step, we manually performed the first pass $n = 1$ of the modified Lentz's algorithm, since we observed some instabilities with the initialization $f = C = d_m$, $D = 0$, presented in [Press et al., 1992]. Indeed, the setting $C = d_m$ may yield $C = +\infty$ after the pass $n = 1$ (when a_1/d_m exceed the highest representable number), and then $\Delta = f = +\infty$, which propagates through the next iterations. By computing manually the first pass, even when the initialization $C = a_1/d_m$ yields $C = +\infty$, the pass $n = 2$ yields $C = b_2 + a_2/C = b_2$, which has a finite value.

2.1.3 Integration by parts

Since we only consider integer values of the parameter p, the generalized lower incomplete gamma function $\gamma_\mu(p,x)$ can be written as a closed-form formula using a recursive integration

by parts. Considering the case $\mu < 0$, one gets

$$\gamma_\mu(p,x) = \gamma_\mu^{\mathrm{ibp}}(p,x) := m^{\mathrm{ibp}}(\mu x, p) \cdot e^{-\mu x + p \log x}\,,$$
$$\text{where} \quad m^{\mathrm{ibp}}(\mu x, p) = \frac{1}{\mu x}\left(\tfrac{(p-1)!\, e^{\mu x}}{(\mu x)^{p-1}} - \sum_{k=0}^{p-1} \tfrac{(p-1)!(\mu x)^{-k}}{(p-1-k)!} \right). \quad (14)$$

Although the computation of $\gamma_\mu^{\mathrm{ibp}}(p,x)$ is not efficient in general, it happens to be faster than $\gamma_\mu^{\mathrm{cfrac}}(p,x)$ for small values of p. We must however be carefull when computing the alternating sum $m^{\mathrm{ibp}}(\mu x, p)$ since, as usual with alternating sums, it may suffer from dramatic cancellation errors.

Let $t = |\mu x| > 0$, we rewrite (14) into

$$m^{\mathrm{ibp}}(-t,p) = \frac{1}{t}\left(\tfrac{(-1)^p (p-1)!\, e^{-t}}{t^{p-1}} + s(t) \right), \quad \text{where} \quad s(t) = \sum_{k=0}^{p-1} (-1)^k \frac{(p-1)!\, t^{-k}}{(p-1-k)!}\,. \quad (15)$$

By grouping by two the consecutive terms with indexes $k = 2l$ and $k = 2l+1$ of the alternating sum $s(t)$, we get

$$s(t) = \tilde{s}(t) := \sum_{l=0}^{\lfloor \frac{p-2}{2} \rfloor} \frac{(p-1)!\, t^{-(2l+1)}}{(p-1-2l)!}\,(t - (p-1-2l)) + \varepsilon_p(t)\,, \quad (16)$$

where $\lfloor z \rfloor$ denotes the integer part of z, and the residual term $\varepsilon_p(t)$ is defined by

$$\varepsilon_p(t) = \begin{cases} (p-1)!\, t^{-(p-1)} & \text{if } p \text{ is odd} \\ 0 & \text{otherwise.} \end{cases}$$

Let us now assume that $t \geq \max(1, p-1)$. First, using $t \geq p-1$, we see that all terms in the sum $\tilde{s}(t)$ are nonnegative, so that we can evaluate $\tilde{s}(t)$, which has exactly the same value as the alternating sum $s(t)$, without any cancellation error using (16). It follows that, when p is even, we have

$$m^{\mathrm{ibp}}(-t,p) = \frac{1}{t}\left(\frac{(p-1)! e^{-t}}{t^{p-1}} + \tilde{s}(t) \right),$$

which is a sum of positive terms, so that it does not suffer from cancellation error. When p is odd, (15) yields

$$m^{\mathrm{ibp}}(-t,p) = \frac{1}{t}\left(-\frac{(p-1)! e^{-t}}{t^{p-1}} + \tilde{s}(t) \right). \quad (17)$$

Noting $\alpha(t) = \frac{(p-1)! e^{-t}}{t^{p-1}}$ and using the fact that $t \geq 1$, we get

$$\frac{\tilde{s}(t)}{\alpha(t)} \geq \frac{\varepsilon_p(t)}{\alpha(t)} = \exp(t) \geq \exp(1)\,,$$

which ensures that no cancellation error occurs when computing the difference between $\tilde{s}(t)$ and $\alpha(t)$, involved in (17). Finally, we are able to evaluate (14) without cancellation in the region $t \geq \max(1, p-1)$.

Last, from $t > p-1$, we infer that the sequence $\{a_k(t)\}_{k\geq 0}$ defined by

$$\forall k \geq 0, \quad a_k(t) = \begin{cases} \frac{(p-1)!\, t^{-k}}{(p-1-k)!} & \text{if } k \leq p-1 \\ 0 & \text{otherwise,} \end{cases}$$

is nonincreasing, with limit 0. It follows that the remainder $r_n(t) = \sum_{k=n+1}^{+\infty} (-1)^k a_k(t)$ of the alternating series $s(t) = \sum_{k=0}^{+\infty} (-1)^k a_k(t)$ satisfies $|r_n(t)| \leq a_{n+1}(t)$, so that we can numerically estimate $s(t)$ with the partial sum $s_n(t) = \sum_{k=0}^{n} (-1)^k a_k(t)$ as soon as

$$a_{n+1}(t) \leq |s_n(t)| \cdot \varepsilon_{\text{machine}}\,,$$

which may occur for $n < p-1$, making possible in that case to save some computation time. In practice, we compute $s(t) = \tilde{s}(t)$ with (16) instead of (15), but this stopping criterion can be easily evaluated at each iteration of the summation procedure. Indeed, remarking that the sequence $\{a_{2l}(t) - a_{2l+1}(t)\}_{l\geq 0}$ is positive and nonincreasing (because $t > p-1$), we get

$$\forall l \in \mathbb{N}, \quad a_{2l+2}(t) \leq a_{2l}(t) - a_{2l+1}(t) + a_{2l+3}(t) \leq a_{2l}(t) - a_{2l+1}(t)\,,$$

so that

$$\forall l \in \mathbb{N}, \quad |r_{2l+1}(t)| \leq a_{2l+2}(t) \leq |a_{2l}(t) - a_{2l+1}(t)|\,.$$

This yields Algorithm 2.

Algorithm 2: Accurate evaluation of $m^{\text{ibp}}(\mu x, p) = \frac{1}{\mu x}\left(\frac{(p-1)!\, e^{\mu x}}{(\mu x)^{p-1}} - \sum_{k=0}^{p-1} \frac{(p-1)!(\mu x)^{-k}}{(p-1-k)!}\right)$.

Input: Two real numbers $x \in \mathbb{R}_+$, $\mu < 0$, and a positive integer p, satisfying $|\mu x| > \max(1, p-1)$.

Output: An accurate estimate of $m^{\text{ibp}}(\mu x, p)$.

Initialization: $t \leftarrow |\mu x|$; $c \leftarrow \frac{1}{t}$; $d \leftarrow p-1$; $s \leftarrow c \cdot (t-d)$; $l \leftarrow 1$; *stop* $\leftarrow$ *false*

repeat

- $c \leftarrow \frac{d(d-1)}{t^2}$
- $d \leftarrow d-2$
- $\Delta \leftarrow c(t-d)$ `// Now` $\Delta = a_{2l}(t) - a_{2l+1}(t)$
- $s \leftarrow s + \Delta$ `// Now` $s = s_{2l+1}(t) = \sum_{k=0}^{2l+1} (-1)^k a_k(t)$
- **if** $\Delta < s \cdot \varepsilon_{machine}$ **then** *stop* $\leftarrow$ *true*
- $l \leftarrow l+1$

until $l > \lfloor \frac{p-2}{2} \rfloor$ *or stop*

if *(not stop) and (p is odd)* **then** $s \leftarrow s + \frac{d\,c}{t}$ `// add the term` $\varepsilon_p(t) = (p-1)!\, t^{-(p-1)}$

return $\frac{1}{t}\left((-1)^p \cdot e^{-t+\log(p-1)!-(p-1)\log(t)} + s\right)$

There is a more elegant way to avoid the cancellation errors in the computation of $s(t)$, inspired from Horner's algorithm for polynomial evaluation. It consists in computing

$$s(t) = 1 - \frac{p-1}{t} \cdot \left(1 - \frac{p-2}{t} \cdot \left(1 - \frac{p-3}{t} \cdot \left(\dots\left(1 - \frac{1}{t}\right)\right)\right)\dots\right)\,,$$

or more precisely, $s(t) = v_{p-1}(t)$, where $\{v_n(t)\}_{n\geq 1}$ is the sequence defined recursively by

$$\forall n \geq 1, \quad v_n(t) = \begin{cases} 1 - \frac{1}{t} & \text{if } n = 1\,, \\ 1 - \frac{n}{t} \cdot v_{n-1}(t) & \text{if } n \geq 2\,. \end{cases}$$

Assuming $t \geq 2p$, one can show that the terms of $\{v_n\}_{1 \leq n \leq p-1}$ remain in $(\frac{1}{2}, 1)$, so that they can be evaluated without cancellation errors. However, a drawback of this approach is the absence of a simple stopping criterion making possible to end up the computation of $s(t)$ before computing all the first $p-1$ terms of the sequence $\{v_n\}_{n \geq 1}$.

2.1.4 Algorithm for the evaluation of the generalized lower incomplete gamma function

The evaluation of $\gamma_\mu(x,p)$ using one of the computation methods presented above can be done using Algorithm 3. This algorithm returns a mantissa-exponent representation (m,n) of $\gamma_\mu(x,p)$, such as $\gamma_\mu(x,p) = m \cdot e^n$, and returns $m = 0, n = -\infty$, when $\gamma_\mu(x,p) = 0$ (this will be more generally the case for the mantissa-exponent representations returned by all algorithms we propose).

Algorithm 3: Evaluation of $\gamma_\mu(x,p) = \int_0^x s^{p-1}\, e^{-\mu s}\, ds$ using a series expansion, a continued fraction, or a recursive integration by parts.

Input: Two numbers $x \in \mathbb{R}_+ \cup \{+\infty\}$, $\mu \in \mathbb{R} \setminus \{0\}$, and a positive integer p. Notice that the value $x = +\infty$ is allowed only when $\mu > 0$.

Output: Two numbers $m \in \mathbb{R}$ and $n \in \mathbb{R} \cup \{-\infty\}$ such as $\gamma_\mu(p,x) = m \cdot e^n$.

if $x = 0$ **then** $(m,n) \leftarrow (0, -\infty)$
else if $x = +\infty$ *and* $\mu > 0$ **then** $(m,n) \leftarrow (1, \log\Gamma(p) - p\log\mu)$
else
  switch *choice of the evaluation method for the mantissa* **do**
    case *series expansion*
      $m \leftarrow m^{\text{ser}}(\mu x, p)$ `// using Equation (11)`
    case *continued fraction*
      $m \leftarrow m^{\text{cfrac}}(\mu x, p)$ `// using Equation (13) and Algorithm 1`
    case *recursive integration by parts (only when* $\mu < 0$ *and* $|\mu x| > \max(1, p-1)$*)*
      $m \leftarrow m^{\text{ibp}}(\mu x, p)$ `// using Equation (14)`
    end
  endsw
  $n \leftarrow -\mu x + p \log x$
end
return (m,n)

2.2 Evaluation of the generalized upper incomplete gamma function

Let $p \geq 1$, $x > 0$ and $\mu > 0$. The evaluation of $\Gamma_\mu(p,x)$ can be done thanks to another fraction continuation as detailed in [Abramowitz and Stegun, 1964, Press et al., 1992]. We accordingly set $\Gamma_\mu(p,x) = \Gamma_\mu^{\text{cfrac}}(p,x)$, where

$$\Gamma_\mu^{\text{cfrac}}(p,x) = M^{\text{cfrac}}(\mu x, p) \cdot e^{-\mu x + p\log x}, \quad M^{\text{cfrac}}(\mu x, p) = \frac{\alpha_1}{\beta_1 +} \frac{\alpha_2}{\beta_2 +} \frac{\alpha_3}{\beta_3 +} \cdots, \tag{18}$$

with $\alpha_1 = 1$, $\alpha_n = -(n-1)\cdot(n-p-1)$ for any $n > 1$, and $\beta_n = \mu x + 2n - 1 - p$ for any $n \geq 1$. The continued fraction $M^{\text{cfrac}}(\mu x, p)$ can be numerically evaluated using again

Algorithm 1, except when $\beta_1 = 1$ (i.e. when $\mu x = p - 1$), in which case we must use

$$M^{\mathrm{cfrac}}(\mu x, p) = \frac{\alpha_1}{M}, \quad \text{where} \quad M = \frac{\alpha_2}{\beta_2 +} \, \frac{\alpha_3}{\beta_3 +} \, \frac{\alpha_2}{\beta_2 +} \, \cdots \quad \text{and} \quad \beta_2 \neq 0\,. \tag{19}$$

We extend the computation of $\Gamma_\mu(p, x)$ to the cases $x = 0$ and $x = +\infty$ using a similar approach as for $\gamma_\mu(p, x)$. This yields Algorithm 4.

Algorithm 4: Evaluation of $\Gamma_\mu(x, p) = \int_x^{+\infty} s^{p-1}\, e^{-\mu s}\, ds$ using a continued fraction.

Input: Two numbers $x \in \mathbb{R}_+ \cup \{+\infty\}$, $\mu > 0$, and a positive integer p.

Output: Two numbers $M \in \mathbb{R}$ and $N \in \mathbb{R} \cup \{-\infty\}$ such that $\Gamma_\mu(p, x) = M \cdot e^N$.

if $x = 0$ **then** $(M, N) \leftarrow (1, \log\Gamma(p) - p \log \mu)$
else if $x = +\infty$ **then** $(M, N) \leftarrow (0, -\infty)$
else
  if $\mu x \neq p - 1$ **then**
    $M \leftarrow M^{\mathrm{cfrac}}(\mu x, p)$ `// using Equation (18) and Algorithm 1`
  else
    $M \leftarrow M^{\mathrm{cfrac}}(\mu x, p)$ `// using Equation (19) and Algorithm 1`
  end
  $N \leftarrow -\mu x + p \log x$
end
return (M, N)

2.3 Accuracy of the mantissa-exponent representation and its conversion into scientific notation

Thanks to Algorithms 3 and 4, we are now able to evaluate the integrals $\gamma_\mu(p, x)$ and $\Gamma_\mu(p, x)$ with a mantissa-exponent representation of type $\rho \cdot e^\sigma$, where, in absence of additional multiprecision library, the quantities ρ and σ are evaluated in standard double floating-point precision. Alhough this representation considerably extends the range over which the integrals $\gamma_\mu(p, x)$ and $\Gamma_\mu(p, x)$ can be represented (in comparison with a direct evaluation of those integrals in double precision), the evaluation of the term $\rho \cdot e^\sigma$ may suffer from important loss of precision, according to the values of ρ and σ. Indeed, using a first order approximation of the relative error associated to the term $\rho \cdot e^\sigma$, we get

$$\left|\frac{\Delta(\rho \cdot e^\sigma)}{\rho \cdot e^\sigma}\right| \approx \left|\frac{\Delta\rho}{\rho}\right| + \left|\frac{\Delta(e^\sigma)}{e^\sigma}\right| = \left|\frac{\Delta\rho}{\rho}\right| + |\Delta\sigma| = \left|\frac{\Delta\rho}{\rho}\right| + |\sigma| \cdot \left|\frac{\Delta\sigma}{\sigma}\right| := E \tag{20}$$

where $|\Delta X|$ and $|\Delta X/X|$ respectively denote the absolute and relative errors between the actual value of X and its computed value. Unfortunately, we see that E gets large as $|\sigma|$ increases, and since the quantity ρ and σ are in the best case estimated at the machine precision (i.e. $|\Delta\rho/\rho| = |\Delta\sigma/\sigma| = \varepsilon_{\mathrm{machine}}$), we have the lower bound

$$E \geq E_{\min} := 1 + |\sigma| \cdot \varepsilon_{\mathrm{machine}}\,. \tag{21}$$

For instance, when $\sigma \approx 4503.5$, the best relative accuracy that can be expected is $E_{\min} \approx 4504.5 \times 2.22 \cdot 10^{-16} \approx 10^{-12}$ using the IEEE 754 Standard for Floating-Point Arithmetic on a 64-bits computer (which yields $\varepsilon_{\mathrm{machine}} = 2.22 \cdot 10^{-16}$). Since in Algorithms 3 and 4, the exponent σ associated to the computation of $\gamma_\mu(p, x)$ or $\Gamma_\mu(p, x)$ is

given by $\sigma = -\mu x + p \log x$, we can already establish some theoritical bounds for the relative error E reachable by these algorithms with respect to μ, p, x. This is done in Figure 1, and our numerical experiments performed in Section 2.4 (Figures 2 and 3) will show that this theoritical bound is in practice attained by our algorithms.

Unsurprisingly, the same limitation arises when we format the quantity $\rho \cdot e^{\sigma}$ in scientific notation (that is $\rho \cdot e^{\sigma} = a \cdot 10^b$, where $a \in [1, 10)$ and $b \in \mathbb{Z}$). This operation can be done using

$$a = \rho \cdot e^{c - \lfloor c \rfloor}, \quad b = \lfloor c \rfloor, \quad \text{where} \quad c = \frac{\sigma}{\log(10)} + \log_{10}(\rho). \tag{22}$$

This time, the evaluation of a suffers from the loss of precision occuring in the evalution of $c - \lfloor c \rfloor$, the fractional part of c, simply because all digits used to represent the integer part of c are as many digits which are lost in the evaluation of its fractional part. Assuming that $\Delta b = 0$ (i.e. that the quantity c is estimated with at least one digit of precision), we get

$$\left| \frac{\Delta(a \cdot 10^b)}{a \cdot 10^b} \right| = \left| \frac{\Delta a}{a} \right| \approx \left| \frac{\Delta \rho}{\rho} \right| + |\Delta c| \leq (1 + |c|) \cdot \varepsilon_{\text{machine}}, \tag{23}$$

which is similar to (21). Although some numerical strategies to retrieve several significant digits may be developed, the most straightforward way to compensate the loss of precision of those two representations would be to evaluate σ and ρ with a more generous floating point precision, which can be easily done using the *x86 Extended Precision Format* (which corresponds to the long double datatype in C langage, and yields $\varepsilon_{\text{machine}} = 1.08 \cdot 10^{-19}$), or using some multiprecision library (such as the *GNU MPFR* C-library, which provides an exact control of the number of significant number of bits used for each variable).

2.4 Selection of a fast and accurate computational method according to the parameters

We detailed in (11), (13), (14), (18), several methods for the numerical evaluation of γ_μ and Γ_μ, with a mantissa-exponent representation. Let us now focus on the accuracy and the computation time of these methods, according to the value of the parameters μ, p, x. For that purpose, we evaluated $\gamma_\mu^{\text{ser}}(x,p)$, $\gamma_\mu^{\text{cfrac}}(x,p)$, $\gamma_\mu^{\text{ibp}}(x,p)$ and $\Gamma_\mu^{\text{cfrac}}(x,p)$ for a large range of parameters:

$$\mu = \pm 1, \quad x \in [0, 1000] \cap \mathbb{N}, \quad p \in [1, 1000] \cap \mathbb{N},$$

more precisely, $\gamma_\mu^{\text{cfrac}}(x,p)$ and $\gamma_\mu^{\text{ser}}(x,p)$ were computed for all theses values of $(\mu x, p)$, but $\gamma_\mu^{\text{ibp}}(x,p)$ was computed only in the case $\mu < 0$, $|\mu x| > \max(1, p-1)$, in accordance to the discussion made in Section 2.1.3, and $\Gamma_\mu^{\text{cfrac}}(x,p)$ was computed only in the case $\mu x \geq 0$.

For each tested value of (μ, x, p) and each evaluation method, we compared the computed values of $\gamma_\mu(p,x)$ and $\Gamma_\mu(p,x)$ (formatted in scientific notation using (22)) to those computed with Maple$^{\text{TM}}$ (version 17), with 30 significant decimal digits (which requires large amounts of memory and a long computation time), using the instructions

```
evalf(Int(s^(p-1)*exp(-mu*s),s=0..x,digits=30));
evalf(Int(s^(p-1)*exp(-mu*s),s=x..infinity,digits=30));
```

The values of the integrals estimated with Maple were used as references to evaluate the relative accuracy reached for each method, and each tested value of μ, x, p. The results are displayed in Figure 2 and 3. We observed from these experiments that for each μ, x, p, at least one computation method yields a relative error less than $2 \cdot 10^{-12}$ when using the standard double floating-point precision in C langage, and only this many in the region

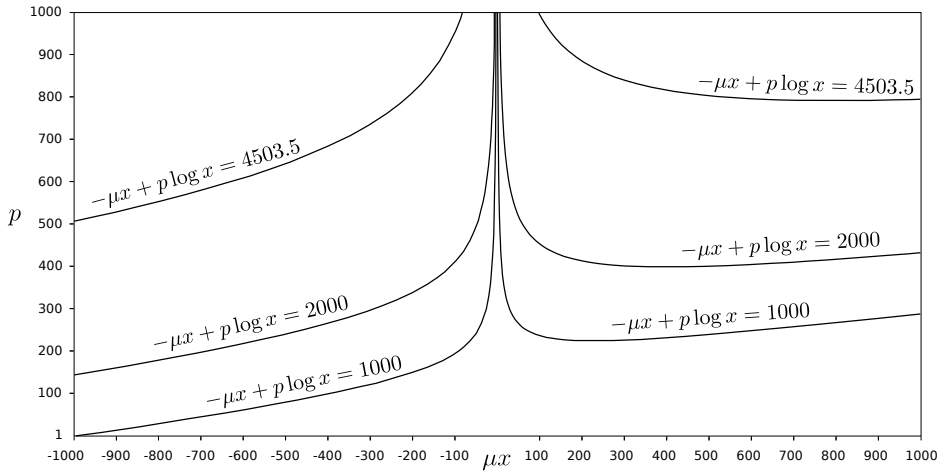


Figure 1: Isovalues of the exponent $\sigma = (\mu, x, p) \mapsto -\mu x + p \log x$ ($\mu = -1$, left) and ($\mu = 1$, right). In this figure, we display some isovalues of the exponent part of γ_μ and Γ_μ computed with Algorithm 3 and 4, and whose parametric equation is recalled above. As discussed in Section 2.3, the relative precision related to the evaluation of γ_μ and Γ_μ using a mantissa-exponent representation $\rho \cdot e^\sigma$ deteriorates as σ increases. Indeed, even when ρ and σ are estimated with the best available precision ($\Delta\rho/\rho = \Delta\sigma/\sigma = \varepsilon_{\text{machine}}$), the best relative precision that we can expect for the evaluation of $\rho \cdot e^\sigma$ is $E_{\min} = (1 + |\sigma|) \cdot \varepsilon_{\text{machine}}$, as stated in (21). The standard precision (on a 64-bits computer) is $\varepsilon_{\text{machine}} = 2.22 \cdot 10^{-16}$, so that $E_{\min} \geq 10^{-12}$ as soon as $\sigma \geq 4503.5$. Interestingly enough, the curve corresponding to the isovalue $\sigma(\mu, x, p) = 4503.5$ fits particularly well with the frontier of the domain where Algorithm 3 and 4 yield a relative accuracy more than 10^{-12} (see Figure 2). When using the extended double floating-point precision (corresponding to the long double datatype in C langage), we have $\varepsilon_{\text{machine}} = 1.08 \cdot 10^{-19}$, so that we get $1 \cdot 10^{-16} \leq E_{\min} \leq 2 \cdot 10^{-16}$, in the region $1000 \leq \sigma(\mu, x, p) \leq 2000$ (delimited by the two other isovalues represented above). Again, these two frontiers were experimentally observed in Figure 3, where we measure the relative error reached by our algorithms using extended double precision.

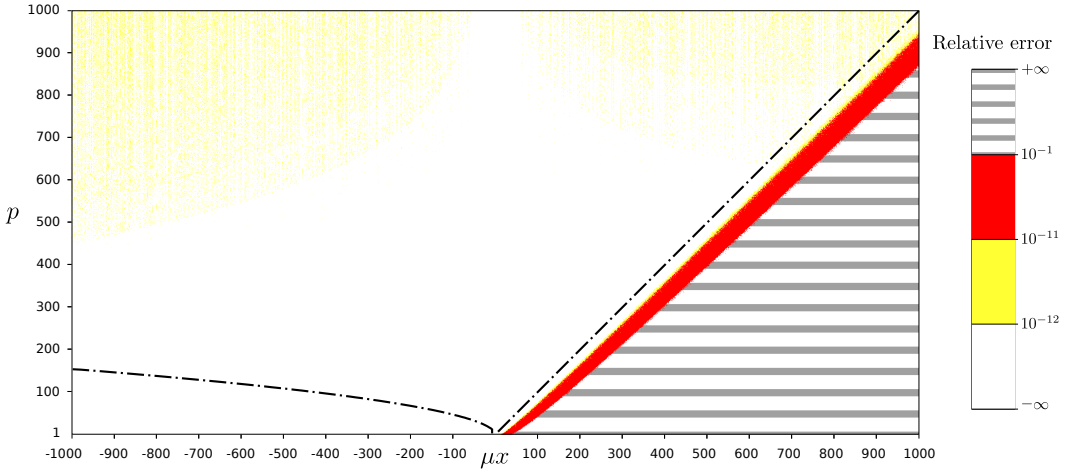
(μ, x, p) where the exponent part is above 4503.5 (this region is represented in Figure 1). Outside of this region, at least one method yields a relative error less than 10^{-12} (more precisely close to 10^{-13}). Interestingly enough, the observed relative errors perfectly match with the bound (21) predicted in Section 2.3, showing that, in practice, the accuracy of Algorithms 3 and 4 is only limited by the mantissa-exponent representation. When using the extended double precision (see Figure 3), we improve the precision of three orders of magnitude, and again, the selection of the most accurate method yields a relative error which is very close to that predicted in Section 2.3, so that we could expect even more accuracy with higher precision computer arithmetics.

By measuring the computation time for each method and each value of (μ, x, p), and thanks to the control of the relative error presented in Figure 2, we derived a partition into three domains of the plan $(\mu x, p)$ which makes possible the fast and accurate computation of at least one quantity between $\gamma_\mu(p, x)$ and $\Gamma_\mu(p, x)$. We accordingly propose a parametric equation for the boundary of those three domains, given by

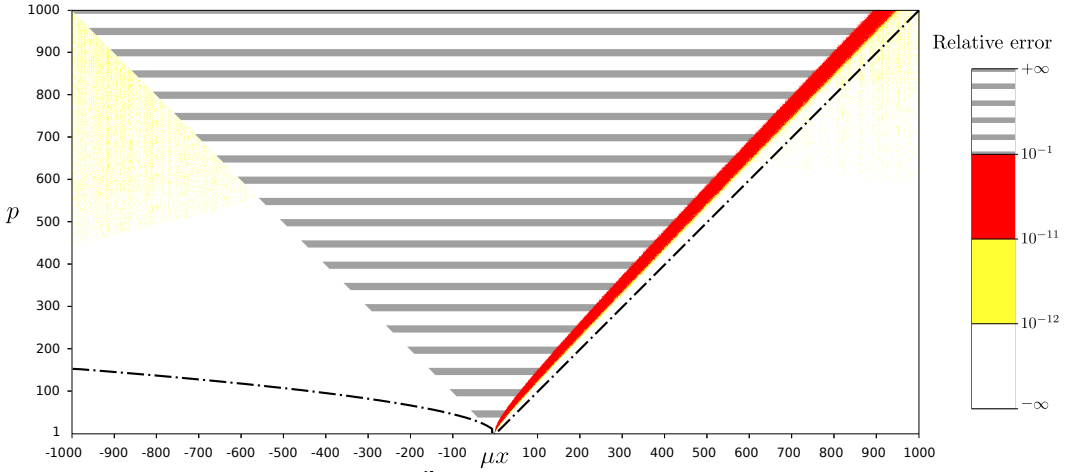
$$\forall \mu x \in \mathbb{R} \cup \{+\infty\}, \quad p_{\lim}(\mu x) = \begin{cases} 5\sqrt{|\mu x|} - 5 & \text{if } \mu x < -9\,, \\ 0 & \text{if } -9 \leq \mu x \leq 0\,, \\ \mu x & \text{otherwise.} \end{cases} \tag{24}$$

This equation can be used in the following way (as illustrated in Figure 4):

- when $p \geq p_{\lim}(\mu x)$: compute $\gamma_\mu^{\text{cfrac}}(p, x)$;
- otherwise: compute $\gamma_\mu^{\text{ibp}}(p, x)$ when $\mu < 0$, or $\Gamma_\mu^{\text{cfrac}}(p, x)$ when $\mu > 0$.

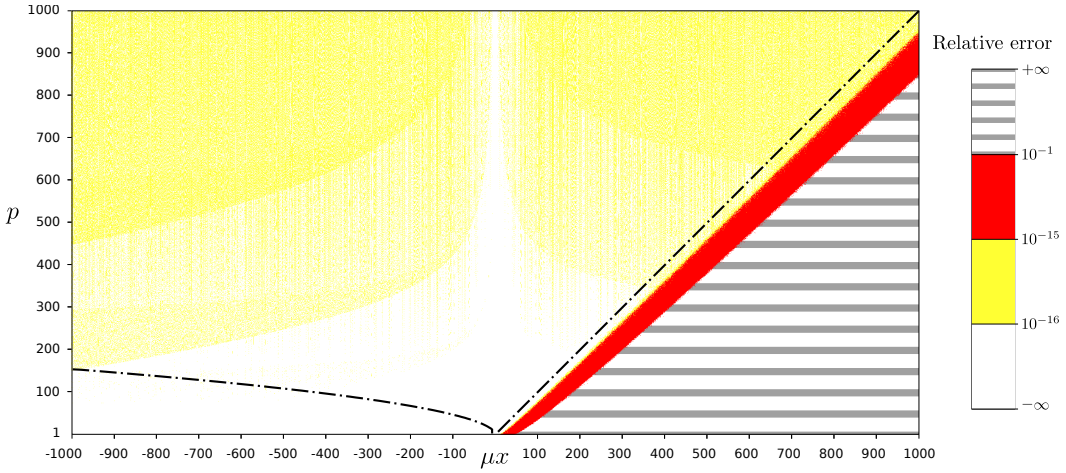


(a) relative error for $\gamma_\mu^{\mathbf{cfrac}}$.

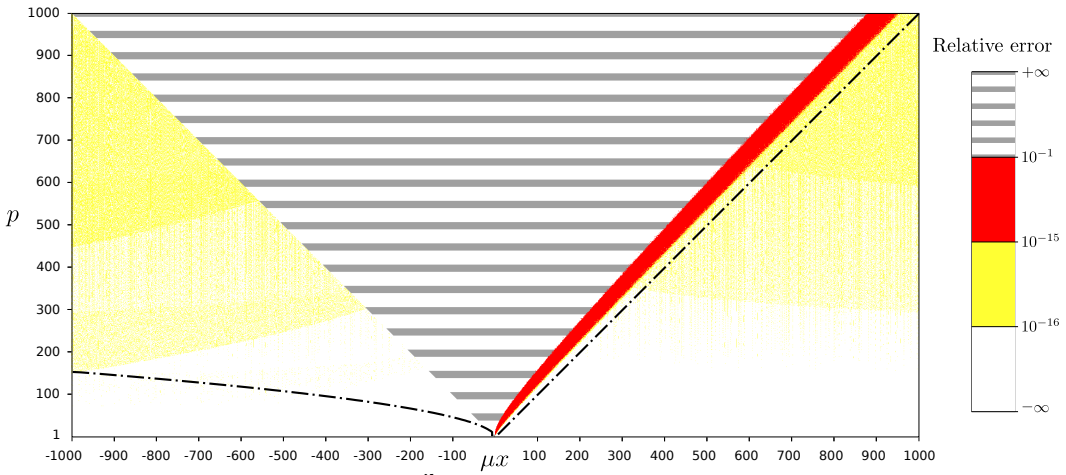


(b) relative error for $\gamma_\mu^{\mathbf{ibp}}$ ($\mu x < 0$, left) and $\Gamma_\mu^{\mathbf{cfrac}}$ ($\mu x > 0$, right).

Figure 2: Control of the relative error associated to the computation of $\gamma_\mu(p,x)$ and $\Gamma_\mu(p,x)$ using different methods. We used Algorithms 3 and 4 (implemented in C langage, using the standard double datatype on a 64-bits computer) to compute $\gamma_\mu^{\text{cfrac}}(p,x)$, $\gamma_\mu^{\text{ibp}}(p,x)$, $\Gamma_\mu^{\text{cfrac}}(p,x)$ for $\mu = \pm 1$, x integer in $[0, 10^3]$, and p integer in $[1, 10^3]$. Using Maple, we measured the relative errors reached by each method. The error reached by $\gamma_\mu^{\text{cfrac}}$ is displayed in (a), the error reached by γ_μ^{ibp} (computed only for $\mu < 0$ and $|\mu x| \geq \max(1, p-1)$) is displayed in the left-side of (b), and the error reached by $\Gamma_\mu^{\text{cfrac}}$ (computed only for $\mu > 0$) is displayed in the right-side of (b). The dashed curve (see its parametric equation in (24)) splits the plan $(\mu x, p)$ into three domains, each one is associated to one of the three computation methods (γ_μ^{ibp} : left, $\gamma_\mu^{\text{cfrac}}$: middle, $\Gamma_\mu^{\text{cfrac}}$: right), and corresponds to the region where the method is at the same time fast and accurate, compared to the others (see also Figure 4). We can see that inside each one of the three domains, the corresponding computation method reaches a relative error always less than 10^{-11} (the actual maximal observed error is in fact close to $2 \cdot 10^{-12}$), and most of the time less than 10^{-12} (in practice close to 10^{-13}). We also observe that the boundary of the region where the relative error (of the selected algorithm) is greater than 10^{-12} coincides almost perfectly with the isovalue $\sigma = 4503.5$ displayed in Figure 1, showing that the relative error is in practice only limited by the mantissa-exponent representation (see discussion in Section 2.3). This limitation can be compensated by using a more precise floating-point representation, as shown in Figure 3.



(a) relative error for $\gamma_\mu^{\mathbf{cfrac}}$.



(b) relative error for $\gamma_\mu^{\mathbf{ibp}}$ ($\mu x < 0$, left) and $\Gamma_\mu^{\mathbf{cfrac}}$ ($\mu x > 0$, right).

Figure 3: Improving the accuracy of $\gamma_\mu(p,x)$ and $\Gamma_\mu(p,x)$ using extended double precision. We performed here the same experiment as in Figure 2, using a C implementation of Algorithms 3 and 4 with extended double precision (corresponding to the long double datatype in C langage, with machine epsilon $\varepsilon_{\text{machine}} = 1.08{\cdot}10^{-19}$, which is around three orders of magnitude better than the standard double precision). We see that our algorithm fully benefits from this additional precision, since the observed relative error is decreased of around three magnitude orders as well. Besides, we observe again that the main limitation to the precision remains that involved by the mantissa-exponent representation, since, within each domain, the level lines of the relative error of the selected algorithm match very well to the isovalues of σ, and the value of the relative error is in practice very similar to that predicted in (21). This suggests that the error bounds we obtain could be reduced even further by simply using a more precise floating-point arithmetic (for instance the GNU MPFR C library).

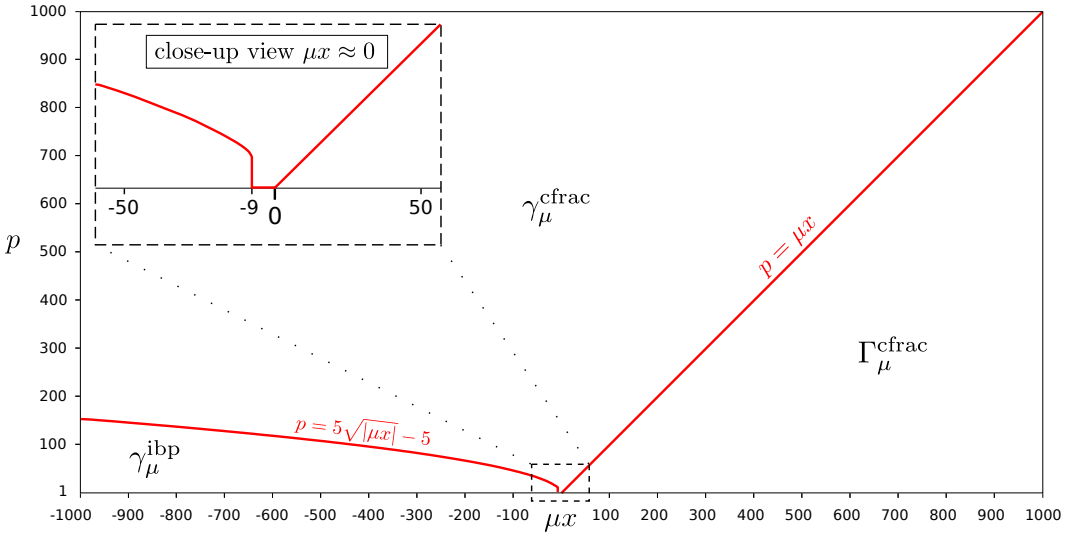


Figure 4: Numerical evaluation of the generalized lower or upper incomplete gamma functions. In this figure, we display the graph (red curve) of the frontier p_{lim} defined in (24). This curve delimits the plan $(\mu x, p)$ into three regions, each corresponding to the region where one of the three computation methods $\gamma_\mu^{\text{cfrac}}$, γ_μ^{ibp} and $\Gamma_\mu^{\text{cfrac}}$ is optimal (in the sense that its computation is fast and reaches a good relative error). According to our partition, and as indicated on the figure, γ_μ^{ibp} must be computed in the bottom-left region, $\gamma_\mu^{\text{cfrac}}$ in the middle region, and $\Gamma_\mu^{\text{cfrac}}$ in the bottom-right region. More precisely, we select $\gamma_\mu(p, x)$ as soon as $p \geq p_{\text{lim}}(\mu x)$, otherwise we select $\gamma_\mu^{\text{ibp}}(p, x)$ when $\mu < 0$, or $\Gamma_\mu^{\text{cfrac}}(p, x)$ when $\mu > 0$. A close-up view near $\mu x = 0$ shows that $\gamma_\mu^{\text{cfrac}}$ is automatically selected near $\mu x = 0$ since $p \geq 1 \geq p_{\text{lim}}(\mu x) = 0$ when $-9 \leq \mu x \leq 0$ (this avoids the computation of $\gamma_\mu^{\text{ibp}}(p, x)$ for $|\mu x| < \max(1, p - 1)$, which is not allowed according to the discussion of Section 2.1.3).

3 Evaluation of the generalized incomplete gamma function

As stated before, the accurate evaluation of $I_{x,y}^{\mu,p}$ raises two different issues. First, this integral can be approximated as the difference $A - B$ between two terms $A \geq B \geq 0$ involving the evaluation of the generalized upper and lower incomplete gamma functions γ_μ and Γ_μ, thanks to the relations (8)-(9). Therefore, we must select which difference can be accurately and efficiently computed according to the parameters μ, x, y, p; this selection is discussed in Section 3.1. Second, we must be careful that the accurate evaluation of A and B is not sufficient to garantee an accurate evalutation of the difference $A - B$, because cancellation errors arise when A and B are too close to each other, which happens in practice when $x \approx y$. In that case, we propose to approximate the integral $I_{x,y}^{\mu,p}$ using a first order trapezoidal approximation, as discussed in Section 3.2. In order to decide which approximation must be used (between the computation by means of a difference $A - B$, or a first order approximation), we propose in Section 3.3 a simple criterion based on the absolute errors. This study results in Algorithm 6 for the evaluation of $I_{x,y}^{\mu,p}$.

3.1 Computing $I_{x,y}^{\mu,p}$ as a difference of generalized incomplete gamma functions

According to the numerical experiments presented in Figures 2-4, we are now able to decide which integral between $\gamma_\mu(p,x)$ and $\Gamma_\mu(p,x)$ can be computed and how it must be evaluated, according to the value of (μ,p,x), to reach at the same time a good accuracy and a small computation time. We used these results to derive which difference $I_{\text{diff}} = A - B$ should be considered to approximate $I_{x,y}^{\mu,p}$, according to x, y, μ, p. The results are gathered in Table 1. A mantissa-exponent representation of I_{diff} is obtained from the mantissa-exponent representations (m_A, n_A) and (m_B, n_B) of A and B (returned by Algorithm 3 or 4):

$$I_{\text{diff}} = \rho_{\text{diff}} \cdot e^{\sigma_{\text{diff}}}, \quad \text{where} \quad \rho_{\text{diff}} = m_A - m_B\, e^{n_B - n_A}, \quad \sigma_{\text{diff}} = n_A\,. \tag{25}$$

If I_{diff} was computed directely as the difference $A - B$ (which is in practice difficult because A and B may not be representable in the floating-point arithmetic), the absolute error $|\Delta I_{\text{diff}}| = |I_{x,y}^{\mu,p} - I_{\text{diff}}|$ would satisfy $|\Delta I_{\text{diff}}| \approx A\,\varepsilon_{\text{machine}}$ (since $A \geq B \geq 0$), but this is not the case here, due to the mantissa-exponent representation used for A, B, and I_{diff}. A more precise estimation of $|\Delta I_{\text{diff}}|$ is obtained by using a first order approximation

$$|\Delta I_{\text{diff}}| \approx (|\Delta\rho_{\text{diff}}| + |\rho_{\text{diff}}\,\Delta\sigma_{\text{diff}}|)\, e^{\sigma_{\text{diff}}}\,.$$

Besides, as discussed in Figures 2 and 3, we can reasonably consider that the relative precision reached for the quantities m_A, n_A, m_B and n_B is close to the machine epsilon (which is of course not the case for $m_A \cdot e^{n_A}$ and $m_B \cdot e^{n_B}$, as discussed in Section 2.3). It follows that the quantity σ_{diff} is also evaluated at the machine precision, and thus, the corresponding absolute error is $|\Delta\sigma_{\text{diff}}| = |\sigma_{\text{diff}}| \cdot \varepsilon_{\text{machine}}$. The same kind of equality does not holds for the mantissa ρ_{diff}, whose numerical evaluation suffers from an additional loss of precision due to the exponential term. Indeed, using again a first order approximation, we get $|\Delta\rho_{\text{diff}}| \approx |\Delta m_A| + (|\Delta m_B| + |m_B\,\Delta(n_B - n_A)|)\, e^{n_B - n_A}$, therefore

$$|\Delta\rho_{\text{diff}}| \approx \left(|m_A| + |m_B| \cdot (1 + |n_B| + |n_A|)\; e^{n_B - n_A}\right) \varepsilon_{\text{machine}}\,,$$

and we can drop the absolute values around m_A, m_B and ρ_{diff} (which are nonnegative) to get the approximation

$$|\Delta I_{\text{diff}}| \approx |\widehat{\Delta} I_{\text{diff}}| := \left(m_A + m_B \cdot (1 + |n_B| + |n_A|)\, e^{n_B - n_A} + \rho_{\text{diff}}\,|\sigma_{\text{diff}}|\right) \varepsilon_{\text{machine}}\, e^{\sigma_{\text{diff}}}\,.$$

We will use $|\widehat{\Delta} I_{\text{diff}}|$ as an estimate of the actual absolute error $|\Delta I_{\text{diff}}|$.

3.2 Computing $I_{x,y}^{\mu,p}$ using a trapezoidal rule

A simple first order trapezoidal approximation of $I_{x,y}^{\mu,p}$ yields

$$I_{x,y}^{\mu,p} \approx I_{\text{trapezoid}} := (y - x)\,\frac{f_{\mu,p}(x) + f_{\mu,p}(y)}{2}\,, \tag{26}$$

where $f_{\mu,p}(s) = s^{p-1}\, e^{-\mu s}$. For the practical implementation, we will compute $I_{\text{trapezoid}}$ using the mantissa-exponent representation $I_{\text{trapezoid}} = \rho_{\text{trapezoid}} \cdot e^{\sigma_{\text{trapezoid}}}$, where

$$\sigma_{\text{trapezoid}} = \max(n_x, n_y), \quad \rho_{\text{trapezoid}} = \frac{y - x}{2\,x} e^{n_x - \sigma_{\text{trapezoid}}} + \frac{y - x}{2\,y} e^{n_y - \sigma_{\text{trapezoid}}}\,,$$

noting $n_x = -\mu x + p \log x$ and $n_y = -\mu y + p \log y$. The following proposition gives an upper bound of the absolute error $|\Delta I_{\text{trapezoid}}| = |I_{x,y}^{\mu,p} - I_{\text{trapezoid}}|$ associated to the approximation of $I_{x,y}^{\mu,p}$ by $I_{\text{trapezoid}}$. Remark that this upper bound is not interesting for all values of μ, x, y, p, but it gets precise as the distance betwen x and y gets small.

Proposition 1. *For any $\mu \in \mathbb{R} \setminus \{0\}$, for any positive integer p, and any nonnegative real numbers x, y, such as $x \leq y$, we have the upper bound*

$$|\Delta I_{trapezoid}| \leq |\widehat{\Delta} I_{trapezoid}| := \frac{(y-x)^3}{12} D_{x,y}^{\mu,p}\, y^{\max(0,p-3)}\, e^{\max(-\mu x,-\mu y)}\,,$$

where

$$D_{x,y}^{\mu,p} = \begin{cases} \mu^2 & \text{if } p = 1\,, \\ \max\left(|\mu^2 x - 2\mu|, |\mu^2 y - 2\mu|\right) & \text{if } p = 2\,, \\ C_{x,y}^{\mu,p} & \text{if } p \geq 3\,, \end{cases}$$

and

$$C_{x,y}^{\mu,p} = \begin{cases} |P_\mu(y)| & \text{if } \mu < 0\,, \\ \max\left(|P_\mu(x)|, p-1, |P_\mu(y)|\right) & \text{if } \mu > 0 \text{ and } x \leq \frac{p-1}{\mu} \leq y\,, \\ \max\left(|P_\mu(x)|, |P_\mu(y)|\right) & \text{otherwise,} \end{cases}$$

with $P_\mu(s) = (\mu s)^2 - 2(p-1)\mu s + (p-1)(p-2)$.

Proof (abridged). The first order trapezoidal rule yields the upper bound

$$|\Delta I_{\text{trapezoid}}| \leq \frac{(y-x)^3}{12} \sup_{s\in[x,y]} |f''_{\mu,p}(s)|\,. \tag{27}$$

In the case $p \geq 3$, for any $s \in [x,y]$, we have $f''_{\mu,p}(s) = P_\mu(s)\, s^{p-3}\, e^{-\mu s}$, and a straightforward study of the second degree polynomial P_μ yields $C_{x,y}^{\mu,p} = \sup_{s\in[x,y]} |P_\mu(s)|$. It follows that

$$\sup_{s\in[x,y]} |f''_{\mu}(s)| \leq C_{x,y}^{\mu,p}\, y^{p-3}\, e^{\max(-\mu x,-\mu y)}\,.$$

In the cases $p = 1$ and $p = 2$, a similar study can be led without difficulty. Finally, for any $p \geq 1$, we get

$$\sup_{s\in[x,y]} |f''_{\mu}(s)| \leq D_{x,y}^{\mu,p}\, y^{p-3}\, e^{\max(-\mu x,-\mu y)}\,,$$

which, combined to (27), yields the announced result. □

3.3 Criterion for the selection of the approximation by trapezoidal rule or differences

In order to choose between the two approximation methods (trapezoidal or difference), we propose to select the one yielding the smallest absolute error. To this aim, we consider the ratio between $|\widehat{\Delta} I_{\text{diff}}|$ and $|\widehat{\Delta} I_{\text{trapezoid}}|$, i.e.,

$$\forall x \neq y, \quad R_{x,y}^{\mu,p} = \frac{|\widehat{\Delta} I_{\text{diff}}|}{|\widehat{\Delta} I_{\text{trapezoid}}|}\,,$$

which is an approximation of the ratio between the effective relative errors ΔI_{diff} and $\Delta I_{\text{trapezoid}}$. Then, we will approximate $I_{x,y}^{\mu,p}$ by $I_{\text{trapezoid}}$ when $R_{x,y}^{\mu,p} > 1$, or by I_{diff} otherwise. Notice that for the practical evaluation of $R_{x,y}^{\mu,p}$, we will use again a mantissa-exponent representation $R_{x,y}^{\mu,p} = \rho_r \cdot e^{\sigma_r}$, where

$$\rho_r = \frac{12 \cdot (m_A + m_B \cdot (1 + |n_B| + |n_A|)\, e^{n_B - n_A} + \rho_{\text{diff}}\, |\sigma_{\text{diff}}|)\, \varepsilon_{\text{machine}}}{D_{x,y}^{\mu,p}}, \quad \sigma_r = \sigma_{\text{diff}} - \sigma_t\,,$$

noting (m_A, n_A) and (m_B, n_B) the mantissa-exponent representations of the quantities A and B, returned by Algorithms 3 and 4, noting $\rho_{\text{diff}} = m_A - m_B \cdot e^{nB-nA}$, $\sigma_{\text{diff}} = n_A$, and $\sigma_t = 3\log(y-x) + \max(0, p-3) \cdot \log y + \max(-\mu x, -\mu y)$. We validate the ability of this criterion to automatically select the most accurate approximation in Figure 5.

Computation of $I_{x,y}^{\mu,p}$:	$\mu < 0$	$\mu > 0$
$p < p_{\lim}(\mu x)$	$\gamma_\mu^{\text{ibp}}(p,y) - \gamma_\mu^{\text{ibp}}(p,x)$	$\Gamma_\mu^{\text{cfrac}}(p,x) - \Gamma_\mu^{\text{cfrac}}(p,y)$
$p_{\lim}(\mu x) \leq p < p_{\lim}(\mu y)$	$\gamma_\mu^{\text{ibp}}(p,y) - \gamma_\mu^{\text{cfrac}}(p,x)$	$\frac{\Gamma(p)}{\mu^p} - \left(\gamma_\mu^{\text{cfrac}}(p,x) + \Gamma_\mu^{\text{cfrac}}(p,y)\right)$
$p_{\lim}(\mu y) \leq p$	$\gamma_\mu^{\text{cfrac}}(p,y) - \gamma_\mu^{\text{cfrac}}(p,x)$	$\gamma_\mu^{\text{cfrac}}(p,y) - \gamma_\mu^{\text{cfrac}}(p,x)$

Table 1: Computing $I_{x,y}^{\mu,p}$ as a difference of generalized incomplete gamma functions. We propose here a practical computational method for the evaluation of $I_{x,y}^{\mu,p}$ by means of a difference of type $I_{x,y}^{\mu,p} = A - B$, where $A = \gamma_\mu(p,y)$, $B = \gamma_\mu(p,x)$, or, when $\mu > 0$, $A = \Gamma_\mu(p,x)$, $B = \Gamma_\mu(p,y)$, or $A = \Gamma(p)/\mu^p$, $B = \gamma_\mu(p,x) + \Gamma_\mu(p,y)$. Thanks to Figure 4, we derive which difference $A - B$, and which numerical method must be used for the efficient evaluation of A and B, according to the value of x, y, μ, p. It is important to notice that the evaluation of $I_{x,y}^{\mu,p}$ by means of difference $A - B$ is inaccurate when $A \approx B$, which happens when $x \approx y$. In that case, the integral $I_{x,y}^{\mu,p}$ must be approximated differently, as discussed in section 3.2.

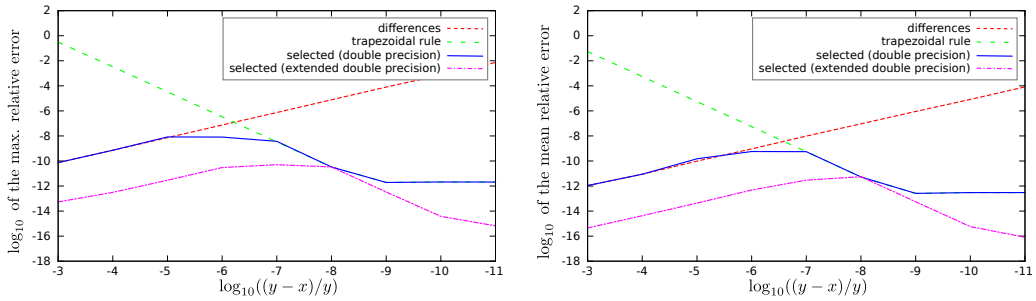


Figure 5: Control of maximum and mean relative errors associated to the computation of $I_{x,y}^{\mu,p}$ using differences or a first order trapezoidal rule. In Section 3.2, we proposed to approximate the integral $I_{x,y}^{\mu,p}$ by a difference of generalized incomplete gamma functions, $I_{x,y}^{\mu,p} \approx I_{\text{diff}} = A - B$ (see Table 1 to derive the values of A and B according to x, y, μ, p), or using a trapezoidal rule, $I_{x,y}^{\mu,p} \approx I_{\text{trapezoid}}$. We proposed in Section 3.3 an explicit criterion, based on the computation of a ratio of (some estimates of) the absolute errors $|\Delta I_{\text{diff}}|$ and $|\Delta I_{\text{trapezoid}}|$ associated to those two approximations, which can be used to automatically select which approximation should be used. For several values of $\delta_r = (y-x)/y$, we computed $I_{x,y}^{\mu,p}$ for a large range of parameters ($\mu = \pm 1$, p integer in $[1, 1000]$, y integer in $[1, 1000]$, and x being the floating-point number closest to $y\,(1 - \delta_r)$). We display here the evolution, as a function of $\log_{10}(\delta_r)$, of the maximal (left-side) and mean (right-side) relative error observed when using the approximation by differences I_{diff} (dashed red curve, standard double precision implementation), or when using the approximation by trapezoidal rule $I_{\text{trapezoid}}$ (dashed green curve, standard double precision implementation), or when automatically selecting the computation method (plain blue curve for the standard double precision implementation, dotted purple curve for the extended double precision implementation), thanks to the criterion proposed in Section 3.3. We see that the plain blue curve lies almost everywhere below the dashed curves, showing that the criterion efficiently selects the best accurate approximation. We can see also that the error can be improved by three orders of magnitude using extended double precision, except when $\delta_r \approx 10^{-8}$ (in that case, the precision is limited by the fact that we only use one term in the trapezoidal rule).

4 Discussion on the evaluation of the complete gamma function

The computation of the complete gamma function $\Gamma(p)$ for $p \in \mathbb{N}$, $\mathbb{R}$ or $\mathbb{C}$ constitutes itself a wide subject of research. The object of this section is to compare several methods from the litterature and end up with a practical efficient algorithm for computing the quantity

$\Gamma(p)$, which is needed to compute $I_{x,y}^{\mu,p}$ as right-hand difference (9), i.e.

$$I_{x,y}^{\mu,p} = \frac{\Gamma(p)}{\mu^p} - \gamma_\mu(p,x) - \Gamma_\mu(p,y)\,.$$

Since $\Gamma(p)$ gets huge as p increases, in practice we approximate its logarithm $\log\Gamma(p)$. Note that when p is a positive integer, as it is the case in this work, we have $\Gamma(p) = (p-1)!$ so that $\log\Gamma(p)$ can be easily computed using

$$\log(p-1)! = \sum_{k=1}^{p-1} \log k\,.$$

However, the numerical computation of this sum becomes rapidely inaccurate when p is large, because of the cumulation of small numerical errors made at each step of the summation. Besides, in order to facilitate the adaptation of the present work to noninteger values of p, we prefer to focus on more general methods.

The first evaluation method that we will consider was proposed in [Lanczos, 1964], and uses a Stirling formula-like approximation:

$$\forall p > 1, \quad \Gamma(p) = \sqrt{2\pi}\left(p+\gamma-\tfrac{1}{2}\right)^{p-\frac{1}{2}} e^{-(p+\gamma-\frac{1}{2})}\left(A_\gamma(p-1)+\varepsilon_\gamma\right), \tag{28}$$

where $\gamma > 0$ is a numerical parameter (different from the Euler-Mascheroni constant), $A_\gamma(p-1)$ is a truncated rational fraction of type $A_\gamma(p-1) = c_0(\gamma) + \sum_{k=1}^{N_\gamma} \frac{c_k(\gamma)}{p-1+k}$, and N_γ and the coefficients $\{c_k(\gamma)\}_{0\le k\le N_\gamma}$ depend on the value of γ. In the case $\gamma = 5$, Lanczos claims that the relative error $|\varepsilon_5|$ associated to (28) satisfies $|\varepsilon_5| < 2\cdot 10^{-10}$, and this claim was confirmed by our numerical experiments. In the case $\gamma = 5$, we have $N_\gamma = 6$ and the numerical values of the coefficients $\{c_k(\gamma)\}_{0\le k\le N_\gamma}$ are available in [Lanczos, 1964]. These values are refined to double floating-point precision in [Press et al., 1992], so we used them in our implementation of (28).

A more recent computation method (see [Char, 1980, Olver et al., 2010, Cuyt et al., 2008] and references therein), also based on a Stirling approximation, consists in computing

$$\forall p > 1, \quad \Gamma(p) = \sqrt{2\pi}\, e^{-p}\, p^{p-\frac{1}{2}}\, e^{J(p)}, \quad \text{where} \quad J(p) = \frac{a_0}{p+}\,\frac{a_1}{p+}\,\frac{a_2}{p+}\,\cdots, \tag{29}$$

where some numerical approximations, with 40 decimal digits of precision, of the coefficients $\{a_k\}_{0\le k\le 40}$ of the continued fraction $J(p)$ can be found in [Char, 1980].

The last approximation that we present, and that we will select in practice as the most simple an accurate method, is a refinement of the Lanczos formula, proposed in [Pugh, 2004]. In his work, Pugh adapted (28) into

$$\forall p > 1, \quad \Gamma(p) \approx 2\sqrt{\frac{e}{\pi}}\left(\frac{p+r-\frac{1}{2}}{e}\right)^{p-\frac{1}{2}}\left[d_0 + \sum_{k=1}^{N_r}\frac{d_k}{p-1+k}\right], \tag{30}$$

where r is again a numerical parameter (which replaces the parameter γ of (28), to avoid confusion with the Euler-Mascheroni constant). Pugh studied the accuracy of the approximation (30) for differents settings r. In the case $r = 10.900511$, he sets $N_r = 11$, and gives the numerical values of the coefficients $\{d_k\}_{0\le k\le 10}$ with 20 significant decimal digits (see Table 2). According to Pugh, this setting yields a relative error less than 10^{-19},

k	d_{3k}	d_{3k+1}	d_{3k+2}
0	2.48574089138753565546E-5	1.05142378581721974210E+0	-3.45687097222016235469E+0
1	4.51227709466894823700E+0	-2.98285225323576655721E+0	1.05639711577126713077E+0
2	-1.95428773191645869583E-1	1.70970543404441224307E-2	-5.71926117404305781283E-4
3	4.63399473359905636708E-6	-2.71994908488607703910E-9	

Table 2: coefficients $\{d_k\}_{0\leq k\leq 10}$ of Equation (30) with 20 significant decimal digits [Pugh, 2004].

Algorithm 5: Accurate computation of $\log\Gamma(p)$ using Pugh's method .

Input: A real number $p \geq 1$.

Output: An accurate estimation of $\log\Gamma(p)$.

Require: Coefficients $\{d_k\}_{0\leq k\leq 10}$ defined in Table 2.

return $\log\left(2\sqrt{\frac{e}{\pi}}\left[d_0 + \sum_{k=0}^{10} \frac{d_k}{p-1+k}\right]\right) - \left(p-\frac{1}{2}\right) + \left(p-\frac{1}{2}\right)\log\left(p+10.900511-\frac{1}{2}\right)$

which is effectively what we observed when computing (30) with Maple for $1 \leq p \leq 10^4$ in multiprecision (30 digits).

In order to select which method will be used in our algorithms, we used the three approximations (28), (29), and (30) to compute $\log\Gamma(p)$ for $1 \leq p \leq 10^4$. The values of $\Gamma(p) = e^{\log\Gamma(p)}$ were converted into scientific notation using (22), in order to avoid the overflow issues that fatally occur when computing $\Gamma(p)$ directly. The accuracy was evaluated using Maple, the results (restricted to $1 \leq p \leq 5000$) are displayed in Figure 6. It follows from our experiments that the approximations (29) and (30) are the most accurate, except for small values of p where the continued fraction is inacurrate (certainly because more than 40 approximants are needed for those values of p). Besides, both methods suffer from the loss of accuracy involved by the conversion of $\Gamma(p)$ into scientific notation from the value of its logarithm $\log\Gamma(p)$, as discussed in Section 2.3. Again, this loss of precision can be compensated by improving the floating-point accuracy, as illustrated in Figure 6. The Lanczos method yields a relative error close to $2\cdot 10^{-10}$, and in this case, the loss of accuracy is due to the approximation itself. Finally we decided to use Pugh's method, for its simplicity and the nice theoretical study provided in [Pugh, 2004]. Our implementation is described in Algorithm 5.

5 Comparison with algorithm 435

In this section, we compare our algorithm with Algorithm 435, proposed in [Fullerton, 1972], for the evaluation of the generalized incomplete gamma function $I_{x,y}^{\mu,p}$. Remind that, in his work, Fullerton focused on a slightly different integral than us, since he proposed an algorithm for the evaluation of the integral $J_{x,y}^{p}$ defined in (6), however the computation of $I_{x,y}^{\mu,p}$ using $J_{x,y}^{p}$, or conversely of $J_{x,y}^{p}$ using $I_{x,y}^{\mu,p}$, is immediate, as we already explained in Section 1. Similarly, the function

$$\gamma'(p,x) = \int_0^x |s|^{p-1}\, e^{-s}\, ds, \quad -\infty < x \leq +\infty\,,$$

he introduced is also closely related to our lower generalized incomplete gamma function γ_μ, since for any $x \geq 0$, and any $p > 0$, we have $\gamma_\mu(p,x) = \mu^{-p}\,\gamma'(p,x)$ when $\mu > 0$, and $\gamma_\mu(p,x) = -|\mu|^{-p}\,\gamma'(p,x)$ when $\mu < 0$. Fullerton poposed an algorithm for the numerical

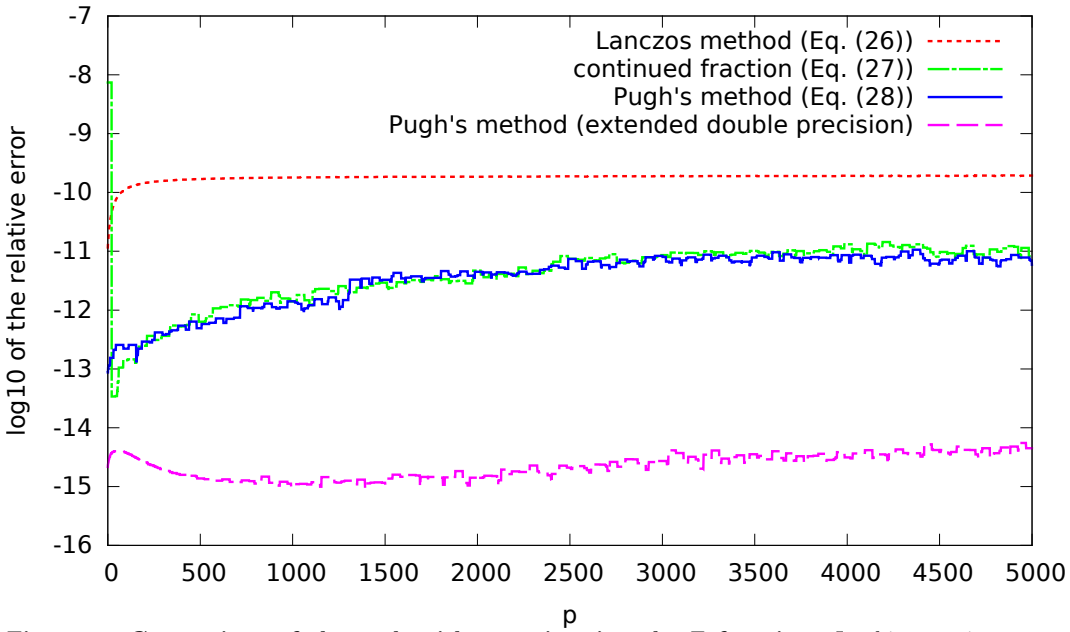


Figure 6: Comparison of three algorithms estimating the Γ function. In this experiment, we used (28), (29) and (30) to compute $\log\Gamma(p)$ for $1 \leq p \leq 5000$, and estimated the relative error using Maple. We here display the graphs of the relative errors related to each approximation, as a function of the parameter p (the curves were smoothed by replacing the relative error associated to $\Gamma(p)$ by its maximum value over the range $[p-20, p+20]$, in order to improve the readability). The three top curves were obtained using a C-implementation in standard double precision of the three approximation methods. We observe that the continued fraction approximation (29) is inacurrate for small values of p. However, we see that the precision reached by the Pugh's approximation (30) (and also the continued fraction, excepting for small values of p) is only limited by the loss of precision occuring when formatting $\Gamma(p)$ in scientific notation from its mantissa-exponent representation $\Gamma(p) = \rho \cdot e^{\sigma}$ (where $\rho = 1$ and $\sigma = \log\Gamma(p)$), since the bound predicted in (23) is attained (for instance, we check that for $p = 800$ (respectively $p = 5000$), we have $\sigma = \log\Gamma(p) \approx 4545$ (respectively $\sigma = 37582$), so that the optimal relative error predicted in (23) is close to 10^{-12} (respectively 10^{-11}), which is more or less the value attained by the two algorithms). This is not the case for the Lanczos approximation (28), whose precision is limited by the approximation itself. The last curve (bottom) corresponds to the implementation of Pugh's method in C langage using extended double precision (long double datatype); the loss of precision resulting from the formatting of $\Gamma(p)$ in scientific notation is significantly reduced.

evaluation of γ', then suggested to evaluate the integral $J^p_{x,y}$ using the difference

$$J^p_{x,y} = e^x \int_x^y |s|^{p-1}\, e^{-s}\, ds = e^x\, (\gamma'(a,y) - \gamma'(a,x))$$

when $1 \leq p \leq 2$, and using a forward (when $p > 2$) or backward (when $p < 1$) recurrence relation, leading back to the computation of a quantity $J^q_{x,y}$, with $1 \leq q \leq 2$. In the case $1 \leq p \leq 2$, the evaluation of $\gamma'(p,x)$ relies on different approximation methods (such as continued fractions, approximation using Chebyshev polynomials, or asymptotic expansions), according to the value of x.

As already pointed in [Schoene, 1978], Algorithm 435 suffers from several numerical instabilities, arising when $p > 2$. We indeed observed in our own numerical experiments, presented in Tables 3, 4 and 5, some computed values with very low accuracy, or incorrect sign, typically when $p \geq 10$, or when $x \leq p \leq y$. We also also observed some overflow issues, for instance when working with $p \geq 100$.

In the experiments of Tables 3-5, we evaluated the integral $I_{x,y}^{\mu,p}$, for several sets of parameters x, y, μ, p, using both Fullerton's Algorithm 435 and Algorithm 6. The accuracy of the returned result was controlled using the softwares Maple$^{\text{TM}}$ (using the instruction "`evalf(Int(s^(p-1)*exp(-mu*s),s=x..y,digits=30));`" to approximate the integral with 30 digits of precision), and Mathematica$^{\text{TM}}$ in [Wolfram Research Inc, 1998] for the online evaluation of $I_{x,y}^{\mu,p}$.

6 Conclusion and perspectives

In this work, we proposed an algorithm for the accurate evaluation of the generalized incomplete gamma function $I_{x,y}^{\mu,p}$. According to our experiments, the implementation of this algorithm with a standard double floating-point precision yields a relative error less than 10^{-10} (in the worst case scenario), and in general less than 10^{-13} for a large range of parameters, which constitutes a drastic gain of accuracy in comparison to that obtained using the algorithm proposed in [Fullerton, 1972]. Besides, our algorithm delivers the estimated value of the integral $I_{x,y}^{\mu,p}$ under a mantissa-exponent representation $I_{x,y}^{\mu,p} = \rho \cdot e^{\sigma}$, which greatly extends the range over which it can be computed (which proved useful in [Abergel et al., 2015], where the computation of sums and ratios of generalized incomplete gamma functions $I_{x,y}^{\mu,p}$ was required).

Note also that the general accuracy of the algorithm we propose could certainly be improved further using a floating-point arithmetic with more digits and a higher order generalization of the trapezoidal rule that we use for nearly identical integral bounds. Another interesting perspective would be to extend our approach to the computation of complex values for the integral $I_{x,y}^{\mu,p}$ (for instance, when p is noninteger and $\mu < 0$, or when x or y takes a nonreal value). The continued fractions we used remain valid for complex values of x and noninteger values of p (except when $I_{x,y}^{\mu,p}$ is indefinite), but as the recursive integration by part (14) cannot be used any more when p is noninteger and $\mu < 0$, another strategy would be needed to cover the corresponding parameters region.

Algorithm 6: Accurate computation of $I_{x,y}^{\mu,p} = \int_x^y s^{p-1}\, e^{-\mu s}\, ds$.

Input: Three numbers $\mu \in \mathbb{R} \setminus \{0\}$, $x \in \mathbb{R}_+$, $y \in \mathbb{R}_+ \cup \{+\infty\}$ such as $y \geq x$, and a positive integer $p \geq 1$. Notice that the value $y = +\infty$ is allowed only when $\mu > 0$.

Output: Two numbers $\rho \in \mathbb{R}$ and $\sigma \in \mathbb{R} \cup \{-\infty\}$ such as $I_{x,y}^{\mu,p} \approx \rho \times e^{\sigma}$.

Requirements: Functions γ_μ^{ibp} (Algo. 3: select recursive integration by parts), $\gamma_\mu^{\text{cfrac}}$ (Algo. 3: select continued fraction), $\Gamma_\mu^{\text{cfrac}}$ (Algo. 4), and $\log\Gamma$ (Algo. 5).

Initialization: Define function $p_{\text{lim}} : \mathbb{R} \cup \{+\infty\} \to \mathbb{R} \cup \{+\infty\}$ by

$$\forall z \in \mathbb{R} \cup \{+\infty\}, \quad p_{\text{lim}}(z) = \begin{cases} 5\sqrt{|z|} - 5 & \text{if } z < -9\,, \\ 0 & \text{if } -9 \leq z \leq 0\,, \\ z & \text{otherwise.} \end{cases}$$

if $x = y$ *at machine precision* **then** $(\rho, \sigma) \leftarrow (0, -\infty)$
else
  `// Evaluate` (m_A, n_A), (m_B, n_B) `and` $(\rho_{\texttt{diff}}, \sigma_{\texttt{diff}})$, `some mantissa-exponent`
  `// representations of` A, B `and` $I_{\texttt{diff}}$, `such as` $A = m_A\, e^{n_A}$, $B = m_B\, e^{n_B}$,
  `//` $I_{\texttt{diff}} = \rho_{\texttt{diff}}\, e^{\sigma_{\texttt{diff}}} = A - B$, `and` $I_{x,y}^{\mu,p} \approx I_{\texttt{diff}}$.
  if $\mu < 0$ **then**
    if $p < p_{lim}(\mu y)$ **then** $(m_A, n_A) \leftarrow \gamma_\mu^{\text{ibp}}(p, y)$
    else $(m_A, n_A) \leftarrow \gamma_\mu^{\text{cfrac}}(p, y)$
    if $p < p_{lim}(\mu x)$ **then** $(m_B, n_B) \leftarrow \gamma_\mu^{\text{ibp}}(p, x)$
    else $(m_B, n_B) \leftarrow \gamma_\mu^{\text{cfrac}}(p, x)$
  else if $\mu > 0$ **then**
    if $p < p_{lim}(\mu x)$ **then**
      $(m_A, n_A) \leftarrow \Gamma_\mu^{\text{cfrac}}(p, x)$
      $(m_B, n_B) \leftarrow \Gamma_\mu^{\text{cfrac}}(p, y)$
    else if $p_{lim}(\mu x) \leq p < p_{lim}(\mu y)$ **then**
      $(m_A, n_A) \leftarrow (1, \log\Gamma(p) - p\,\log\mu)$
      $(m_x, n_x) \leftarrow \gamma_\mu^{\text{cfrac}}(p, x)$
      $(m_y, n_y) \leftarrow \Gamma_\mu^{\text{cfrac}}(p, y)$
      $n_B \leftarrow \max(n_x, n_y)$
      if $n_B = -\infty$ **then** $n_B \leftarrow 0$ `// may happen when` $x = 0$ `and` $y = +\infty$
      $m_B \leftarrow m_x\, e^{n_x - n_B} + m_y\, e^{n_y - n_B}$
    else
      $(m_A, n_A) \leftarrow \gamma_\mu^{\text{cfrac}}(p, y)$
      $(m_B, n_B) \leftarrow \gamma_\mu^{\text{cfrac}}(p, x)$
  $(\rho_{\text{diff}}, \sigma_{\text{diff}}) \leftarrow \left(m_A - m_B \cdot e^{n_B - n_A}, n_A\right)$
  `// Compute the ratio` $R_{x,y}^{\mu,p} = \widehat{\Delta} I_{\texttt{diff}} / \widehat{\Delta} I_{\texttt{trapezoid}}$ `with a mantissa-exponent`
  `// representation` (ρ_r, σ_r), `as described in Section 3.3.`
  $D \leftarrow D_{x,y}^{\mu,p}$ `// the explicit expression of` $D_{x,y}^{\mu,p}$ `is given in Proposition 1`
  $\rho_r \leftarrow 12 \cdot \frac{m_A + m_B \cdot (1 + |n_B| + |n_A|) e^{n_B - n_A + \rho_{\text{diff}} |\sigma_{\text{diff}}|}}{D} \cdot \varepsilon_{\text{machine}}$
  $\sigma_r \leftarrow \sigma_{\text{diff}} - 3\,\log(y - x) - \max(0, p - 3)\log y - \max(-\mu x, -\mu y)$
  if $m_R\, e^{n_R} > 1$ **then**
    $n_x \leftarrow -\mu x + p \log x$
    $n_y \leftarrow -\mu y + p \log y$
    $\sigma \leftarrow \max(n_x, n_y)$
    $\rho \leftarrow \frac{y-x}{2x}\, e^{n_x - \sigma} + \frac{y-x}{2y}\, e^{n_y - \sigma}$
  else $(\rho, \sigma) \leftarrow (\rho_{\text{diff}}, \sigma_{\text{diff}})$

return (ρ, σ)

Parameters setting ($\mu = 1$)	Algorithm 435 in [Fullerton, 1972]	Relative error	Algorithm 6	Relative error
$\mu = 1, x = 9, y = 11, p = 1$	$1.067081029759719 \cdot 10^{-4}$	$3 \cdot 10^{-9}$	$1.0670810329643395 \cdot 10^{-4}$	$6 \cdot 10^{-16}$
$\mu = 1, x = 9, y = 11, p = 5$	$9.567113518714904 \cdot 10^{-1}$	$1 \cdot 10^{-4}$	$9.5661698023023700 \cdot 10^{-1}$	$1 \cdot 10^{-15}$
$\mu = 1, x = 9, y = 11, p = 10$	$1.085447578125000 \cdot 10^{5}$	$2 \cdot 10^{-1}$	$8.9594201765236983 \cdot 10^{4}$	$1 \cdot 10^{-14}$
$\mu = 1, x = 9, y = 11, p = 12$	$1.632943040000000 \cdot 10^{8}$	17	$8.9310494815538749 \cdot 10^{6}$	$3 \cdot 10^{-15}$
$\mu = 1, x = 9, y = 11, p = 14$	$-2.977905664000000 \cdot 10^{10}$	34	$9.0203414117080807 \cdot 10^{8}$	$2 \cdot 10^{-15}$
$\mu = 1, x = 9, y = 11, p = 100$	$-\mathrm{NaN}$	N/A	$2.5825265278752760 \cdot 10^{97}$	$2 \cdot 10^{-14}$
$\mu = 1, x = 9, y = 11, p = 300$	$-\mathrm{NaN}$	N/A	$1.5122076179085018 \cdot 10^{305}$	$3 \cdot 10^{-14}$
$\mu = 1, x = 9, y = 11, p = 1000$	$-\mathrm{NaN}$	N/A	$4.1710431880333560 \cdot 10^{1033}$	$3 \cdot 10^{-13}$
$\mu = 1, x = 100, y = 120, p = 1$	$3.783505853677006 \cdot 10^{-44}$	$2 \cdot 10^{-2}$	$3.7200759683531697 \cdot 10^{-44}$	$5 \cdot 10^{-15}$
$\mu = 1, x = 100, y = 120, p = 5$	$3.873433252162870 \cdot 10^{-36}$	$3 \cdot 10^{-9}$	$3.8734332644314730 \cdot 10^{-36}$	$4 \cdot 10^{-15}$
$\mu = 1, x = 100, y = 120, p = 10$	$4.083660502797843 \cdot 10^{-26}$	$2 \cdot 10^{-8}$	$4.0836605881700520 \cdot 10^{-26}$	$8 \cdot 10^{-15}$
$\mu = 1, x = 100, y = 120, p = 20$	$4.579807864502072 \cdot 10^{-6}$	$9 \cdot 10^{-8}$	$4.5798082802928473 \cdot 10^{-6}$	$2 \cdot 10^{-14}$
$\mu = 1, x = 100, y = 120, p = 21$	$+\infty$	N/A	$4.6360373381202165 \cdot 10^{-4}$	$1 \cdot 10^{-14}$
$\mu = 1, x = 100, y = 120, p = 100$	$-\mathrm{NaN}$	N/A	$4.2821563816534019 \cdot 10^{155}$	$1 \cdot 10^{-13}$
$\mu = 1, x = 100, y = 120, p = 170$	$-\mathrm{NaN}$	N/A	$4.2461593130874860 \cdot 10^{299}$	$3 \cdot 10^{-14}$
$\mu = 1, x = 100, y = 120, p = 1000$	$-\mathrm{NaN}$	N/A	$1.3223863318125477 \cdot 10^{2024}$	$1 \cdot 10^{-12}$

Table 3: Comparison between Algorithm 435 proposed in [Fullerton, 1972] and Algorithm 6, for the computation of $I_{x,y}^{\mu,p}$ with $\mu = 1$. In this series of experiments, we focus on the case $\mu = 1$. We tested, for $(x, y) = (9, 11)$, and $(x, y) = (100, 120)$, different integer values of p between 1 and 1000. In the second column, we display the values of $I_{x,y}^{\mu,p}$ returned by Fullerton's Algorithm (that we slightly adapted to compute $I_{x,y}^{\mu,p}$ instead of $J_{x,y}^{\mu,p}$). The corresponding relative errors, evaluated using Mathematica or Maple softwares (both softwares yield the same relative error), are displayed on the third and fourth columns. We see that some numerical instabilities arise when $x \leq p$, and we observe some overflow issues, as p gets high. Some inaccurate results are also observed for low values of p, when $x = 100, y = 120, p = 1$ (but also for many other values of (x, y, p), not represented here). Note also the settings $x = 100, y = 120, p \in \{20, 21\}$, for which the value returned by the algorithm shifts from 10^{-6} to $+\infty$. In the fourth column, we display the values returned by Algorithm 6 (using a C implementation with standard double precision), followed by the corresponding relative errors. The relative errors reached by Algorithm 6 are nearly optimal, since they are mostly due to the loss of precision involved by the mantissa-exponent representation (see the optimality bounds predicted in (21) and (23)), showing that both mantissa and exponents are in practice computed with a relative precision close to the machine precision.

Parameters setting ($\mu = -1$)	Algorithm 435 in [Fullerton, 1972]	Relative error	Algorithm 6	Relative error
$\mu = -1, x = 5, y = 10, p = 1$	$2.187916015625000 \cdot 10^{4}$	$5 \cdot 10^{-5}$	$2.1878052635704163 \cdot 10^{4}$	$1 \cdot 10^{-15}$
$\mu = -1, x = 5, y = 10, p = 3$	$1.803647750000000 \cdot 10^{6}$	$3 \cdot 10^{-7}$	$1.8036471714694066 \cdot 10^{6}$	$1 \cdot 10^{-16}$
$\mu = -1, x = 5, y = 10, p = 10$	$1.129511596851200 \cdot 10^{13}$	$4 \cdot 10^{-8}$	$1.1295115549498505 \cdot 10^{13}$	$4 \cdot 10^{-15}$
$\mu = -1, x = 5, y = 10, p = 60$	$+\infty$	N/A	$3.1530071119035434 \cdot 10^{62}$	$2 \cdot 10^{-14}$
$\mu = -1, x = 5, y = 10, p = 100$	$-\text{NaN}$	N/A	$2.0040499509396790 \cdot 10^{102}$	$1 \cdot 10^{-14}$
$\mu = -1, x = 5, y = 10, p = 300$	$-\text{NaN}$	N/A	$7.1060487642415961 \cdot 10^{301}$	$9 \cdot 10^{-14}$
$\mu = -1, x = 5, y = 10, p = 1000$	$-\text{NaN}$	N/A	$2.1808595556561760 \cdot 10^{1001}$	$3 \cdot 10^{-13}$
$\mu = -1, x = 20, y = 25, p = 1$	$7.151973171200000 \cdot 10^{10}$	$3 \cdot 10^{-8}$	$7.1519734141975967 \cdot 10^{10}$	$2 \cdot 10^{-15}$
$\mu = -1, x = 20, y = 25, p = 10$	$2.068890077987267 \cdot 10^{23}$	$3 \cdot 10^{-2}$	$2.0016822370845540 \cdot 10^{23}$	$9 \cdot 10^{-16}$
$\mu = -1, x = 20, y = 25, p = 20$	$1.821993954177914 \cdot 10^{37}$	$2 \cdot 10^{-1}$	$1.4733948083664500 \cdot 10^{37}$	$1 \cdot 10^{-15}$
$\mu = -1, x = 20, y = 25, p = 30$	$+\infty$	N/A	$1.1449672725827719 \cdot 10^{51}$	$3 \cdot 10^{-15}$
$\mu = -1, x = 20, y = 25, p = 210$	$-\text{NaN}$	N/A	$1.1321187815658028 \cdot 10^{302}$	$2 \cdot 10^{-14}$
$\mu = -1, x = 20, y = 25, p = 1000$	$-\text{NaN}$	N/A	$6.1186720860190186 \cdot 10^{1405}$	$2 \cdot 10^{-13}$

Table 4: Same as Table 3, but for $\mu = -1$. We performed here a similar experiment as in Table 3, but in the case $\mu < 0$. We tested, for $(x, y) = (5, 10)$ and $(x, y) = (20, 25)$, different values of p between 1 and 1000. As we can see, Fullerton's algorithm is unbable to provide accurate estimates as p increases. In contrast, the relative errors reached by Algorithm 6 remain nearly optimal, mostly limited by the mantissa-exponent representation according to the optimality bounds derived in (21) and (23).

Parameters setting	Algorithm 435 in [Fullerton, 1972]	Relative error	Algorithm 6	Selected approx.	Relative error
$\mu = 1, x = d(5 - 10^{0}), y = 5, p = 10$	$8.598737304687500 \cdot 10^{3}$	$2 \cdot 10^{-8}$	$8.5987371691242424 \cdot 10^{3}$	difference	$7 \cdot 10^{-17}$
$\mu = 1, x = d(5 - 10^{-1}), y = 5, p = 10$	$1.263989379882812 \cdot 10^{3}$	$8 \cdot 10^{-7}$	$1.2639903706449711 \cdot 10^{3}$	difference	$1 \cdot 10^{-15}$
$\mu = 1, x = d(5 - 10^{-3}), y = 5, p = 10$	$1.315382766723632 \cdot 10^{1}$	$7 \cdot 10^{-5}$	$1.3154789325748350 \cdot 10^{1}$	difference	$1 \cdot 10^{-13}$
$\mu = 1, x = d(5 - 10^{-4}), y = 5, p = 10$	$1.317400574684143 \cdot 10^{0}$	$1 \cdot 10^{-3}$	$1.3159526336877760 \cdot 10^{0}$	difference	$2 \cdot 10^{-11}$
$\mu = 1, x = d(5 - 10^{-5}), y = 5, p = 10$	$1.322335302829742 \cdot 10^{-1}$	$5 \cdot 10^{-3}$	$1.3160000091971793 \cdot 10^{-1}$	trap. rule	$2 \cdot 10^{-12}$
$\mu = 1, x = d(5 - 10^{-6}), y = 5, p = 10$	$1.179141830652952 \cdot 10^{-2}$	$1 \cdot 10^{-1}$	$1.3160047470408107 \cdot 10^{-2}$	trap. rule	$2 \cdot 10^{-14}$
$\mu = 1, x = d(5 - 10^{-7}), y = 5, p = 10$	0	1	$1.3160052243091611 \cdot 10^{-3}$	trap. rule	$4 \cdot 10^{-16}$
$\mu = 1, x = d(17 - 10^{0}), y = 17, p = 17$	$3.725839564800000 \cdot 10^{12}$	$8 \cdot 10^{-1}$	$2.0551230250736027 \cdot 10^{12}$	difference	$3 \cdot 10^{-14}$
$\mu = 1, x = d(17 - 10^{-1}), y = 17, p = 17$	$2.998156984320000 \cdot 10^{11}$	$5 \cdot 10^{-1}$	$2.0202925544709274 \cdot 10^{11}$	difference	$2 \cdot 10^{-13}$
$\mu = 1, x = d(17 - 10^{-3}), y = 17, p = 17$	$2.941651456000000 \cdot 10^{9}$	$5 \cdot 10^{-1}$	$2.0146022707357914 \cdot 10^{9}$	difference	$1 \cdot 10^{-11}$
$\mu = 1, x = d(17 - 10^{-4}), y = 17, p = 17$	$2.928078720000000 \cdot 10^{8}$	$5 \cdot 10^{-1}$	$2.0145489617316359 \cdot 10^{8}$	trap. rule	$4 \cdot 10^{-11}$
$\mu = 1, x = d(17 - 10^{-6}), y = 17, p = 17$	$6.554762500000000 \cdot 10^{6}$	$2 \cdot 10^{0}$	$2.0145430981932636 \cdot 10^{6}$	trap. rule	$2 \cdot 10^{-15}$
$\mu = 1, x = d(17 - 10^{-9}), y = 17, p = 17$	0	1	$2.0145432036144500 \cdot 10^{3}$	trap. rule	$2 \cdot 10^{-15}$
$\mu = -1, x = d(21 - 10^{0}), y = 21, p = 10$	$5.859836137154984 \cdot 10^{20}$	$5 \cdot 10^{-2}$	$5.5623377927217197 \cdot 10^{20}$	difference	$4 \cdot 10^{-15}$
$\mu = -1, x = d(21 - 10^{-1}), y = 21, p = 10$	$1.025911814748488 \cdot 10^{20}$	$5 \cdot 10^{-2}$	$9.7609411144076116 \cdot 10^{19}$	difference	$7 \cdot 10^{-15}$
$\mu = -1, x = d(21 - 10^{-3}), y = 21, p = 10$	$1.099215584270221 \cdot 10^{18}$	$5 \cdot 10^{-2}$	$1.0467611548908131 \cdot 10^{18}$	difference	$6 \cdot 10^{-12}$
$\mu = -1, x = d(21 - 10^{-5}), y = 21, p = 10$	$1.045801880623513 \cdot 10^{16}$	$2 \cdot 10^{-3}$	$1.0475015408230294 \cdot 10^{16}$	trap. rule	$2 \cdot 10^{-11}$
$\mu = -1, x = d(21 - 10^{-7}), y = 21, p = 10$	0	1	$1.0475089604363028 \cdot 10^{14}$	trap. rule	$2 \cdot 10^{-15}$
$\mu = -1, x = d(21 - 10^{-9}), y = 21, p = 10$	0	1	$1.0475091089401447 \cdot 10^{12}$	trap. rule	$3 \cdot 10^{-15}$

Table 5: Comparison between Fullerton's algorithm and Algorithm 6, for the computation of $I_{x,y}^{\mu,p}$ when $x \approx y$. In this last experiment, we compute $I_{x,y}^{\mu,p}$ in the case $x \approx y$ (the notation $d(s)$ used in the left column denotes the double-precision floating-point number that is closest to s). We see that the relative error reached by Algorithm 435 deteriorates as x and y get close to each other, and as already remarked before, Algorithm 435 is very inacurrate when $\mu x < p < \mu y$. In contrast, the relative errors observed with Algorithm 6 never exceed 10^{-10}, thanks to the first order estimate (trap. rule in column 5) that takes over to avoid cancellation errors when x and y are very close to each other.

Acknowledgments

The authors would like to thank the GDS Mathrice 2754 as well as MathStic and LAGA (Laboratoire Analyse, Géométrie et Applications) at Université Paris 13, for having kindly provided us an access to the computing server GAIA.

References

R. Abergel, C. Louchet, L. Moisan, and T. Zeng. Total variation restoration of images corrupted by poisson noise with iterated conditional expectations. In *Scale Space and Variational Methods in Computer Vision*, pages 178–190. Springer, 2015.

M. Abramowitz and I. A. Stegun. *Handbook of mathematical functions: with formulas, graphs, and mathematical tables*. Number 55. Courier Corporation, 1964.

G. P. Bhattacharjee. Algorithm as 32: The incomplete gamma integral. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 19(3):285–287, 1970.

M. Bleicher and P. Nicolini. Large extra dimensions and small black holes at the lhc. In *Journal of Physics: Conference Series*, volume 237, page 012008. IOP Publishing, 2010.

C. J. Cannon and I. M. Vardavas. The effect of redistribution on the emission peaks from chromospheric-type stellar atmospheres. *Astronomy and Astrophysics*, 32:85, 1974.

B. W. Char. On stieltjes's continued fraction for the gamma function. *Mathematics of Computation*, 34(150):547–551, 1980.

M. A. Chaudhry and S. M. Zubair. *On a class of incomplete gamma functions with applications*. CRC press, 2001.

A. Cuyt, F. Backeljauw, and C. Bonan-Hamada. *Handbook of continued fractions for special functions*. Springer Science & Business Media, 2008.

DLMF. NIST Digital Library of Mathematical Functions. http://dlmf.nist.gov/, Release 1.0.10, 2015. Online companion to Olver et al. [2010].

W. Fullerton. Algorithm 435: Modified incomplete gamma function [s14]. *Communications of the ACM*, 15(11):993–995, November 1972. ISSN 0001-0782. doi: 10.1145/355606.361891.

W. Gautschi. A computational procedure for incomplete gamma functions. *ACM Transactions On Mathematical Software*, 5(4):466–481, December 1979. ISSN 0098-3500.

W. Gautschi. The incomplete gamma functions since tricomi. In *Tricomi's Ideas and Contemporary Applied Mathematics, Atti dei Convegni Lincei, Accademia Nazionale dei Lincei*, volume 147, pages 203–237, 1998.

I. I. Guseinov and B. A. Mamedov. Evaluation of Incomplete Gamma Functions Using Downward Recursion and Analytical Relations. *Journal of Mathematical Chemistry*, 36 (4):341–346, August 2004.

J. G. Hills. Effect of binary stars on the dynamical evolution of stellar clusters. ii-analytic evolutionary models. *The Astronomical Journal*, 80:1075–1080, 1975.

W. B. Jones and W. J. Thron. *Continued fractions: analytic theory and applications*. Number 11 in Encyclopedia of mathematics and its applications. Addison-Wesley Pub. Co, Reading, Mass, 1980. ISBN 978-0-201-13510-7.

L. Kissel, R. H. Pratt, and S. C. Roy. Rayleigh scattering by neutral atoms, 100 ev to 10 mev. *Phys. Rev. A*, 22:1970–2004, Nov 1980. doi: 10.1103/PhysRevA.22.1970. URL http://link.aps.org/doi/10.1103/PhysRevA.22.1970.

C. Lanczos. A precision approximation of the gamma function. *Journal of the Society for Industrial and Applied Mathematics, Series B: Numerical Analysis*, 1(1):86–96, 1964.

W. J. Lentz. Generating bessel functions in mie scattering calculations using continued fractions. *Applied Optics*, 15(3):668–671, 1976.

V. Linetsky. Pricing equity derivatives subject to bankruptcy. *Mathematical finance*, 16(2): 255–282, 2006.

Y. Moreno, R. Pastor-Satorras, and A. Vespignani. Epidemic outbreaks in complex heterogeneous networks. *The European Physical Journal B-Condensed Matter and Complex Systems*, 26(4):521–529, 2002.

F. W. J. Olver, D. W. Lozier, R. F. Boisvert, and C. W. Clark, editors. *NIST Handbook of Mathematical Functions*. Cambridge University Press, New York, NY, 2010. Print companion to DLMF.

W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical recipes in C: the art of scientific computing*. Cambridge University Press, Cambridge ; New York, 2nd edition, 1992. ISBN 978-0-521-43108-8 978-0-521-43720-2.

G. R. Pugh. *An analysis of the Lanczos gamma approximation*. PhD thesis, University of British Columbia, 2004.

A. Robin, L. Moisan, and S. Le Hégarat-Mascle. An a-contrario approach for sub-pixel change detection in satellite imagery. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(11):1977–1993, 2010.

A. Y. Schoene. Remark on "algorithm 435: Modified incomplete gamma function [s14]". *ACM Trans. Math. Softw.*, 4(3):296–304, September 1978. ISSN 0098-3500. doi: 10.1145/355791.355803.

I. Thompson. Algorithm 926: Incomplete gamma functions with negative arguments. *ACM Transactions On Mathematical Software*, 39(2):14:1–14:9, February 2013. ISSN 0098-3500. doi: 10.1145/2427023.2427031.

I. J. Thompson and A. R. Barnett. Coulomb and bessel functions of complex arguments and order. *Journal of Computational Physics*, 64(2):490–509, 1986.

F. G. Tricomi. Sulla funzione gamma incompleta. *Annali di Matematica Pura ed Applicata*, 31(1):263–279, 1950.

S. Winitzki. Computing the incomplete gamma function to arbitrary precision. In *Proceedings of the 2003 International Conference on Computational Science and Its Applications: Part I*, ICCSA'03, pages 790–798, Berlin, Heidelberg, 2003. Springer-Verlag. ISBN 3-540-40155-5.

Wolfram Research Inc. Generalized incomplete gamma function, 1988. URL http://reference.wolfram.com/language/ref/Gamma.html. (documentation page).

Wolfram Research Inc. Generalized incomplete gamma function, 1998. URL http://functions.wolfram.com/webMathematica/FunctionEvaluation.jsp?name=Gamma3. (online evaluation page).