Numer Algor (2013) 63:187–211
DOI 10.1007/s11075-012-9636-0

ORIGINAL PAPER

ReLaTIve. An Ansi C90 software package for the Real Laplace Transform Inversion

Luisa D'Amore · Rosanna Campagna · Valeria Mele · Almerico Murli

Received: 10 December 2010 / Accepted: 3 August 2012 /
Published online: 4 September 2012

Abstract A software package for numerical inversion of Laplace transforms computable everywhere on the real axis is described. Besides the function to invert the user has only to provide the numerical value (even if it is an approximate value) of the abscissa of convergence and the accuracy required for the inverse function. The software provides a *controlled accuracy*, i.e. it dynamically computes the so-called *maximum attainable accuracy* such that numerical results are provided within the greatest value between the user's required accuracy and the maximum attainable accuracy. This is done because the intrinsic ill posedness of the real inversion problem sometime may prevent to reach the desired accuracy. The method implemented is based on a Laguerre polynomial series expansion of the inverse function and belongs to the class of polynomial-type methods of inversion of the Laplace transform, formally characterized as Collocation methods (C-methods).

Keywords Inverse ill-posed problems · Laplace transform real inversion · Numerical regularization · Collocation methods · Laguerre series expansion

L. D'Amore (✉) · R. Campagna · V. Mele
Department of Mathematics and Applications,
University of Naples Federico II, Naples, Italy
e-mail: luisa.damore@unina.it

R. Campagna
e-mail: rosanna.campagna@unina.it

V. Mele
e-mail: valeria.mele@unina.it

A. Murli
CMCC (Centro Euro-Mediterraneo sui Cambiamenti Climatici), Lecce, Italy
e-mail: almerico.murli@cmcc.it

1 Introduction

Let [20]:

$$F(z) = \int_0^\infty e^{-zx} f(x)dx, \quad z \in \mathcal{C} : \ Re(z) > \sigma_0 \tag{1}$$

be the Laplace transform (Lt) of the real-valued function $f(x)$, $x \in \Re^+ = \{x \in \Re,\ x \geq 0\}$.

The software package `ReLaTIve` (Real Laplace Transform Inversion) obtains approximations of $f(x)$ from numerical values of $F(z)$, when these are available for real values of z.

The user has to provide:

- The Laplace Transform function, F, in the form of any user-supplied black-box;
- The value of $x \geq 0$ where the inverse function f has to be computed;
- The value of $\sigma_0 \in \Re$ (or an approximated value of it), the abscissa of convergence of F;
- A tolerance TOL to the accuracy on $f(x)$

then, the software is designed to return, with N terms, an approximation of $f(x)$ as a Laguerre series representation:

$$\tilde{f}_N(x) = e^{\sigma x} \sum_{k=0}^{N-1} \tilde{c}_k e^{-bx} L_k(2bx), \quad \sigma > \sigma_0,\ b > 0,$$

which satisfies:

$$|f(x) - \widetilde{f}_N(x)| \leq \max\{TOL, \epsilon^*\} \quad or \quad \frac{|f(x) - \widetilde{f}_N(x)|}{|\widetilde{f}_N(x)|} \leq \max\{TOL, \epsilon^*\}, \tag{2}$$

where ϵ^* is the maximum attainable accuracy on $\widetilde{f}_N$ dynamically computed by the algorithm, as detailed in Section 3.1. The role of parameters σ_0, σ, b and recommendations about their use are explained in Section 4. Section 5 describes the software performance and its limitations. Software is designed to provide the inverse Laplace function at a single point or at a set of points, as requested in input. Moreover, if the user wants the inverse function as it is generated, the software returns the coefficients of the Laguerre series expansion.

2 Related works

Since 1930 there are more than 1,000 publications about numerical methods for inverting Lt functions [39]. Typically, the inversion problem is referred to

as *complex inversion* or as *real inversion*, depending on the values available for the Lt function.

This type of classification also takes into account the strong difference underlying the numerical approach. In the *real inversion* Lt is a real function computable on the real axis only and almost all complex inversion methods (such those based on the numerical quadrature of Riemann inversion formula, or on the series expansion) cannot be used in this case just restricting to the real axis Lt evaluations. Indeed, additional difficulties arise in the real case because of the ill posedness of the inversion problem. In this case, due to the ill conditioning of the discrete problem, strong amplifications of the errors occur on the final result. As a consequence, inversion algorithms have to control and mitigate such instabilities in order to compute the solution within the maximum attainable accuracy. This can be obtained introducing the so-called *numerical regularization* or using extended precision calculations if data can be obtained with no limited precision.

In the following we cite numerical algorithms for inverting the Laplace function available in the literature:

1. **Complex inversion**: TOMS Algorithms: `619` [31], `662` [15], `682` [27], `796` [11] and the Nag routine `C06Ilf` [28]. Three `Mathematica` software packages: `FixedTalbotNumericalLaplace` [3], `NumericalInversion` [25] and `NumericalInversion2` [6]. The package `FixedTalbotNumericalLaplace` contains only one function. Laplace transform should be provided as a function ready for multiple precision evaluation in the complex plane. The package `NumericalInversion` provides four complex inversion methods plus one real inversion method. Inversion algorithms are due to Crump, Durbin, Piessens, Stehfest (the Stehfest's algorithm is a real inversion method) and Weeks. The package `NumericalInversion2` contains two complex inversion algorithms (Fourier series approximation and Zakian's method).
2. **Real inversion**: the `MATLAB`© routine `InvertLT.m` [23], an implementation of the method proposed in [22], based on the quadrature of the Mellin transform operator.[1] Two `Mathematica` implementations of the Gaver-Stefhest algorithm proposed originally in [36]: the BigNumber–Stehfest algorithm [4] and the GWR (Gaver–Wynn–Rho) algorithm [2] (these routines use multiprecision, for details see the web page maintained by Valko [39]). The Stehfest's algorithm of `Mathematica` package `NumericalInversion` [25]. Finally, we cite the implementation, in C and Fortan [14], of the ACM Stehfest's algorithm [36].

[1] `InvertLT.m` is a DLL—Windows operating system—written in Microsoft Visual `C++` 2003 and recompiled into `MATLAB`© `5.3`(.NET 2003) afterwards.

3. **The Laplace transform is only available at a finite set of measurements on the real axis:** There are some packages used to compute an approximation of inverse function. In this case, inversion problem is solved via a least square approximation of data set [7, 32, 34].

Finally, in [8] a comprehensive list of methods that are available at the current time is presented.

We underline that in the real case we cannot use the software developed for the complex inversion, because the restriction of the (complex) Laplace function to the real axis does not guarantee the applicability of the complex inversion algorithm in the real case. For instance, if we use the `Algorithm 796`, [10, 11] from TOMS repository, by evaluating the input complex function on the real axis, we get the error diagnostic (`IFAIL=-2`) that marks software failure. Indeed, this algorithm is based on the numerical integration of the Riemann inversion formula, using the trapezoidal rule, and returns the inverse function as a Fourier series expansion where the Fourier coefficients are the values of the Laplace function in the complex plane.

2.1 The present work

In conclusion, besides Stehfest's algorithm [14], numerical algorithms for real inversion are mainly implemented as routines of commercial software such as Mathematica and Matlab.

According to [12] we believe that more than one inversion method should be used on any function because no single method gives optimum results for all purposes and all occasions.[2] To this aim, we develop a free fully automatic `Ansi C90` software package for inverting a Laplace Transform function, when this is available everywhere on the real axis with limited precision (at most equals to the machine precision).

In Section 5 we report the database of Laplace transform functions used to test software performance, an application test function and real-world test problems from Duffy's paper [13]. Finally, some comparisons with Stehfest's algorithm [14] are shown. Other numerical results are reported in the software documentation as provided by the code.

The software is available from Netlib (http://www.netlib.org/numeralgo/) as na37 package.

Requirements on the inverse function and on the Laplace transform to invert are the following:

1. Lt function such that:
   - It can be expressed as $F(z) = z^{-1}G(z)$, where G is analytic at infinity,

[2] *There is still no single best method because of the differing success of the methods[...] more than one numerical inversion method should be used on any unknown function. This is because every method breaks down on some functions, and therefore verification by different methods can greatly increase confidence in any results achieved* [12].

- It may be evaluated on the real axis with a preassigned limited precision, at most equals to the machine precision;

2. The inverse function such that:
   - It is infinitely differentiable for all $x > 0$.

WARNING: the rate of convergence of Laguerre series expansion highly affects software performance. Indeed, due to ill posedness of real inversion problem, even if the Laplace transform belongs to the domain of applicability, it may happen that as N grows the partial sum of the first N terms does not always provides a better approximation of the inverse function. This is because the condition error grows as N. Hence, we set by default $N_{\max} = 40$. The software is designed to select the right value of N so the input required accuracy, or failing this, the so-called maximum attainable accuracy, is reached within $N_{\max} = 40$ terms. If the rate of convergence of Laguerre series expansion is too slow it may happen that the maximum attainable accuracy is greater than input required accuracy. In terms of the Laplace transform function F, the rate of convergence of Laguerre series depends on the singularities of F: if F has a singularity at infinity or on imaginary axis the rate of convergence is not geometric [1]. A singularity at zero can be removed by doing exponential damping but the more difficult case is the singularity at infinity, and this is very common for Laplace transforms.

Such situation is detected by the error indicator `flag = 1`.

3 The algorithm

`ReLaTIve` is a software package for computing the inverse Laplace function using evaluations of its Lt function on the real axis. Hence from now on, we assume that $z \equiv Re(z) \in \Re$. It is based on a Collocation method [17], proposed in its original form in [33, 35], and implements the numerical algorithm proposed in [9].

Collocation methods use a bilinear mapping between the complex plane onto itself, such as:

$$w = \frac{z + \alpha}{z + \beta} \quad (\beta > \alpha) \tag{3}$$

where α and β are fixed numbers. They compute the Lagrange interpolation polynomial which interpolates the function $\Phi(w) = F(z(w))$ at a given set of *collocation knots* defined onto the w-plane. Such methods include expansion approximations of the inverse function based on Laguerre polynomials [24, 29, 33, 35, 41] and the Piessens's method [30]. In [17] the authors generalized and formally characterized such inversion schemes, referring to them as Collocation method (C-methods).

Domain of applicability of the C-methods is the function space S_γ where:

Definition 1 S_γ indicates the set of all functions whose analytical continuation, $E(z)$, can be assumed to be of the form:

$$E(z) = z^{-\gamma} G(z), \tag{4}$$

for some fixed $\gamma > 0$, where G is analytic at infinity.

Definition 2 Let $\sigma > \sigma_0$ be a positive real number and $b > 0$. Consider the Möbius transformation between z-plane and w-plane

$$z = \frac{2b}{1-w} + \sigma - b \tag{5}$$

and the transformation H onto S_γ such that:

$$H : E \in S_\gamma \to H(E(w)) = \frac{2b}{1-w} E(w) \in S_\gamma \tag{6}$$

Observe that the Möbius transformation (5) is the inverse bilinear mapping in (3), where $\alpha = -(\sigma + b)$ and $\beta = b - \sigma$.

A C-method approximates $f(x)$ by:

$$\tilde{f}_N(x) = e^{\sigma x} \sum_{k=0}^{N-1} \tilde{c}_k e^{-bx} L_k^{(\gamma-1)}(2bx), \quad \sigma > \sigma_0, \ b > 0, \tag{7}$$

where $L_k^{(\gamma-1)}$ is the generalized Laguerre polynomial of order γ and degree k.

Applying H to F we get the function $\Phi(w)$ as:

$$\Phi(w) := H(F(z)) = \left(\frac{2b}{1-w}\right)^\gamma F\left(\frac{2b}{1-w} + \sigma - b\right). \tag{8}$$

`ReLaTIve` assumes that the Laplace transform $F \in S_1$, then the software is applicable if

$$F(z) = \frac{G(z)}{z}$$

where G is regular at infinity. The inverse Laplace function $f(x)$ is computed as:

$$\tilde{f}_N(x) = e^{\sigma x} \sum_{k=0}^{N-1} \tilde{c}_k e^{-bx} L_k(2bx), \quad \sigma > \sigma_0, \ b > 0 \quad . \tag{9}$$

Laguerre series expansion in (9) has radius of convergence $R \geq 1$ when $f(x) \in C^\infty[0, \infty)$.

The algorithm of ReLaTIve is made of the following three main steps:

step 1: evaluation of the function $\Phi(w) = \frac{2b}{1-w} F\left(\frac{2b}{1-w} + \sigma - b\right)$ at:

$$w_k = cos\left(\frac{2k+1}{N}\frac{\pi}{2}\right), \quad k = 0, \ldots, N-1$$

step 2: computation of the coefficients $\tilde{c}_k$ occurring in (9)

as the coefficients of the polynomial $l_{N-1} \in \Pi_{N-1}$

that interpolates $\Phi(w)$ on w_k:

$$l_{N-1}(w) = \sum_{j=0}^{N-1} \tilde{c}_j w^j.$$

step 3: computation of :

$$\tilde{f}_N(x) = e^{\sigma x} \sum_{k=0}^{N-1} \tilde{c}_k e^{-bx} L_k(2bx)$$

ReLaTIve algorithm

Remark The real C-method implemented by ReLaTIve, is strictly related to the complex C-method underlying the TOMS software package in [15]. Main difference consists in the computation of the $\tilde{c}_k$ occurring in (9). These quantities, being the Taylor series coefficients of Φ in the w-plane [17], in [15] are obtained using in the complex plane Cauchy integral representation of the derivatives of Φ. In ReLaTIve, they are approximated by coefficients of the Lagrange interpolation polynomial of Φ, on Chebyshev zeros.

The core of ReLaTIve is step 2, that is computation of the series coefficients via the solution of a Vandermonde linear system generated by the w_k defined at step 1. We employ the Björck–Pereira algorithm [5]. The error analysis presented in [21] shows that for such problems the *componentwise* rounding error analysis provides more realistic estimates than the *norm-wise* error analysis. This mainly occurs if the components of the solution have different orders of magnitude. Indeed, the standard bounds derived using the ∞-norm of both data and solution cannot take into account the magnitude of each component. The componentwise rounding error depends linearly on N and on the machine precision u. A technique for computing a *forward* error bound, which is valid for any distribution of the points generating the Vandermonde matrix, is also derived in [21].

3.1 The maximum attainable accuracy and the controlled accuracy

ReLaTIve provides a *controlled accuracy* on the computed solution. Let us explain what this means.

Even thought Laplace transform function belongs to domain of applicability, i.e. Laguerre series expansion converges to the inverse function f, due

to ill-posedness of the problem, as N grows $\tilde{f}_N$ does not always provide a better approximation of f.[3] This occurs if the rate of convergence of Laguerre series expansion is too slow. In this case, truncation error of Laguerre series expansion does not decrease fast enough towards zero (i.e. N has to grow too much in order to provide a better approximation) and ill conditioning of the inverse problem (that increases as N does) dominates.

As it will be detailed in the following, the software `ReLaTIve` addresses ill posedness by computing:

1. The so-called *maximum attainable accuracy* as the minimum value assumed by the sum of the truncation error and the condition error,
2. The *controlled accuracy* as the maximum value between the maximum attainable accuracy and the input required tolerance,
3. $\tilde{f}_{N_{\text{opt}}}$ such that it satisfies the controlled accuracy.

Now we describe how computation of N_{opt} is performed (for details see [9]).

Definition 3 Let:

$$f_N(x) = e^{\sigma x} \sum_{k=0}^{N-1} c_k e^{-bx} L_k(2bx), \tag{10}$$

where $c_k = \dfrac{\Phi^{(k)}(0)}{k!}$. The difference:

$$e_{\text{trunc}}(x, \sigma, b, N) = f(x) - f_N(x) = e^{\sigma x} \sum_{k=N}^{\infty} e^{-bx} c_k L_k(2bx) \tag{11}$$

is the **truncation error** introduced on $f_N(x)$.

Definition 4 The difference:

$$e_{\text{discr}}(x, \sigma, b, N) = f_N(x) - \widehat{f}_N(x) = e^{\sigma x} \sum_{k=0}^{N-1} e^{-bx} (c_k - \widehat{c}_k) L_k(2bx) \tag{12}$$

where $\widehat{c}_k$ are obtained as in step 2 of `ReLaTIve` algorithm, is the **discretization error** introduced on $f_N(x)$.

Discretization error occurs substituting coefficients of MacLaurin series $(c_k)_k$ of Φ in the w-plane, by $(\widehat{c}_k)_k$, that is coefficients of the Lagrange interpolation polynomial of Φ, on Chebyshev zeros [17, 18, 33].
Beyond truncation and discretization errors, numerical computation of $\widehat{f}_N(x)$ involves roundoff errors. Starting from the error analysis developed in [9, 17, 18], here we give some definitions concerning how C-method works in a finite precision arithmetic system $\Im$.

[3] Actually, the sensitivity of this method to input errors as N grows has been recognized from the beginnings [35].

Let:

$$\widetilde{f}_N(x) = e^{\sigma x} \sum_{k=0}^{N-1} \widetilde{c}_k e^{-bx} L_k(2bx), \tag{13}$$

where $\{\widetilde{c}_k\}_{k=0,\ldots,N-1}$ are the computed values of coefficients $\{\widehat{c}_k\}_{k=0,\ldots,N-1}$ as in step 2 of `ReLaTIve`, in a finite precision arithmetic system.

Definition 5 The difference:

$$e_{\text{cond}}(x, \sigma, b, N) = \widetilde{f}_N(x) - \widehat{f}_N(x) = e^{\sigma x} \sum_{k=0}^{N-1} e^{-bx} (\widetilde{c}_k - \widehat{c}_k) L_k(2bx) \tag{14}$$

is the **conditioning error** given on $\tilde{f}_N(x)$.

By (11), (12), (13) and (14) it follows that:

$$f(x) = \widetilde{f}_N(x) + e_{\text{trunc}}(x, \sigma, b, N) + e_{\text{discr}}(x, \sigma, b, N) + e_{\text{cond}}(x, \sigma, b, N) \tag{15}$$

The major challenge of designing numerical algorithms for inverting Laplace transforms is to compute $\widetilde{f}_N(x)$ for *any* value of x with the required accuracy. This is primarily due to the exponential factor $e^{\sigma x}$ in (13) which degrades the accuracy of $\widetilde{f}_N(x)$ as x varies. Hence, we first introduce the so-called pseudo-errors:

$$\epsilon_{td}(x, \sigma, b, N) := \frac{e_{\text{discr}} + e_{\text{trunc}}}{e^{\sigma x}}$$

$$\epsilon_{\text{cond}}(x, \sigma, b, N) := \frac{e_{\text{cond}}(x, \sigma, b, N)}{e^{\sigma x}}$$

which provide uniform accuracy with respect to $e^{\sigma x}$, and instead of (15) we get:

$$f(x) = \widetilde{f}_N(x) + e^{\sigma x} \cdot GEE(\sigma, b, N) \tag{16}$$

where the Global (pseudo)Error Estimate GEE is such that:[4]

$$GEE(\sigma, b, N) = \epsilon_{td}(x, \sigma, b, N) + \epsilon_{\text{cond}}(x, \sigma, b, N) \quad .$$

Theorem 1 *The function $GEE(N)$ is convex and it assumes its absolute minimum value at:*

$$N^* = log_R \frac{K^* \, log(R)}{\Omega_\Phi} \tag{17}$$

where $K^* = |\Phi(0)|$, R is the radius of convergence of the Laguerre series expansion, $\Omega_\Phi = u \, |\Phi(0)| \, \max_k |M_k|$, u is the machine precision, and $|M_k|$ are

[4]In the following we refer to GEE as $GEE(N)$ omitting the dependence of GEE on parameters σ and b.

quantities computed during the solution of the Vandermonde linear system arising in step 2*. They depend on the componentwise rounding error of the Björck–Pereira algorithm.*

For the proof see [9].

Definition 6 The minimum of GEE:

$$\epsilon^* = \min_{N \geq 0} GEE(N) = GEE(N^*) \tag{18}$$

is the **maximum attainable accuracy**.

The algorithm obtains $N_{\text{opt}} \leq N^* \leq N_{\max}$ minimizing a computable estimate of GEE (see Fig. 1), given by the sum of:

– E_{td}: a **computable estimate** of ϵ_{td}:

$$E_{td}(\sigma, b\,, N) = (\widetilde{c}_{N-3} + \widetilde{c}_{N-2} + \widetilde{c}_{N-1})/3$$

– E_{cond}: a **computable estimate** of ϵ_{cond},

$$E_{\text{cond}}(\sigma, b\,, N) = N\ u\ K^* \max_{k=0,\ldots N-1} |M_k|.$$

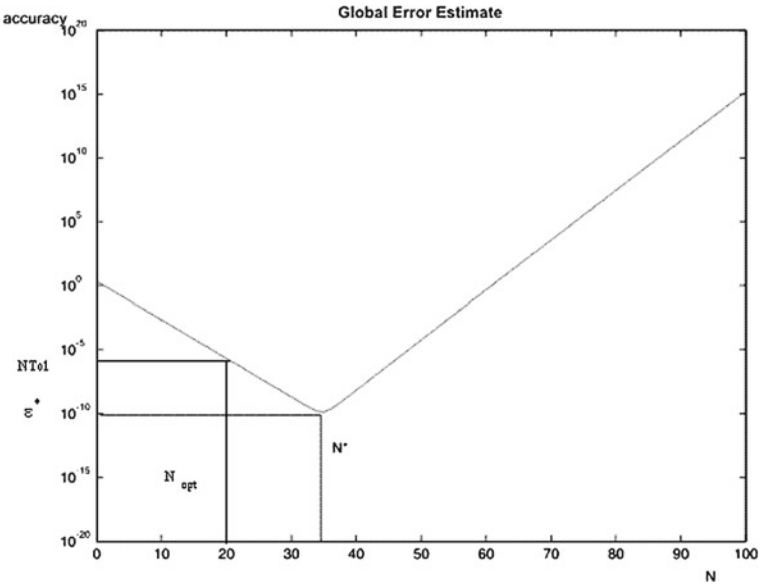


Fig. 1 GEE behavior versus N. The Laplace transform is $F(z) = \frac{z}{(z^2+1)^2}$, $K^* = 1$, $R = 1.8$, the minimum value of GEE is obtained at $N^* = 35$ and $\epsilon^* = GEE(35) = O(10^{-11})$. $x = 1$, $\sigma = 0.7$ and $TOL = 10^{-5}$. Observe that in this case, $N_{\text{opt}} = 20$

Hence, $\widetilde{f}_{\mathrm{Nopt}}(x)$ is computed such that:

$$f(x) \simeq \widetilde{f}_{\mathrm{Nopt}}(x) + e^{\sigma x} \cdot GEE(\sigma, b, N_{\mathrm{opt}}) \tag{19}$$

and:

$$e^{\sigma x} \cdot GEE \leq \max\{\epsilon^*, TOL\} \quad \text{and/or} \quad \frac{|e^{\sigma x} \cdot GEE|}{|\widetilde{f}_{\mathrm{Nopt}}(x)|} \leq \max\{\epsilon^*, TOL\}$$

4 The role of incidental parameters

In this section we make some comments about parameters σ_0, σ and b used in (9) and their computation:

– The role of $\sigma_0 \in \Re$: most inversion algorithms requires the numerical value of this parameter as input. It may be obtained as the real part of the rightmost singularity of $F(z)$. One of the reason for having to know the value of σ_0 is that the Lt function may have a singularity before σ_0. Moreover, the parameter σ must exceed σ_0 in value. If it is does not, the Laguerre series expansion for $f(x)$ will not converge. In `ReLaTIve` the numerical value of σ_0, or an estimate of it, is required as input parameter. Using the input data, the algorithm performs a *damping* of F (and the tolerance TOL), as detailed below

  1. The inversion algorithm is applied to the shifted Lt function, let us say $FS(z)$, where:

     $$FS(z) = F(z + \sigma_0)$$

     which has abscissa of convergence $\sigma_0^{FS} = 0$ and the inverse function of $FS(z)$, let say g_{Nopt}, is provided within the *damped* tolerance $Ntol = TOL/e^{\sigma_0 x}$,
  2. The result $f_{\mathrm{Nopt}}(x) = e^{\sigma_0 x} \cdot g_{\mathrm{Nopt}}(x)$ is returned.

  Observe that if the user cannot determine σ_0 precisely so he/her makes a small error and uses the value $\widetilde{\sigma_0}$ that is either slightly larger or slightly smaller than the right value of σ_0, i.e.

  $$\widetilde{\sigma_0} = \sigma_0 + \eta \quad ,$$

  the perturbation η propagates on $\widetilde{f}_{\mathrm{Nopt}}(x)$ as $e^{\eta x}$.
  Indeed, $\widetilde{\sigma_0}$ is the abscissa of convergence of the shifted Lt

  $$M(z) = F(z - \eta)$$

  whose inverse function is

  $$m(x) = e^{\eta x} f(x)$$

Then, instead of $\widetilde{f}_{\mathrm{Nopt}}(x)$ `ReLaTIve` computes

$$\widetilde{m}_{\mathrm{Nopt}}(x) = e^{\eta x}\,\widetilde{f}_{\mathrm{Nopt}}(x)$$

Hence,

$$\left|\frac{\widetilde{m}_{\mathrm{Nopt}}(x) - \widetilde{f}_{\mathrm{Nopt}}(x)}{\widetilde{f}_{\mathrm{Nopt}}(x)}\right| = |e^{\eta x} - 1| = O(e^{\eta x}) \quad (\eta \cdot x \to \infty)$$

If $\eta < 0$ results may be completely wrong, even if the propagated error $e^{\eta x}$ decreases. This occurs if $F(z)$ has a singularity in the region $\{\widetilde{\sigma_0} < z < \sigma_0\}$, moreover in this region it is not a Lt.

– The role of σ: in order to bound the error amplification due to the exponential factor $e^{\sigma x}$, the best value of σ should be the smallest possible, as x grows. On the other hand, because the inherent instabilities of the discrete problem can be controlled if the truncation error rapidly goes to zero, taking into account that the rate of convergence of Laguerre series expansion increases as σ grows, a suitable choice of σ should be the largest possible. Then, if x is sufficiently small, σ could be large. Of course, it is quite difficult to know what the user should do!
`ReLaTIve` dynamically changes the value of σ according to that of x. Indeed, the value of σ is determined dynamically, by fixing the product σx equals to 2.5. We performed many experiments by choosing the value of the product σx between 1 and 4. A value of σx less than 1 gives a too small σ if x increases degrading the series convergence. On the contrary, a value of σx greater than 4 causes a too large exponential growth factor. We found that,

$$\sigma = \frac{2.5}{x}$$

allows to get, on average, a good trade off between the convergence rate of the series expansion and errors amplification. At $x = 0$, by default we set $\sigma = 4$.

– Regarding b, we follows [15, 16] where the connection between σ and b has been investigated. In particular, we set:

$$b = 2.5\sigma.$$

Parameters σ and b are chosen so as to provide a good trade off between software reliability, robustness and efficiency. We invite the user to not change the values of both σ and b, because in this case reliability and robustness cannot be guarantee.

– Parameter $N_{\max}$. The algorithm determines the maximum attainable accuracy as $GEE(N_{\mathrm{opt}})$ where $N_{\mathrm{opt}} \le N^*$, by checking the values of $GEE(N)$ for $N \le N_{\max}$.
We experimentally found that for $N > 40$ for most of the test functions the condition error blows up then we set $N_{\mathrm{opt}} \le N^* \le 40$, i.e. we set by default $N_{\max} = 40$. If it needs, the user may change the value of $N_{\max}$. Of course, increasing this value alters the software efficiency because the

Table 1 Functions test of software database

Test #	Lt	ILt	σ_0
(a)			
1	$\frac{1}{z}$	1	0
2	$\frac{1}{z^2}$	x	0
3	$\frac{1}{1+2z}$	$0.5e^{-0.5x}$	−0.5
4	$\frac{1}{(z+2)^2}$	xe^{-2x}	−2
5	$\frac{z}{(z-1)^2}$	$(1+x)e^x$	1
6	$\frac{1}{z-2}$	e^{2x}	2
7	$\frac{1}{(z+a)^5}$	$\frac{x^4}{4!e^{ax}}$	-a
8	$\frac{1}{(z+a)(z+b)}$	$\frac{(e^{-ax}-e^{-bx})}{b-a}$	max(-a,-b)
9	$\frac{z}{(z+a)(z+b)}$	$\frac{(ae^{-ax}-be^{-bx})}{a-b}$	max(-a,-b)
10	$\frac{z^2-1}{(z^2+1)^2}$	$x\cos(x)$	0
11	$\frac{z}{z^2+a^2}$	$\cos(ax)$	0
12	$\frac{1}{z^2+1}$	$\sin(x)$	0
13	$\frac{1}{z^2+z+1}$	$\frac{2}{\sqrt{3}}\left[e^{-\frac{x}{2}}\sin\left(\frac{x\sqrt{3}}{2}\right)\right]$	0
14	$\frac{1}{(z+2.5)^2+1}$	$e^{-2.5x}\sin(x)$	0
15	$\frac{1}{z(z^2+a^2)}$	$\frac{(1-\cos(ax))}{a^2}$	0
16	$\frac{1}{z^2(z^2+a^2)}$	$\frac{(ax-\sin(ax))}{a^3}$	0
17	$\frac{1}{(z^2+a^2)^2}$	$\frac{(\sin(ax)-ax\cos(ax))}{2a^3}$	0
18	$\frac{z^2}{(z^2+a^2)^2}$	$\frac{(\sin(ax)+ax\cos(ax))}{2a}$	0
19	$\frac{8a^3z^2}{(z^2+a^2)^3}$	$(1+a^2x^2)\sin(ax)-ax\cos(ax)$	0
20	$\frac{z}{(z^2+a^2)(z^2+b^2)}$	$\frac{(\cos(ax)-\cos(bx))}{(b^2-a^2)}$	0

Table 1 (continued)

Test #	Lt	ILt	σ_0
21	$\frac{z+a}{(z+a)^2+b^2}$	$\cos(bx)e^{-ax}$	0
22	$\frac{3a^2}{z^3+a^3}$	$e^{-ax}-e^{0.5ax}(\cos(0.5\sqrt{3}ax)-$	
		$-\sqrt{3}\sin(0.5\sqrt{3}ax)$	0
23	$\frac{0.5^2}{z^2(z^2-0.5^2)}$	$\frac{1}{0.5}\sinh(0.5x)-x$	0.5
24	$\frac{1}{z^2-a^2}$	$\sinh(ax)/a$	a
25	$\frac{z}{z^2-a^2}$	$\cosh(ax)$	a
26	$\frac{4a^4}{z^4+4a^4}$	$\sin(ax)\cosh(ax)-$	
		$-\cos(ax)\sinh(ax)$	0
27	$\frac{z}{z^4+a^4}$	$\sin(ax)\sinh(ax)/2a^2$	0
28	$\frac{1}{z^4-a^4}$	$(\sinh(ax)-\sin(ax))/2a^3$	a
29	$\frac{z}{z^4-a^4}$	$(\cosh(ax)-\cos(ax))/2a^2$	a
30	$\log\left(\frac{z+a}{z+b}\right)$	$(e^{-bx}-e^{-ax})/x$	$\max(-a,-b)$
31	$\log\left(\frac{z^2+a^2}{z^2}\right)$	$2(1-\cos(ax))/x$	0
32	$\log\frac{z+0.5}{z-0.5}$	$\frac{2}{x}\sinh(0.5x)$	0.5
33	$\frac{1}{\sqrt{z}(z-a^2)}$	$e^{a^2x}\frac{\mathrm{Erf}(a\sqrt{x})}{a}$	a^2
34	$\frac{(b^2-a^2)}{\sqrt{z}(z-a^2)(\sqrt{z}+b)}$	$e^{a^2x}\left(\left(\frac{b}{a}\mathrm{Erf}(a\sqrt{x})\right)-1\right)+$	
		$+e^{b^2x}\mathrm{Erfc}(\sqrt{(t)})$	$\max(\lvert a\rvert,\lvert b\rvert)$
35	$\frac{1}{\sqrt{z+a}(z+b)^{3/2}}$	$\frac{1}{\exp((1/2)(b+a)x)}\cdot$	
		$\cdot x(\mathrm{BesselI}(0,(0.5)(a-b)x)+$	
		$+\mathrm{BesselI}(1,(0.5)(a-b)x))$	$\max(-a,-b)$

algorithm has to search for the minimum of GEE along a larger set of values. Anyway, in general, the value of N at which GEE reaches the minimum does not change too much.

Table 1 (continued)

Test #	Lt	ILt	σ_0
(b)			
36	$\frac{\sqrt{z+2a}-\sqrt{z}}{\sqrt{z+2a}+\sqrt{z}}$	BesselI$(1, ax)/(x\exp(ax))$	0
37	$\frac{(a-b)^n}{(\sqrt{z+a}+\sqrt{z+b})^{2n}}$	$\frac{n\cdot \text{BesselI}(n, 1/2(a-b)x)}{x\exp((1/2)(b+a)x)}$	max (-a,-b)
38	$\frac{1.}{(\sqrt{z+a}+\sqrt{z})^{2\nu}\sqrt{z+a}\sqrt{z}}$	BesselI$(\nu, ax/2)/((a^\nu)e^{0.5ax})$	0
39	$\frac{(z-\sqrt{z^2-a^2})^\nu}{\sqrt{z^2-a^2}}$	(a^ν)BesselI(ν, ax)	a
40	$\frac{1}{\sqrt{z^2+a^2}\exp(k\sqrt{z^2+a^2})-z}$	BesselJ$(0, a\sqrt{x^2+2kx})$	0
41	$\frac{\sqrt{z+2a}}{\sqrt{z}}-1$	$\frac{a(\text{BesselI}(1, ax)+\text{BesselI}(0, ax))}{e^{ax}}$	1
42	$\frac{1}{\sqrt{z^2+a^2}}$	BesselJ$(0, ax)$	0
43	$\frac{1}{z^{n+1/2}}$	$\frac{x^{n-1/2}}{\Gamma(n+1/2)}$	0
44	$\frac{1.}{z^{3/2}\exp(k\sqrt{z})}$	$\frac{2\sqrt{x}\exp(-k^2/(4x))}{\sqrt{\pi}}-k\text{Erfc}(k/(2\sqrt{x}))$	0
45	$F_{du1}(z)$	$f_{du1}(x)$	0
46	$F_{du2}(z)$	$f_{du2}(x)$	0
47	$F_{du3}(z)$	$f_{du3}(x)$	0
48	$F_{du4}(z)$	$f_{du4}(x)$	0
49	$F_{du5}(z)$	$f_{du5}(x)$	0

5 Software performance and limitations

Table 1a and b show the software database of test functions. The database is made of standard Lt functions selected from `SET A` provided by Valko et al. [40]. It is worth to note that if the user needs to compute the inverse function at $x \to \infty$, it is recommended that he/her employs numerical methods that are suitable to compute inverse Laplace transforms at x-values *large*.[5] All numerical results as provided by the software are shown in the software documentation, here we only report errors plots for tests number $5-6-10-24-27$. We select F_5, F_6 and F_{24}, because these Laplace functions have $\sigma_0 > 0$ while functions F_{10} and F_{27} have $\sigma_0 = 0$. Figures 2, 3, 4 and 5 show the following

[5]Such methods usually employ abelian/tauberian properties of both the Laplace transform and of its inverse function to provide reliable numerical results as $x \to \infty$.

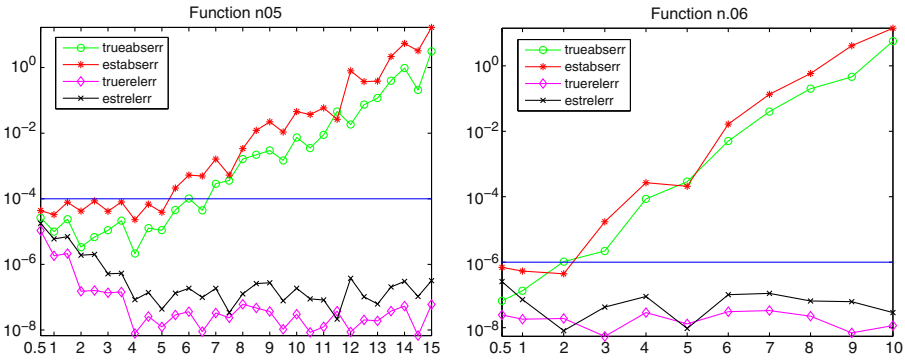


Fig. 2 Errors versus x-values. *Left*: $F_5(z)$, $TOL = 10^{-4}$. *Right*: $F_6(z)$, $TOL = 10^{-6}$

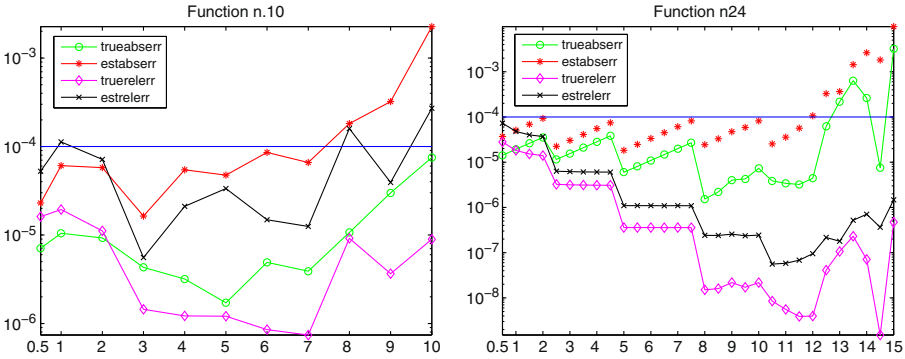
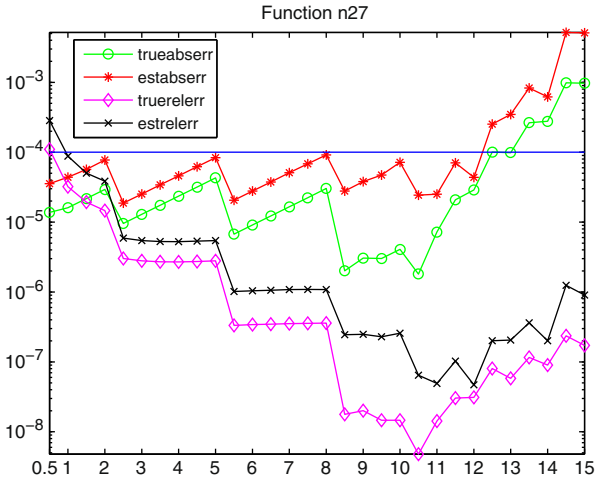


Fig. 3 Errors versus x-values. *Left*: $F_{10}(z)$, $TOL = 10^{-4}$. *Right*: $F_{24}(z)$, $TOL = 10^{-4}$

Fig. 4 Test function $F_{27}(z)$: errors versus x-values. $TOL = 10^{-4}$



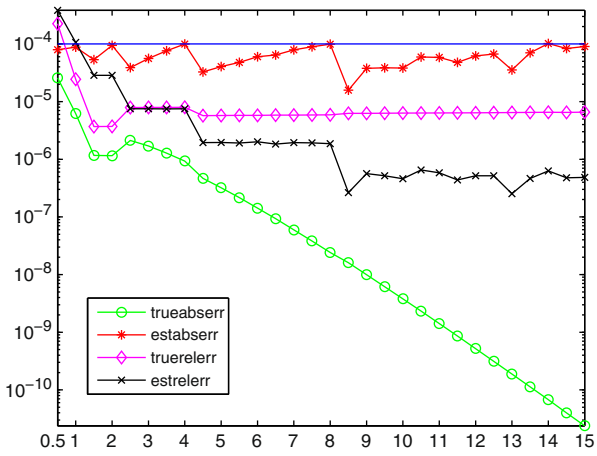


Fig. 5 The application: errors versus x-values. $TOL = 10^{-4}$

quantities: `trueabserr`—the absolute error on the computed result $|f(x) - f_{\text{comp}}(x)|$, `estabserr`—the estimated absolute error, `truerelerr`—the true relative error on the computed result $|f(x) - f_{\text{comp}}(x)|/f(x)$, `estrelerr`—the estimated relative error and `TOL`—the input required accuracy. Looking the errors' plots, it is quite evident that the obtained accuracy is in agreement with the function behavior.

Finally, we compared results obtained by using `ReLaTIve` with those obtained using the routine [14] on functions number 6 and 10. In the software documentation we show results as provided by two routines. Note that the Stefhest's algorithm provides results completely wrong.

5.1 Application test

The problem under consideration has practical application in many chemical, pharmaceutical and petroleum industries as well as in environmental engineering and waste management. It concerns design or selection of an appropriate electrochemical reactor for a specific purpose. Measurement and critical analysis of electrolyte behavior play an important role in the effective design of an electrochemical reactor. In particular, the continuous stirred tank reactor is frequently adopted as a model for electrochemical reactors. Characteristic equations governing the dynamics of the continuous stirred tank mixer/reactor are solved in the Laplace domain and finally the solution is obtained by taking inverse Lt [19, 26, 37, 38]. In such applications, the Lt function to invert is of rational type such those reported in our database. For instance, for the system reported in [19] the Lt to invert is

$$F(z) = \frac{1}{(1 + 0.845z)^3}$$

whose inverse function is

$$f(x) = 0.8287x^2e^{-1.183x}$$

Figure 6 shows errors as provided by the code for $TOL = 10^{-4}$ while table shown numerical results is reported in software documentation. Observe that both absolute and relative errors satisfy the tolerance. In this case $F(z)$ has $\sigma_0 < 0$ and inverse function decreases.

Next four tests comes from [13] (numeration follows that of [13]). First Laplace function arises in circuit theory:

$$F_{du1}(z) = \frac{1}{z(z+c)}\left[\frac{1}{2zh} - \frac{e^{-2zh}}{1-e^{-2zh}}\right]$$

whose inverse function is:

$$f_{du1}(x) = \frac{1}{2c} + \frac{e^{-cx}}{c}\left[\frac{1}{2ch} - \frac{e^{2ch}}{e^{2ch}-1}\right] - \frac{h}{\pi}\sum_{n=1}^{\infty}\frac{\sin\left(\frac{n\pi x}{h} - \tan^{-1}(\frac{n\pi}{ch})\right)}{n\sqrt{n^2\pi^2 + c^2h^2}} \quad ,$$

using $c = h = 1$ and 10^7 terms in the summation.

Second test arises in the study of longitudinal impact on viscoplastic rods:

$$F_{du2}(z) = \frac{(100z-1)\sinh(\sqrt{z}/2)}{z[z\sinh(\sqrt{z}) + \sqrt{z}\cosh(\sqrt{z})]}$$

with inverse function:

$$f_{du2}(x) = -\frac{1}{2} + \sum_{n=1}^{\infty}\left(100 + \frac{1}{b_n^2}\right)\frac{2b_n\sin(b_n/2)e^{-b_n^2x}}{(2+b_n^2)\cos(b_n)}$$

where b_n are such that $b_n\tan(b_n) = 1$.

Third Laplace transform arises in viscous fluid mechanics problems:

$$F_{du3}(z) = \frac{1}{z}e^{-r\sqrt{\frac{z(1+z)}{1+cz}}}$$

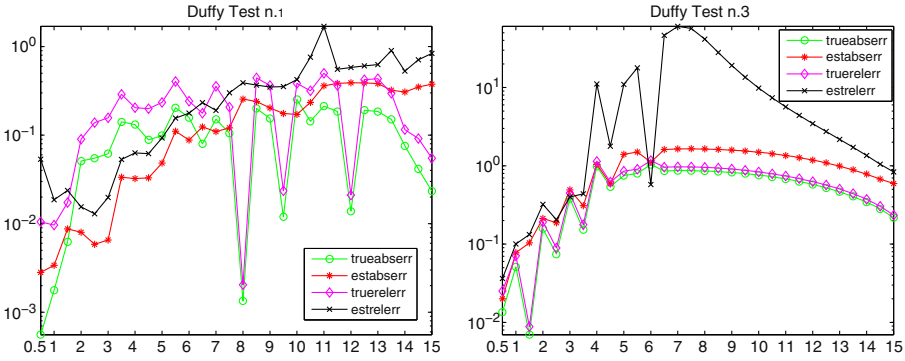


Fig. 6 *Left*: Test function $F_{du1}(z)$: plot of the errors versus x-values. $TOL = 10^{-2}$. RMSE = 3.74062$e-$02. *Right*: Test function $F_{du3}(z)$: plot of the errors. $TOL = 10^{-2}$. RMSEP = 1.72276$e-$01

with inverse function:

$$f_{du3}(x) = \frac{1}{2} + \frac{1}{\pi}\int_0^{\infty} e^{-rm\sqrt{u/2}(\cos(\theta)-\sin(\theta))}$$

$$\sin(xu - rm\sqrt{u/2}(\cos(\theta) - \sin(\theta)))\frac{du}{u} \quad ,$$

where
$$m = \left[\frac{1+u^2}{1+c^2u^2}\right]^{1/4}, \quad 2\theta = \tan^{-1}(u) - \tan^{-1}(cu), \quad c = 0.4,\ r = 0.5$$

Fourth Laplace transform arises in the study of shock waves in diatomic chains:

$$F_{du4}(z) = \frac{e^{-2\Psi}}{z}$$

where

$$\cos(\Psi) = \sqrt{1 + z^2 + z^4/16}$$

The inverse functions is:

$$f_{du4}(x) = 1 - \frac{1}{\pi}\int_0^{u_1} [\sin(ux + 2k) - \sin(ux - 2k)]\frac{du}{u}$$

$$+\frac{1}{\pi}\int_{u_2}^{4} [\sin(ux + 2k) - \sin(ux - 2k)]\frac{du}{u},$$

where

$$\cos(k) = \frac{1}{4}\sqrt{(u_1^2 - u^2)(u_2^2 - u^2)}$$

and

$$u_1 = 2\sqrt{2 - \sqrt{3}}, \quad u_2 = \sqrt{2 + \sqrt{3}}$$

Final test function arises in the theory of beams:

$$F_{du5}(z) = \frac{z - \sqrt{z^2 - c^2}}{\sqrt{z}\sqrt{z^2 - c^2}\sqrt{z - N\sqrt{z^2 - c^2}}}$$

whose inverse function is:

$$f_{du5}(x) = \frac{2}{\pi}\int_0^{c} \cosh(xu)\frac{u\sqrt{(R+u)/2} + \sqrt{c^2 - u^2}\sqrt{(R-u)/2}}{R\sqrt{c^2 - u^2}\sqrt{u}}$$

$$+\frac{2}{\pi}\int_0^{b} \frac{u - \sqrt{c^2 + u^2}}{\sqrt{u}\sqrt{c^2 + u^2}\sqrt{N\sqrt{c^2 + u^2} - u}}\cos(xu)du,$$

where

$$R = \sqrt{u^2 + N^2(c^2 - u^2)}, \quad b = \sqrt{(1 - N)/(1 + N)}, \quad c = (1 - N)/N$$

and $N = 0.5$.

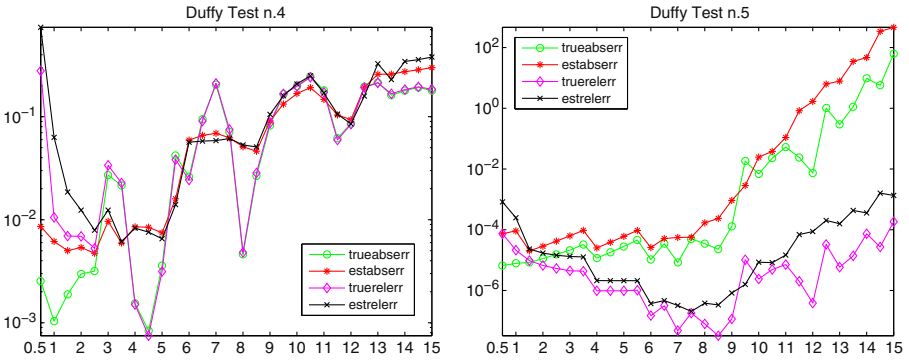


Fig. 7 *Left*: Test function $F_{du4}(z)$: plot of the errors versus x-values. $TOL = 10^{-2}$. RMSEP $= 9.15109e-03$. *Right*: Test function $F_{du5}(z)$: plot of the errors. $TOL = 10^{-4}$. RMSEP $= 2.93327e-02$

Observe that, except test function F_{du5}, all the Duffy's Laplace transforms depend on the exponential function or the square root of z. For such function, as explained in [1], the Laguerre series expansion converges slowly[6] and this is a serious difficulty inherent the Laguerre series expansion. Hence, we expect that the software performs poorly on F_{du1} and F_{du3}, slightly better on F_{du4} and F_{du5} and wrongly on F_{du2}, as also explained in Section 3.1. Anyway, by looking at the maximum attainable accuracy the user is warned that in these cases the problem is extremely ill conditioned (*spike behavior or extremely rapid oscillations are virtually impossible to be filtered out by numerical techniques* [8]).

Figures 6 and 7 show the errors plot for Duffy's tests. Tables shown numerical results are reported in the software documentation. Since these test functions arise from real world applications, to measure the overall difference between the Laplace transform function and the computed result, we also use the Root Mean Square Error (RMSE), defined as in [12, 13]:

$$RMSEP = \sqrt{\frac{\sum_{i=1,\ldots,P} \frac{(s_F(p_i) - F(p_i))^2}{e^{-p_i}}}{\sum_{i=1,\ldots,P} e^{-p_i}}}, \qquad P = 30$$

The P points p_i, $i = 1, \ldots, P$ are uniformly distributed on $[0.5, 15]$. We choose the RMSE because, as in [12, 13], RMSE gives a fair indication of the accuracy of the software on relatively small values of x, as those that we have selected. As already explained we expect the software to achieve not very large accuracy, indeed the RMSE is on average of order 10^{-2}.

[6] In [1] the authors show that in presence of the square root function the Laguerre series coefficients tend to zero as $1/\sqrt{\pi n}$. By running the Garbow's software—based on the Laguerre series expansion in the complex plane [15]—on Duffy's tests, we have experimented that in order to reach an accuracy of 10^{-2} on the inverse function of F_{du1} and F_{du3} it needs the sum of 32 terms, while for the inverses of F_{du4} and F_{du5} only 8 terms and finally for F_{du2} are needed 256 terms.

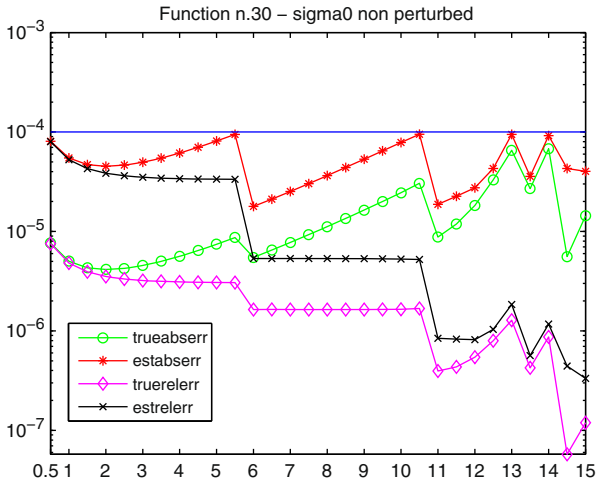


Fig. 8 Test function $F_{30}(z)$: errors versus x-values. $TOL = 10^{-4}$. $\sigma_0 = 0.5$ not perturbed

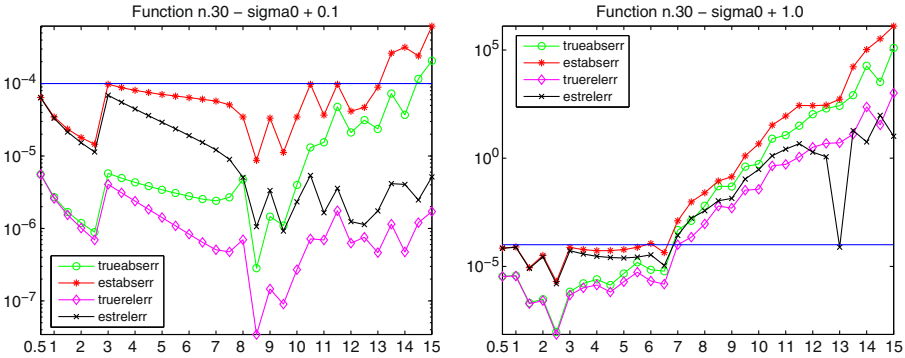


Fig. 9 Test function $F_{30}(z)$: errors versus x-values. $TOL = 10^{-4}$. *Left*: $\sigma_0 = 0.5 + 0.1$. *Right*: $\sigma_0 = 0.5 + 1.0$

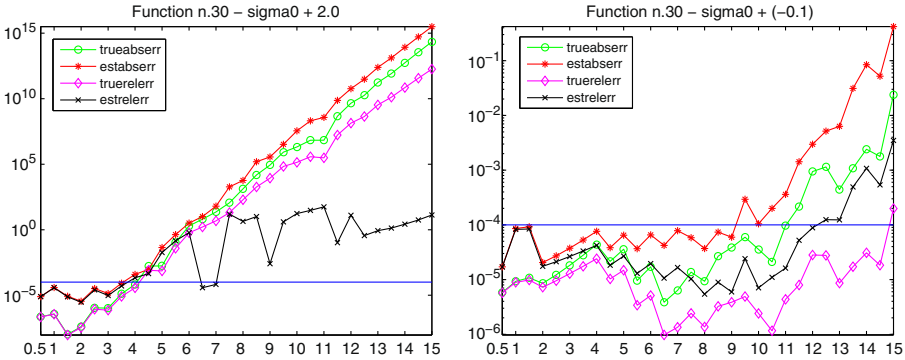


Fig. 10 Test function $F_{30}(z)$: errors versus x-values. $TOL = 10^{-4}$. *Left*: $\sigma_0 = 0.5 + 2.0$. *Right*: $\sigma_0 = 0.5 - 0.1$

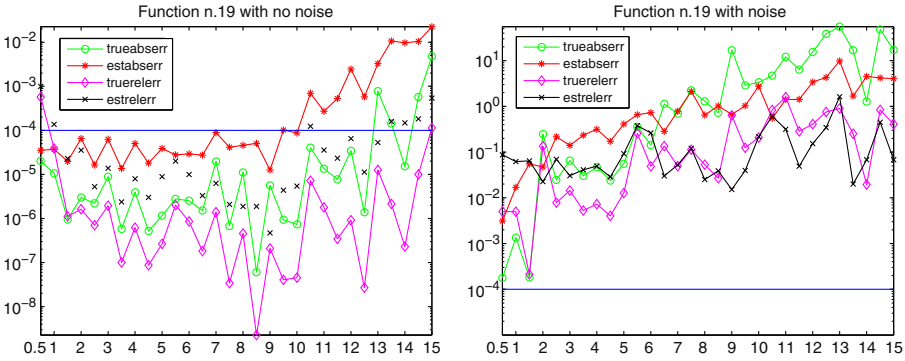


Fig. 11 *Left*: Test function $F_{19}(z)$: errors versus x-values. $TOL = 10^{-4}$. *Left*: no noise on $F(z)$. *Right*: $F(z)$ noisy. $\widetilde{F}_{19}(z) = F_{19}(z) + \text{noise}(z)$, $\text{noise}(z) = \text{rand} \cdot F_{19}(z)$ where rand is a normal distributed noise term with $\|\text{rand}\|_2 = 10^{-4}$

5.2 Software sensitivity

We perform experiments using a perturbed value of the abscissa of convergence σ_0. As test function we consider F_{30}, whose abscissa of convergence is $\sigma_0 = 0.5$. The perturbed value of σ_0 is $\widetilde{\sigma_0} = \sigma_0 + \eta$ where $\eta = \pm 1, \pm 2, \pm 10^{-1}$.

We note that if $\eta = \pm 10^{-1}$, in many cases its amplification on the computed values of the inverse function does not change the order of magnitude or the first significant digit of the reference results. If $\eta = \pm 10^{1}, \pm 10^{2}$ the exponential growth factor $e^{\eta x}$ dominates as x grows, and N_{opt} increases. As expected, if $\widetilde{\sigma_0} < \sigma_0$ the Lt function may have singular points at $\widetilde{\sigma_0} < z < \sigma_0$, and a `nan` occurs.[7] So we recommend the user to use the right value of σ_0, or an upper bound of it. Figure 8, 9 and 10 show the errors' plot. Observe that as σ_0 increases, the more the inverse function does and the errors curves follow this trend.

Further, we use a Laplace transform function with additive random noise on $F(z)$ values with noise level of 1 %, i.e. we consider as Laplace function to invert the following functions

$$\widetilde{F}_{19}(z) = F_{19}(z) + \text{noise}(z), \quad \text{noise}(z) = \text{rand} \cdot F_{19}(z)$$

where rand is a normal distributed noise term with $\|\text{rand}\|_2 = 10^{-4}$. Results, shown in Fig. 11, say that if the relative noise on data is of about 10^{-4} the software reaches a maximum accuracy on inverse function of about $10^{-2} - 10^{-1}$. In other words, in this case the perturbation on data values propagates on the inverse function by an amplification factor of about $10^2 - 10^3$. This fact mainly depends on the intrinsic ill-posedness of such an inverse problem. Tables shown numerical results are reported in the software documentation.

[7]Observe that even if in this case, `flag` returns 0 the user has to pay attention to the *nan* occurrences.

6 Discussion and concluding remarks

We describe a fully automatic `Ansi C90` software package that may be used for real inversion of the Laplace transform, i.e. when the Laplace function is available everywhere in the real line. The method implemented is based on a Laguerre polynomial series expansion of the inverse function and belongs to the class of Collocation methods. This means that the domain of applicability of the method is the function space S_γ of all functions whose analytical continuation, $E(z)$, can be assumed to be of the form $E(z) = z^{-\gamma} G(z)$, for some fixed $\gamma > 0$, where G is analytic at infinity.

If the Laplace function $F \in S_\gamma$ the convergence of the Laguerre series expansion of the inverse function is guaranteed. However, due to the severe ill-posedness of the problem, as N grows the partial sum of the first N terms of the Laguerre series does not always provide a better approximation of the inverse function, because the condition error increases as N grows. As a consequence, the rate of convergence of the Laguerre series significantly affects software performance.

More precisely, the software selects the optimal value of N so the input required accuracy or the so-called maximum attainable accuracy is reached where N has to be less that $N_{\max} = 40$. If it occurs that the rate of convergence of the Laguerre series expansion is too slow it may happen that the maximum attainable accuracy (with $N_{\max} = 40$ terms) is greater than the input required accuracy. As described in [1], this is the case of Lt functions that have a singularity at infinity or on imaginary axis. Such situation is detected by the error indicator `flag = 1`.

In summary, `ReLaTIve` inherits the peculiarities of real inversion problem, this is why its performance may degrade (see for instance the Duffy's tests). For this reason, we introduce the controlled accuracy as stopping criterion: it is a safely-way of the software to handle intrinsic ill posedness of real inversion problem.

Acknowledgements The authors would like to thank the anonymous referees for the valuable comments and remarks.

References

1. Abate J., Choudhury G., Whitt W.: On the Laguerre method for numerically inversting Laplace transforms. INFORMS J. Comput. **8**(4), 413–427 (1996)
2. Abate, J., Valko, P.: Uniform Resource Locators (URL). Wolfram Information Center. http://library.wolfram.com/infocenter/MathSource/4738/ (2002)
3. Abate, J., Valko, P.: Uniform Resource Locators (URL). Wolfram Information Center. http://library.wolfram.com/infocenter/MathSource/5026/ (2003)
4. Abate, J., Valko, P.: Numerical Laplace inversion in rheological characterization. J. Non-Newton. Fluid Mech. **116**, 395–406 (2004)
5. Bjorck, A., Pereira, V.: Solution of Vandermonde systems of equations. Math. Comput. **112**(24), 893—903 (1970)
6. Binous, H.: Uniform Resource Locators (URL). Wolfram Information Center. http://library.wolfram.com/infocenter/MathSource/6557/ (2010)

7. Borgia, G.C., Brown, R.J.S., Fantazzini, P.: Uniform penalty inversion of multiexponential decay data II. J. Magn. Reson. **147**, 273–285 (2000)
8. Cohen A.M.: Numerical Methods for Laplace Transform Inversion. Springer (2007)
9. Cuomo S., D'Amore L., Murli A., Rizzardi, M.: Computation of the inverse Laplace transform based on a collocation method which uses only real values. J. Comput. Appl. Math. **198**, 98–115 (2007)
10. D'Amore, L., Laccetti, G., Murli, A.: An implementation of a Fourier series method for the numerical inversion of the Laplace transform. ACM Trans. Math. Softw. **25**(30), 279–305 (1999)
11. D'Amore L., Laccetti G., Murli, A.: Algorithm 796: a Fortran software package for the numerical inversion of the Laplace transform based on a Fourier series method. ACM Trans. Math. Softw. **25**, 306–315 (1999)
12. Davies, B., Martin, B.: Numerical inversion of the Laplace transform: a survey and comparison of methods. J. Comput. Phys. **33**, 1–32 (1979)
13. Duffy, D.G.: On the numerical inversion of Laplace transforms: comparison of three new methods on characteristic problems from applications. ACM Trans. Math. Softw. **19**(3), 333–359 (1993)
14. Fair, W. Jr.: Numerical Laplace transforms and inverse transforms in C#. http://www.codeproject.com/KB/recipes/LaplaceTransforms.aspx?msg=3150794 (2008)
15. Garbow, S., Giunta, G., Lyness, N.J., Murli, A.: Algorithm 662: a Fortran software package for the numerical inversion of a Laplace transform based on Week's method. ACM Trans. Math. Softw **54**, 163–170 (1988)
16. Giunta, G., Laccetti, G., Rizzardi, M.: More on weeks method for the numerical inversion of the Laplace trasform. Numer. Math. **193**, 193–200 (1988)
17. Giunta, G., Murli, A., Schmid, G.: An analysis of bilinear transform-polynomial methods of inversion of Laplace transforms. Numer. Math. **69**(1), 269–282 (1995)
18. Giunta, G., Murli, A., Schmid, G.: Error analysis of Rjabov algorithm for inverting Laplace transforms. Ric. Mat. **XLIV**(1), 207–219 (1995)
19. Halsted, D.J., Brown, D.E.: Zakian's technique for inverting Laplace transforms. Chem. Eng. J. **3**, 312–313 (1972)
20. Henrici, P.: Applied and Computational Complex Analysis, vol. 2. John Wiley & Sons (1977)
21. Higham, N.: Error analysis of the Björk–Pereira algorthm for solving Vandermonde systems. Numer. Math. **50**, 613–632 (1987)
22. Kryzhniy, V.V.: On regularization of numerical inversion of Laplace transforms. J. Inverse Ill-Posed Probl. **12**(3), 279–296 (2004)
23. Kryzhniy, V.V.: InvertLT. http://www-users.cs.umn.edu/~yelena/web/software.php. Accessed 2006
24. Lyness, N.J., Giunta, G.: A modification of the weeks method for the inversion of the Laplace tranform. Math. Comput. **47**, 313–322 (1987)
25. Mallet, A.: Uniform Resource Locators (URL). Wolfram Information Center. http://library.wolfram.com/infocenter/MathSource/2691/ (2000)
26. Membrez, J., Infelta, P.P., Renken, A.: Use of the Laplace transform technique for simple kinetic parameters evaluation. Application to the adsorption of a protein on porous beads. Chem. Eng. Sci. **51**(19), 4489–4498 (1996)
27. Murli, A., Rizzardi, M.: Algorithm 682: Talbot's method for the Laplace inversion problem. ACM Trans. Math. Softw. **16**, 347–371 (1990)
28. NAg Documentation: Uniform Resource Locators (URL). Numerical Algorithms Group Ltd. http://www.num-alg-grp.co.uk/numeric/Fl/manual/html/examples/source/c06lafe.f (1989)
29. Piessens, R., Branders, P.: Numerical inversion of the Laplace tranform usinge generalized Laguerre polynomials. Proc. IEEE **118**, 1517–1522 (1971)
30. Piessens, R.: A new numerical method for the inversion of the Laplace tranform. J. Inst. Math. Appl. **10**, 185–192 (1972)
31. Piessens, R., Huysmans, R.: Algorithm 619: automatic numerical inversion of the Laplace transform. ACM Trans. Math. Softw. **10**(3), 348–353 (1984)
32. Provencher, S.: CONTIN. http://s-provencher.com/pages/contin.shtml. Accessed 1982
33. Rjabov, V.M.: On the numerical inversion of the Laplace tranform. Vestn. Leningr. Univ. **7**, 177–185 (1974)

34. Sykora, S.: Bortolotti, V., Fantazzini, P.: PERFIDI: parametrically enabled relaxation filters with double and multiple inversion. Magn. Reson. Imaging **25**, 529–532 (2007)
35. Spinelli, R.A.: Numerical inversion of a Laplace transform. SIAM J. Numer. Anal. **3**(4), 636–649 (1966)
36. Stehfest, H.: Algorithm 368: numerical inversion of Laplace transform. Commun. ACM **13**, 47–49 (1970)
37. Saravanathamizhana, R., Paranthamana, R., Balasubramaniana, N., Ahmed Bashab, C.: Residence time distribution in continuous stirred tank electrochemical reactor. Chem. Eng. J. **142**, 209–216 (2008)
38. Thornhill, N.F., Patwardhan, S.C., Shah, S.L.: A continuous stirred tank heater simulation model with applications. J. Process Control **18**, 347–360 (2008)
39. Valko, P.: Numerical Inversion of Laplace Transform. http://www.pe.tamu.edu/valko/Nil/ (2002)
40. Valko, P., Vaida, S.: Inversion of noise-free Laplace transforms: towards a standardized set of test problems. Inverse Probl. Eng. **10**, 467–483 (2002)
41. Weeks, W.: Numerical inversion of the Laplace tranform using Laguerre functions. J. ACM **13**, 419–429 (1966)