Scientific Research Publishing

Applied Mathematics, 2022, 13, 401-418
https://www.scirp.org/journal/am
ISSN Online: 2152-7393
ISSN Print: 2152-7385

The Numerical Inversion of the Laplace Transform in a Multi-Precision Environment

Colin L. Defreitas, Steve J. Kane

School of Physics Astronomy and Mathematics, University of Hertfordshire, Hertfordshire, UK
Email: c.defreitas2@herts.ac.uk, s.j.kane@herts.ac.uk

How to cite this paper: Defreitas, C.L. and Kane, S.J. (2022) The Numerical Inversion of the Laplace Transform in a Multi-Precision Environment. *Applied Mathematics*, **13**, 401-418.
https://doi.org/10.4236/am.2022.135027

Received: May 4, 2022
Accepted: May 20, 2022
Published: May 23, 2022

Abstract

This paper examines the performance of five algorithms for numerically inverting the Laplace transform, in standard, 16-digit and multi-precision environments. The algorithms are taken from three of the four main classes of numerical methods used to invert the Laplace transform. Because the numerical inversion of the Laplace transform is a perturbed problem, rounding errors which are generated in numerical approximations can adversely affect the accurate reconstruction of the inverse transform. This paper demonstrates that working in a multi-precision environment can substantially reduce these errors and the resulting perturbations exist in transforming the data from the *s*-space into the time domain and in so doing overcome the main drawback of numerically inverting the Laplace transform. Our main finding is that both the Talbot and the accelerated Gaver functionals perform considerably better in a multi-precision environment increasing the advantages of using Laplace transform methods over time-stepping procedures in solving diffusion and more generally parabolic partial differential equations.

Keywords

Laplace Transform, Perturbation, Numerical Inversion, Multi-Precision, Stehfest, Talbot

1. Introduction

This paper examines the performance of five algorithms for numerically inverting the Laplace transform, in standard 16-digit and multi-precision environments. The algorithms, whose derivations are outlined in Section 5, are taken from three of the four main classes of numerical methods used to invert the Laplace transform [1].

The Abate-Valko [1] and Logan schemes [2] belong to the class of inversion algorithms which deform the Bromwich contour [3]. They are closely related

versions of this approach as they both use Talbot's method for deformation of the contour [4]. Logan, however, chooses an exponential transform while Abate-Valko extends the original Talbot formulation expressing the contour in trigonometric form.

The Stehfest and Salzer-Gaver algorithms [5], are again two closely related schemes based on the acceleration of the Gaver functional [6]. Stehfest applied a modified Salzer acceleration scheme [7] onto the Gaver functional simplifying this result to yield one of the most widely used algorithms for inverting the Laplace transform. We find, however, that when we implement a direct application of the Salzer acceleration scheme onto the Gaver functional, (Salzer-Gaver), with Stehfest's modifications, we do not obtain the same results as those produced by the Stehfest scheme. We conclude that Stehfest's simplification process is at least in part responsible for the differences in performance of these two versions.

Finally, we examine the Fourier series method [8], which expresses the inversion integral as a Fourier series and then uses the trapezium rule to evaluate the integral numerically. The Fourier series method differs from the other four algorithms as no acceleration scheme is used to force convergence. As such, this is only used in a standard 16 digit precision environment and is compared with the four other schemes using standard precision.

2. The Laplace Transform

The Laplace transform is an integral transform defined as follows:

Let $f(t)$ be defined for $t \geq 0$, then the Laplace transform of $f(t)$ is given by,

$$\mathcal{L}\{f(t)\} = \int_0^\infty f(t)\mathrm{e}^{-st}\mathrm{d}t \tag{1}$$

Thus $\mathcal{L}\{f(t)\}$ is a function of s denoted as $F(s)$. The Laplace transform can be shown to exist for any function which can be integrated over any finite interval $0 < t < l$ for $l > 0$, and for which $f(t)$ is of exponential order, *i.e.*

$$|f(t)| < M\mathrm{e}^{at} \tag{2}$$

as $t \to \infty$, where $M > 0$ is a finite real number and a is a small real positive number.

Analytically the inverse Laplace transform is usually obtained using the techniques of complex contour integration with the resulting set of standard transforms presented in tables [9].

However, using the Laplace transform can generate data in the Laplace domain which is not easily invertible to the real domain by analytical means. Thus numerical inversion techniques have to be used to convert the data from the s-space to the time domain.

3. The Inverse Laplace Transform, Perturbation and Multi-Precision

The recovery of the function $f(t)$ is via the inverse Laplace transform which is

most commonly defined via the Bromwich contour integral,

$$L^{-1}\{F(s)\} = f(t) = \frac{1}{2\pi i}\int_{\alpha-i\infty}^{\alpha+i\infty} f(s)\mathrm{e}^{st}\mathrm{d}s \tag{3}$$

such that $\alpha \in \mathbf{R}$. The inversion integral is widely known to be highly perturbed [10] [11] [12]. This is due to the e^{st} term in the integral which amplifies small changes in the input data. Hence all numerical schemes are vulnerable to this perturbation and this has to be taken into account when using the various algorithms to invert the Laplace transform. This suggests that working in a multi-precision environment can act to reduce errors and the resulting perturbations which exist in the transforming the data from the *s*-space into the time domain.

As Abate-Valko noted [1], "In the traditional development of the inversion methods, most of the effort was directed at controlling round-off errors. This is because the process is numerically unstable in a fixed-precision, computing environment. The problem is that as the user tries to increase the accuracy there comes a point where the round off error causes the error to increase dramatically".

In fact Abate-Valko got further and made the claim that "for our purposes, we add the proviso that values of the transform can be computed to any desired precision as a function of the complex variable".

This suggests that working in a multi-precision environment can act to reduce errors and the resulting perturbations which exist in transforming the data from the *s*-space into the time domain.

4. The Algorithms

Large parts of this section are taken from our earlier work [13]. This is necessary to provide sufficient background on the derivation of the algorithms and their performance. However, we extend this work to include comments on Salzer acceleration and the subsequent Salzer Gaver scheme's derivation and compare this to the Stehfest inversion scheme.

However this has been extended to include analysis of the Salzer acceleration scheme on the Gaver functional. We also comp There are over 100 algorithms available for inverting the Laplace Transform with numerous comparative studies, examples include, Duffy [14], Narayanan and Beskos [15], Cohen [10] and perhaps the most comprehensive by Davies and Martin [16]. However for the purposes of this investigation we apply our tests using "Those algorithms that have passed the test of time" [1]. These fall into four groups,

1) Fourier series expansion.
2) Combination of Gaver Functionals.
3) Laguerre function expansion.
4) Deformation of the Bromwich contour.

4.1. The Fourier Series Method

In their survey of algorithms for inverting the Laplace transform, Davies and

Martin [16] note that the Fourier series method without accelerated convergence gives good accuracy on a wide variety of functions. Since the Laplace Transform is closely related to the Fourier transform it is not surprising that inversion methods based on a Fourier series expansion would yield accurate results. In fact the two sided Laplace transform can be derived from the Fourier transform in the following way. We can define the Fourier transform as

$$\mathcal{F}\left\{f(t)\right\}=\int_{-\infty}^{\infty}f(t)\mathrm{e}^{-2\pi ivt}\mathrm{d}t \tag{4}$$

Then letting $v=2\pi\nu$ we have

$$\mathcal{F}\left\{f(t)\right\}=\int_{-\infty}^{\infty}f(t)\mathrm{e}^{-ivt}\mathrm{d}t \tag{5}$$

This Fourier transform exists provided $f(t)$ is an absolutely integrable function, *i.e.*

$$\int_{-\infty}^{\infty}\left|f(t)\right|\mathrm{d}t<\infty \tag{6}$$

As many functions do not satisfy condition (6), $f(t)$ is multiplied by the exponential dampening factor e^{-ut} thus

$$\mathcal{F}\left\{f(t)\mathrm{e}^{-ut}\right\}=\int_{-\infty}^{\infty}f(t)\mathrm{e}^{-ivt}\mathrm{e}^{-ut}\mathrm{d}t \tag{7}$$

and letting $s=u+iv$ we obtain the two sided Laplace Transform of $f(t)$ as

$$\mathcal{F}\left\{f(t)\mathrm{e}^{-ut}\right\}=\mathcal{L}\left\{f(t)\right\}=\int_{-\infty}^{\infty}\mathrm{e}^{-st}f(t)\mathrm{d}t \tag{8}$$

Le Page [17] notes that the integral given by (8) can be written in two parts as follows:

$$\int_{-\infty}^{\infty}\mathrm{e}^{-st}f(t)\mathrm{d}t=\int_{-\infty}^{0}\mathrm{e}^{-st}f(t)\mathrm{d}t+\int_{0}^{\infty}\mathrm{e}^{-st}f(t)\mathrm{d}t \tag{9}$$

The second term on the RHS in the above expression is referred to as the one sided Laplace transform or simply the Laplace transform. Thus, *s* is defined as a complex variable in the definition of the Laplace transform.

As before the inverse Laplace transform is given as:

$$f(t)=\frac{1}{2\pi i}\int_{u-i\infty}^{u+i\infty}\mathrm{e}^{st}F(s)\mathrm{d}s \tag{10}$$

With $s=u+iv$ in (10) this leads to the result

$$f(t)=\frac{2\mathrm{e}^{ut}}{\pi}\int_{0}^{\infty}\left[\mathrm{Re}\left\{F(u+iv)\right\}\cos(vt)-\mathrm{Im}\left\{F(u+iv)\right\}\sin(vt)\right]\mathrm{d}v \tag{11}$$

[8], which can be replaced by the cosine transform pair

$$\mathrm{Re}\left\{F(u+iv)\right\}=\int_{0}^{\infty}\mathrm{e}^{-ut}f(t)\cos(vt)\mathrm{d}t \tag{12}$$

$$f(t)=\frac{2\mathrm{e}^{ut}}{\pi}\int_{0}^{\infty}\mathrm{Re}\left\{F(u+iv)\right\}\cos(vt)\mathrm{d}v \tag{13}$$

or by the sine transform pair

$$\mathrm{Re}\left\{F(u+iv)\right\}=-\int_{0}^{\infty}\mathrm{e}^{-ut}f(t)\sin(vt)\mathrm{d}t \tag{14}$$

$$f(t) = -\frac{2e^{ut}}{\pi}\int_0^{\infty} \mathrm{Im}\{F(u+iv)\}\sin(vt)\,dv \tag{15}$$

Dubner and Abate [18] applied a trapezoid rule to (13) resulting in the Fourier series approximation,

$$f(t) \approx \frac{2e^{ut}}{T}\left[\frac{1}{2}F(u) + \sum_{k=1}^{\infty}\mathrm{Re}\left\{F\left(u+\frac{k\pi i}{T}\right)\right\}\cos\left(\frac{k\pi t}{T}\right)\right] \tag{16}$$

where $f(t)$ is expanded in the interval $0 \le t < T$. For faster computation Simon *et al.* [19] proposed the following version of (16).

$$f(t) \approx \frac{e^{ut}}{t}\left[\frac{1}{2}F(u) + \sum_{k=1}^{\infty}\mathrm{Re}\left\{F\left(u+\frac{k\pi i}{t}\right)\right\}(-1)^k\right] \tag{17}$$

This series can be summed much faster than (16) as there are no cosines to compute [20]. This algorithm is relatively easy to implement with u being the only real varying parameter.

However, as Crump [8] points out, for the expression in (17) the transform $F(s)$ must now be computed for a different set of s-values for each distinct t. Since this type of application occurs often in practice in which the numerical computations of $F(s)$ is itself quite time consuming this may not be an economical inversion algorithm to use. These drawbacks to some extent can be overcome by using the fast Fourier transform techniques in [20] [21].

Crump [8] also extends this method to one with faster convergence by making use of the already computed imaginary parts. There are several other acceleration schemes for example, those outlined by Cohen [10], however these acceleration methods in general require the introduction of new parameters which for the purposes of this investigation we wish to avoid.

4.2. The Stehfest Algorithm

Davies and Martin [16] cite the Stehfest [5] algorithm as providing accurate results on a wide variety of test functions. Since that time, this algorithm has become widely used for inverting the Laplace Transform, being favoured due its reported accuracy and ease of implementation.

Here we give a brief overview of the evolution of the algorithm from a probability distribution function to the Gaver functional whose asymptotic expansion leads to an acceleration scheme which yields the algorithm in its most widely used form.

Gaver [6] investigated a method for obtaining numerical information on the time dependent behavior of stochastic processes which often arise in queuing theory. The investigation involved examining the properties of the three parameter class of density functions namely

$$p_{n,m}(a,t) = \frac{(n+m)!}{n!(m-1)!}\left(1-e^{-at}\right)^n ae^{-mat} \tag{18}$$

with $n, m \in \mathbb{N}$.

After the binomial expansion of the term $\left(1-\mathrm{e}^{-at}\right)^n$, Gaver went on to find the expectancy $E\left[f\left(T_{n,m}\right)\right]$, where $T_{n,m}$ is the random variable with density (18). From this Gaver was able to express the inverse Laplace transform in terms of the functional

$$f_{n,m}(t)=\frac{\ln 2}{t}\frac{(n+m)!}{n!(m-1)!}\sum_{j=0}^{n}\binom{n}{k}(-1)^k F\left((k+m)\frac{\ln 2}{t}\right) \tag{19}$$

with certain conditions on n and m, Gaver makes $n=m$ and Expresses (19) as

$$f_n(t)=\frac{\ln 2}{t}\frac{(2n)!}{n!(n-1)!}\sum_{k=0}^{n}\binom{n}{k}(-1)^k F\left((k+n)\frac{\ln 2}{t}\right) \tag{20}$$

While the expression in (20) can be used to successfully invert the Laplace transform for a large class of functions its rate of convergence is slow [9] [10]. However Gaver [6] has shown that (20), with $t=\frac{\ln 2}{a}$ has the asymptotic expansion

$$f_n(t)\approx f\left(\frac{\ln 2}{a}\right)+\frac{\alpha_1}{n}+\frac{\alpha_2}{n^2}+\frac{\alpha_3}{n^3}+\cdots \tag{21}$$

where the α_j's are constant coefficients in the asymptotic series. Hence (21) in the limit converges to

$$\lim_{n\to\infty} f_n(t)=f\left(\frac{\ln 2}{a}\right)$$

For the conditions on m and n and justification for the substitution for a referred to above see Gaver [6]. This asymptotic expansion provides scope for applying various acceleration techniques enabling a more viable application of the basic algorithm.

Much of the literature alludes to the fact that a Salzer [7] acceleration scheme is used on the Gaver functional in (20) which results in the Stehfest algorithm. In fact Stehfest's approach was a little more subtle than a direct application of the Salzer acceleration.

4.2.1. Using Salzer Acceleration

The Salzer acceleration scheme makes use of the "Toeplitz limit theorem" [7], "this concerns the convergence of a transformation of a sequence ζ_s where the $(n+1)$ th member of the transformed sequence is a weighted mean of the first $(n+1)$ terms"

$$\overline{S}_n=\sum_{k=0}^{n}\mu_{nk}\cdot S_k \tag{22}$$

Here $\overline{S}_n$ is the transformed sequence and S_k the original sequence which for our purposes $=f_n(t)$ in (20). The Salzer means are given by

$$\mu_{nk}=(-1)^{n+k}\frac{(1+k)^n}{n!}\binom{n}{k} \tag{23}$$

For the sake of compatibility with (22) we make the change $k\to i$ and

$n \to k$ in (20). With this change of variables we also write

$$\frac{(2k)!}{k!(k-1)!} = \frac{k(2k)!}{k!k!}$$

This allows the sum to be taken from $k=0$ to n without $(0-1)!$ in the denominator of (20). So with Salzer acceleration we can express the Salzer-Gaver algorithm as

$$f_n(t) = \frac{\ln 2}{t}\sum_{k=0}^{n}(-1)^{n+k}\frac{(k+1)^n}{k!(n-k)!}\frac{k(2k)!}{k!k!}\sum_{i=0}^{k}\frac{k!}{i!(k-i)!}(-1)^i F\left\{\frac{(k+i)\ln 2}{t}\right\} \quad (24)$$

4.2.2. Stehfest's Acceleration Scheme

For the purposes of following Stehfest's derivation it will be convenient to rewrite (20) as

$$f_n(t) = F_n = \frac{(2n)!a}{n!(n-1)!}\sum_{k=0}^{n}\binom{n}{k}(-1)^k F\left((k+n)a\right) \quad (25)$$

Stehfest [5] begins by supposing we have N values for $F\left[(k+n)a\right]$ with $F(a)$, $F(2a)$, $F(3a)$, $\cdots$ $F(Na)$ for N even. Using (25) we can then determine $\frac{N}{2}$ values $F_1, F_2, \cdots, F_{N/2}$. Now each of these $\frac{N}{2}$ values satisfy the asymptotic series in (21) with the same coefficients.

As Stehfest [5] points out, the a_j's are the same for each of the above expansions and by using a suitable linear combination the first $(\frac{N}{2}-1)$ error terms in (21) can be eliminated. That is

$$f\left(\frac{\ln 2}{a}\right) = \sum_{n=1}^{\frac{N}{2}} a_n F_{\left(\frac{n}{2}+i-1\right)} + O\left(\frac{1}{N^{\frac{N}{2}}}\right) \quad (26)$$

which may be achieved by selecting the coefficients to satisfy

$$\sum_{n=1}^{\frac{N}{2}} a_n \frac{1}{\left(\frac{N}{2}+1-n\right)^k} = \delta_{k,0}, \quad k = 1, \cdots, N/2-1 \quad (27)$$

which produce the same coefficients as the Salzer acceleration scheme used in (22). In fact for any n, Stehfest generates the required coefficients using what is in effect a modified Salzer acceleration scheme giving

$$a_n = \frac{(-1)^{n-1}}{\left(\frac{N}{2}\right)!}\binom{\frac{N}{2}}{n} n\left\{\left(\frac{N}{2}+1-n\right)^{\frac{N}{2}-1}\right\} \quad (28)$$

Finally, Stehfest substitutes these results into (26) and gets the inversion formula

$$f(t) \approx \frac{\ln 2}{t}\sum_{j=1}^{N} A_j F\left(\frac{j\ln 2}{t}\right) \quad (29)$$

for N even and

$$A_j = (-1)^{\frac{N}{2}+j} = \sum_{k=\left\lfloor \frac{j+1}{2} \right\rfloor}^{\min\left(j,\frac{N}{2}\right)} \frac{k^{\frac{N}{2}}(2k)!}{\left(\frac{N}{2}-k\right)!k!(k-1)!(j-k)!(2k-j)!} \tag{30}$$

However, a direct application of the modified Salzer acceleration scheme in (28) onto the Gaver functional in (25) does not produce the same results for the expression in (30) so they are not exactly equal to each other.

To show this we consider the function $\sin(t)$ whose Laplace transform is

$$\frac{1}{s^2+1}$$

The eight weights produced by the Salzer acceleration for $n=8$ are exactly the same for $n=18$ in Stehfest's modified Salzer acceleration scheme in (28).

However, Table 1 shows that for $\sin(t)$ with these same weights, the Salzer-Gaver scheme produces different results when compared to Stehfest's scheme in (30). We believe this is due to Stehfest's simplification of the Salzer-Gaver scheme to the expression in (30).

This simplification was necessary because as we show in our results in Section 6, Stehfest's final expression in (30) is faster and works better in standard double precision. As the algorithm was developed in 1970, this would be far more efficient when taking into consideration the computing power available at the time. Again as we show in Section 6, a direct application of a Salzer acceleration scheme onto the Gaver functional is only advantageous in a multi-precision environment.

4.3. The Talbot Algorithm

Equations (4) to (8) showed that the Laplace transform can be seen as a Fourier transform of the function

$$\mathrm{e}^{-ut} f(t),\ t>0 \tag{31}$$

i.e.

$$F\left\{\mathrm{e}^{-ut} f(t)\right\} = \mathcal{L}\left\{f(t)\right\} = F(s) \tag{32}$$

Table 1. Salzer-Gaver stehfest for $\sin(t)$.

t	Stehfest	Salzer-Gaver
5	0.89	1.02
10	0.08	0.18
15	0.002	0.03
20	0.03	0.02
25	0.001	0.004
30	0.001	0.004

Hence the Fourier transform inversion formula can be applied to recover the function thus:

$$F^{-1}\{F(s)\} = \mathrm{e}^{-ut} f(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(s)\mathrm{e}^{ivt}\mathrm{d}v \tag{33}$$

as $s = u + iv$ we have that $\mathrm{d}s = i\mathrm{d}v$ and so

$$f(t) = \frac{1}{2\pi i}\int_{u-i\infty}^{u+i\infty} F(s)\mathrm{e}^{st}\mathrm{d}s \tag{34}$$

This result provides a direct means of obtaining the inverse Laplace transform. In practice the integral in (34) is evaluated using a contour

$$\frac{1}{2\pi i}\int_{B} \mathrm{e}^{st} F(s)\mathrm{d}s \tag{35}$$

with B here denoting the Bromwich contour [3]. The contour is chosen so that it encloses all the possible singularities of $F(s)$. The idea of the contour is introduced so that the Cauchy residue theorem can be used to evaluate the integral.

However, when $f(t)$ is to be calculated using numerical quadrature it may be more appropriate to devise a new contour. To ensure that (35) is integrable we may wish to control the growth of the magnitude of the integrand e^{st} by moving the contour to the left and so giving the real part of s a large negative component [1] [22].

However, the deformed contour must not be allowed to pass through any singularities of $F(s)$. This is to ensure that the transform is analytic in the region to the right of B.

4.3.1. Derivation of the Fixed Talbot Contour

In the derivation that follows [1] and [22] are used as the primary basis for extending the explanation of the derivation of the Talbot algorithm for inverting the Laplace Transform.

Abate-Valko [1] began with the Bromwich inversion integral along the Bromwich contour B with the transform

$$F(s) = \frac{1}{s^{\alpha}}, \ \alpha > 0 \tag{36}$$

So $f(t)$ can be expressed as

$$f(t) = \frac{1}{2\pi i}\int_{B} \mathrm{e}^{t(s-a\ln s)}\mathrm{d}s \tag{37}$$

with $a = \frac{\alpha}{t}$ in (37). As Abate-Valko [1] indicated, numerically evaluating the integral in (37) is difficult due to the oscillatory nature of the integrand.

However, this evaluation can be achieved by deforming the contour B into a *path of constant phase* thus eliminating the oscillations in the imaginary component. These paths of constant phase are also paths of steepest decent for the real part of the integrand [1] [22] [23].

There are in general a number of contours for which the imaginary component remains constant so we choose one on which the real part attains a maxi-

mum on the interior (a saddle point) and this occurs at $g'(s)=0$ at some point on the contour. At these saddle points the $\mathrm{Im}\{g(s)\}=0$ [17]. Where

$$g(s)=s-a\ln s \tag{38}$$

in (37). Thus we have

$$g'(s)=1-\frac{a}{s} \tag{39}$$

So the stationary point occurs when $s=a$.

With $s=u+iv$ we have

$$\mathrm{Im}\{u+iv-a\ln(u+iv)\}=0 \tag{40}$$

Expressing $u+iv$ as $Re^{i\theta}$ we have

$$\mathrm{Im}\{(u-a\ln R)+i(v-a\theta)\}=0 \tag{41}$$

then

$$v=a\theta \tag{42}$$

and as

$$\tan(\theta)=\frac{v}{u} \tag{43}$$

Then

$$u=a\theta\cot(\theta) \tag{44}$$

[1].

With $v=a\theta$ then s can be parameterized to (44) Talbots contour

$$s(\theta)=a\theta(\cot(\theta)+i),\quad -\pi<\theta<+\pi \tag{45}$$

[4].

4.3.2. Conformal Mapping of the Talbot Contour

While the above parametrization can be used as a basis for inverting the Laplace transform we proceed with the algorithm's development via a convenient conformal mapping as follows.

$$\cot\theta=\frac{i\left(e^{i\theta}+e^{-i\theta}\right)}{e^{i\theta}-e^{-i\theta}} \tag{46}$$

Then

$$\theta\cot\theta+i\theta=\frac{2i\theta}{1-e^{-2i\theta}} \tag{47}$$

with $z=2i\theta$ (47) is equal to

$$\frac{z}{1-e^{-z}} \tag{48}$$

Hence (45) becomes,

$$\frac{az}{1-e^{-z}} \tag{49}$$

The function

$$s(z)=\frac{az}{1-e^{-z}} \tag{50}$$

maps the closed interval $M=[-2\pi i,2\pi i]$ on the imaginary z-plane onto the curve L in the s plane giving the integral,

$$f(t)=\frac{1}{2\pi i}\int_L F(s)e^{st}ds \tag{51}$$

For the details of this transformation one can refer the study of Logan [2].

Next we follow the procedure as adopted by Logan [2] for numerically integrating (51). With the change of variable (51) becomes

$$f(t)=\frac{1}{2\pi i}\int_M F(s(z))e^{s(z)t}s'(z)dz \tag{52}$$

where

$$s'(z)=\frac{-a\left(ze^{-z}+e^{-z}-1\right)}{\left(e^{-z}-1\right)^2} \tag{53}$$

For convenience we write,

$$f(t)=\frac{1}{2\pi i}\int_M I(z,t)dz \tag{54}$$

where

$$I(z,t)=F(s(z))e^{s(z)t}s'(z) \tag{55}$$

The integral in (54) is then rotated by $\frac{\pi}{2}$ so the interval of integration is now real and becomes $[-2\pi,2\pi]$ and then we use the trapezoid rule with n odd and $w=-iz$ to obtain

$$f(t)\cong\frac{1}{n}\left\{I(2\pi i)+T(-2\pi i)+2\sum_{j=1}^{n-1}I(iw_j)\right\} \tag{56}$$

where

$$w_j=2\pi\left(\frac{2j}{n}-1\right) \tag{57}$$

and we note that $I(2\pi i)=I(-2\pi i)=0$ [2].

4.3.3. Valko

Abate-Valko [1] deformed the Bromwich contour using the Talbot path which has the form,

$$s(h)=rh\left(\cot(h)+i\right),\quad -\pi<h<\pi \tag{58}$$

Which is the same as (47) with $a=r$ and $h=\theta$

So we have

$$s'(h)=ir\left(1+ir(h)\right) \tag{59}$$

where,

$$r(h) = h + (h\cot(h) - 1)\cot(h) \tag{60}$$

Then from (52) we find,

$$f(t) = \frac{r}{\pi}\int_0^{\pi} \mathrm{Re}\left[e^{ts(h)}F(s(h))(1+ir(h))\right]\mathrm{d}h \tag{61}$$

They then approximate the value of the integral in (63) by using the trapezoidal rule with step size $\frac{\pi}{m}$ and $h_k = \frac{k\pi}{m}$,

$$f(t,M) = \frac{r}{m}\left[\frac{1}{2}F(r)\exp(rt) + \sum_{k=1}^{M-1}\mathrm{Re}\left[e^{ts(h_k)}F(s(h_k))(1+ir(h_k))\right]\right] \tag{62}$$

Based on numerical experiments, Abate-Valko then fixed the parameter r to the value,

$$r = \frac{2M}{5t}. \tag{63}$$

[1]. We also use this value for a in Logan's transformation.

5. Results

We tested the five algorithms on the functions listed in Table 2 and Table 3

Table 2. Test functions.

Function No.	$f(s)$	$f(t)$
1	$\frac{1}{1+s^2}$	$\sin(t)$
2	$\frac{1}{(s+1)^2}$	te^{-t}
3	$\frac{1}{s^2}$	t
4	$\frac{1}{\sqrt{s}}$	$\frac{1}{\sqrt{\pi t}}$
5	$\frac{\ln s}{s}$	$-(\ln t + \gamma)$
6	$\frac{1}{s}$	1
7	$\frac{1}{\sqrt{s^2+1}}$	$J(0,t)$
8	$\frac{e^s K(1,s)}{s}$	$\sqrt{t(t+2)}$
9	$\frac{1}{s+0.5}$	$e^{-\frac{t}{2}}$
10	$\frac{1}{(s+0.2)^2+1}$	$e^{-0.2t}\sin(t)$
11	$\arctan\left(\frac{1}{s}\right)$	$\frac{\sin(t)}{t}$

Table 3. Test functions continued.

Function No.	$f(s)$	$f(t)$
12	$\frac{1}{\sqrt{s}+\sqrt{s+1}}$	$\frac{1-\mathrm{e}^{-t}}{2\sqrt{\pi t^3}}$
13	$\frac{1}{\sqrt{s}\left(1+\sqrt{s}\right)}$	$\mathrm{e}^{t}\mathrm{erfc}\left(\sqrt{t}\right)$
14	$\mathrm{e}^{-2\sqrt{s}}$	$\frac{\mathrm{e}^{-\frac{1}{t}}}{\sqrt{\pi t^3}}$
15	$\frac{\mathrm{e}^{-\frac{1}{4s}}}{s^{\frac{3}{2}}}$	$\frac{2\sin\left(\sqrt{t}\right)}{\pi}$
16	$\log\left(1+\frac{1}{s}\right)$	$\frac{1-\mathrm{e}^{-t}}{t}$
17	$\frac{\arccos(s-1)}{\sqrt{s(s-2)}}$	$\mathrm{e}^{t}K(0,t)$
18	$\frac{\mathrm{e}^{-\frac{1}{s}}}{\sqrt{s}}$	$\frac{\cos\left(2\sqrt{t}\right)}{\sqrt{\pi t}}$
19	$\frac{1}{\sqrt{s+\sqrt{s^2+1}}}$	$\frac{\sqrt{2}\sin(t)}{2t^{\frac{3}{2}}\pi}$

below. Functions 1 - 11 and 18 are taken from the 16 functions tested by Davies and Martin [16]. The remaining functions are selected from those tested by Abate-Valko [1].

The first set of tests were carried out using 16 digits double precision. These results are shown in **Table 4**. The Fourier, Logan and Abate-Valko schemes were run with weights $M=50$, $M=100$ and $M=200$, however for brevity we include only the result for $M=200$.

For the Stehfest and the Salzer-Gaver algorithms best results were obtained with weights of $M=16$ and $M=8$ respectively. This is in keeping with Stehfest's observations on the instability of this method as M increases above an optimal level [5].

In multi-precision, the number of precision digits N for Abate-Valko were set equal to $N=M$ [1] and for the Slazer-Gaver, and Stehfest schemes, best results were obtained when the number of precision digits was set equal to $N=2M$.

$$L=\sqrt{\sum_{i=1}^{30}\frac{\left[f(t_i)-\tilde{f}(t_i)\right]^2}{30}} \tag{64}$$

And

Table 4. Standard double precision.

Function	Fourier		Logan		Valko		Stehfest		Salzer-Gaver	
	L	L_e	L	L_e	L	L_e	L	L_e	L	L_e
1	1.5 (−4)	2.8 (−4)	3.7 (−9)	1.2 (−11)	3.0 (−11)	9.4 (−14)	1.4 (−3)	2.6 (−5)	2.0 (−2)	9.1 (−4)
2	6.1 (−4)	1.4 (−4)	1.8 (−9)	2.7 (−10)	1.0 (−11)	3.9 (−14)	8.9 (−6)	8.9 (−6)	2.8 (−6)	3.2 (−6)
3	1.2 (−3)	1.2 (−3)	7.5 (−9)	2.3 (−11)	4.4 (−11)	1.4 (−13)	7.0 (−8)	7.1 (−8)	1.1 (−8)	1.1 (−8)
4	7.3 (−2)	8.4 (1.0)	7.4 (−9)	2.3 (−11)	4.8 (−11)	1.7 (−13)	5.4 (−8)	6.2 (−7)	2.8 (−6)	3.2 (−6)
5	6.8 (−2)	6.9 (−2)	4.1 (−11)	1.2 (−11)	6.4 (−12)	2.2 (−14)	2.2 (−8)	1.4 (−7)	2.3 (−5)	8.5 (−5)
6	6.1 (−4)	6.1 (−4)	7.5 (−9)	2.3 (−11)	6.5 (−11)	2.1 (−13)	0.0 (0)	0.0 (−)	8.7 (−15)	4.8 (−14)
7	2.8 (−4)	3.4 (−4)	Fail	Fail	Fail	Fail	1.9 (−2)	6.7 (−3)	1.7 (−2)	5.1 (−3)
8	Fail	Fail	1.2 (−8)	3.8 (−11)	8.1 (−11)	2.7 (−13)	9.2 (−8)	1.2 (−7)	6.3 (−4)	6.3 (−4)
9	5.7 (−4)	4.7 (−4)	4.6 (−9)	1.6 (−11)	3.3 (−11)	1.2 (−13)	1.2 (−6)	4.7 (−6)	1.9 (−8)	2.8 (−7)
10	6.1 (−4)	1.9 (−4)	3.1 (−9)	9.5 (−12)	3.0 (−11)	9.7 (−14)	9.2 (−3)	4.8 (−3)	5.2 (−4)	2.9 (−5)
11	2.8 (−4)	3.8 (−4)	5.9 (−9)	1.8 (−11)	3.4 (−11)	1.2 (−13)	7.4 (−3)	2.7 (−3)	6.5 (−3)	1.9 (−3)

$$L_e = \sqrt{\frac{\sum_{i=1}^{30}\left[f(t_i)-\tilde{f}(t_1)\right]^2 \mathrm{e}^{-t_i}}{\sum_{i=1}^{30}\mathrm{e}^{-t_i}}} \tag{65}$$

where $f(t)$ is the analytical solution and $\tilde{f}(t)$ is the numerical solution. Hence L is the root-mean-square error and L_e is the same as L but weighted by the factor e^{-t} [14].

All computations were done using a 64-bit operating system with an Intel(R) Core(TM) i7-855ou CPU processor. The algorithms were implemented in Maple 2018 using the Maple's digits command to set the required precision.

5.1. Standard Double Precision

Table 4 and Table 5 show that when compared with the other four algorithms, the Fourier series method performs with the least accuracy on all the functions tested. It also fails to reconstruct functions 8, 15, 17 and 18, with poor results for functions 4, 5 and 12.

However, for the functions which it successfully reconstructs it does so with a RMS accuracy of between L = 3.6(−5) and 1.2(−2). We believe that this scheme will improve greatly when an acceleration scheme is applied. This is an issue we intend to investigate in our future work.

With the exception of function 7, $J_0(t)$, Logan's algorithm successfully inverts all the functions given in Table 2 and Table 3 with very good accuracy. We found that in SDP best results are obtained by equating $a=1$ in (50). Table 3 and Table 4 show that for these functions the RMS error varies between 3.6(−8) to 8.4(−12).

Except for function 7 the $J_0(t)$, the Abate-Valko scheme successfully inverts all the functions in Table 3 and Table 4. Moreover, it does so with greater

Table 5. Standard double precision continued.

Function	Fourier		Logan		Valko		Stehfest		Salzer-Gaver	
	L	L_e	L	L_e	L	L_e	L	L_e	L	L_e
12	2.0 (−1)	4.2 (−1)	3.1 (−9)	9.5 (−12)	1.6 (−11)	6.2 (−14)	6.2 (−8)	2.0 (−8)	8.2 (−8)	1.4 (−6)
13	3.1 (−3)	1.6 (−3)	3.7 (−9)	1.2 (−11)	2.7 (−11)	8.9 (−14)	8.6 (−7)	4.0 (−7)	3.3 (−7)	3.7 (−7)
14	6.2 (−5)	3.6 (−5)	3.6 (−8)	1.2 (−7)	6.2 (−12)	1.3 (−13)	1.3 (−6)	1.7 (−5)	1.6 (−6)	2.2 (−5)
15	Fail	Fail	3.3 (−9)	1.0 (−11)	3.9 (−11)	1.2 (−13)	5.7 (−7)	1.0 (−7)	7.9 (−6)	1.9 (−5)
16	5.8 (−4)	4.3 (−4)	5.2 (−9)	1.6 (−11)	2.5 (−12)	9.1 (−14)	2.2 (−8)	1.4 (−7)	4.9 (−7)	8.8 (−6)
17	Fail	Fail	1.2 (−8)	3.9 (−11)	7.3 (−11)	2.5 (−13)	6.0 (−8)	3.0 (−6)	2.6 (−6)	1.1 (−5)
18	Fail	Fail	2.7 (−9)	8.4 (−12)	1.7 (−11)	6.3 (−14)	2.9 (−5)	1.6 (−6)	5.9 (−6)	3.4 (−6)

Table 6. Multi-Precision $N = 200$.

Function	Logan		Valko		Stehfest		Salzer-Gaver	
	L	L_e	L	L_e	L	L_e	L	L_e
1	6.2 (−63)	1.1 (−63)	6.9 (−110)	1.3 (−110)	6.1 (−41)	8.5 (−43)	7.6 (−124)	1.0 (−125)
2	7.5 (−63)	1.2 (−63)	6.0 (−110)	1.2 (−110)	7.1 (−77)	9.7 (−79)	1.4 (−184)	2.0 (−184)
3	7.5 (−63)	1.2 (−63)	6.9 (−110)	1.3 (−110)	5.0 (−92)	5.0 (−92)	3.9 (−184)	3.9 (−184)
4	1.2 (−60)	2.5 (−60)	3.3 (−107)	6.9 (−107)	6.0 (−94)	7.0 (−93)	5.2 (−132)	6.0 (−132))
5	1.1 (−60)	1.7 (−60)	2.0 (−118)	2.8 (−118)	4.8 (−93)	2.6 (−92)	1.1 (−293)	2.0 (−292)
6	3.6 (−61)	3.6 (−61)	4.5 (−108)	4.5 (−108)	0.0 (0)	0.0 (0)	2.0 (−293)	1.1 (−292)
8	8.3 (−62)	3.4 (−62)	3.6 (−119)	1.5 (−119)	2.0 (−72)	7.4 (−74)	3.8 (−133)	1.1 (−132)
9	3.6 (−61)	3.6 (−61)	4.5 (−108)	4.5 (−108)	2.5 (−94)	3.5 (−93)	6.0 (−182)	2.1 (−184)
10	6.5 (−63)	1.1 (−63)	6.8 (−110)	1.2 (−110)	1.1 (−45)	1.5 (−47)	1.1 (−128)	1.5 (−130)
11	3.6 (−61)	3.6 (−61)	4.5 (−108)	4.5 (−108)	1.2 (−42)	1.6 (−44)	3.1 (−126)	4.2 (−128)

Table 7. Multi-Precision continued $N = 200$.

Function	Logan		Valko		Stehfest		Salzer-Gaver	
	L	L_e	L	L_e	L	L_e	L	L_e
12	5.9 (−61)	1.3 (−60)	6.9 (−110)	1.3 (−110)	6.1 (−41)	8.5 (−43)	3.4 (−132)	4.4 (−131)
13	3.0 (−61)	3.4 (−61)	6.0 (−110)	1.2 (−110)	7.1 (−77)	9.7 (−79)	1.6 (−133)	2.9 (−133)
14	7.4 (−63)	2.0 (−64)	6.9 (−110)	1.3 (−110)	5.0 (−92)	5.0 (−92)	3.4 (−102)	6.5 (−101)
15	6.7 (−62)	2.7 (−62)	3.3 (−107)	6.9 (−107)	6.0 (−94)	7.0 (−93)	1.3 (−133)	2.6 (−133))
16	3.6 (−61)	3.6 (−61)	2.0 (−118)	2.8 (−118)	4.8 (−93)	2.6 (−92)	2.5 (−131)	3.3 (−130)
17	1.4 (−60)	2.0 (−60)	2.3 (−107)	3.1 (−107)	0.0 (0)	0.0 (0)	8.5 (−131)	3.5 (−132)
18	1.1 (−60)	2.5 (−60)	3.2 (−107)	6.9 (−107)	1.4 (−41)	1.9 (−43)	6.3 (−132)	6.8 (−131)
19	8.3 (−62)	3.4 (−62)	3.6 (−119)	1.5 (−119)	2.0 (−72)	7.4 (−74)	3.8 (−133)	1.1 (−132)

accuracy than the Logan scheme. The tables show that the RMS error varied between 6.5(−11) and 6.2(−12).

Table 3 and Table 4 show that the Stehfest algorithm shows poor accuracy when inverting functions 1, 7, 10 and 11. For these functions the RMS error varies between 2.01(−2) to 9.2(−3). Its poor performance is due to the fact that the Stehfest algorithm has difficulty inverting functions of a cyclic nature [5]. However, it inverts the remaining functions with good accuracy with a RMS error of between 0 to 2.9(−5). Table 3 and Table 4 shows that the Salzer-Gaver algorithm shows poor accuracy for functions 1, 7, 10 and 11. These are the very same functions that the Stehfest algorithm has problems inverting. Again this is due to the difficulties it encounters when inverting cyclic functions. It inverts the remaining functions with less accuracy than the Stehfest with an RMS error varying between 10(−15) to 10(−5).

5.2. Multi Precision

With the exception of function 7, the Logan and Abate-Valko algorithms successfully inverted the remaining functions to a high degree of accuracy, see Table 6 and Table 7. Duffy [14] also remarks that when using the Talbot contour he had difficulties accurately inverting the Bessel function. This may related to the combination of the singularity on the imaginary axis and the branching nature of the square root function.

Abate-Valko [1] stated that they were able to overcome this by increasing the weights and hence the precision as a function of t. However, we were unable to replicate their results for this function.

Overall, the Abate-Valko scheme showed far greater accuracy than Logan's across all the functions tested. However, Logan's algorithm was still able to produce highly accurate results with RMS errors varying between 10(−60) to 10(−63). Moreover, Table 8 shows that Logan's scheme was able to perform the inversion of these functions with shorter elapsed times.

The Stehfset and Salzer-Gaver algorithms were able to invert all the functions to a high degree of accuracy. The Salzer-Gaver scheme was in general about twice as accurate as the Stehfest algorithm which, was less accurate than Abate-Valko's scheme. Nevertheless, the Stehfest scheme inverted the functions

Table 8. Elapsed time τ for $t = 100$.

Function	Logan	Valko	Stehfest	Salzer-Gaver
	τ	τ	τ	τ
2	0.89	1.44	0.43	2.37
8	0.55	0.88	0.41	3.06
11	0.94	3.87	1.74	2.56
13	0.72	0.94	0.41	0.99
18	0.56	1.05	0.52	1.27

well within any generally desired accuracy with the RMS error varying from 0.0 to 10(−41). Moreover, as Table 8 shows it in terms of the elapsed time it was the fastest of all the algorithms for the most part twice as fast as the Abate-Valko scheme which in turn was at least twice as fast as the Salzer-Gaver scheme.

6. Conclusions

In standard-double-precision, the Abate-Valko algorithm provides the best results for the numerical reconstructions for the functions tested in this paper. The Fourier algorithm had the worst performance of the five algorithms tested. Both the Stehfest and Salzer-Gaver algorithms had difficulty reconstructing functions of a cyclic nature. None of the algorithms was able to invert the $J_0(t)$ function accurately.

In multi-precision, the Stehfest and the Salzer-Gaver schemes inverted all the functions with high accuracy. The Logan and Abate-Valko schemes were only able to invert the $J_0(t)$ with limited accuracy. However they were both able to reconstruct all the other functions with a high degree of accuracy. The most accurate algorithm in multi-precision was the Salzer-Gaver scheme. However, as Table 8 shows it also had the longest elapsed times. On the other hand, the Stehfest algorithm had the shortest elapsed times for the selected functions in Table 8. The algorithms that used the Abate-Valko were the most accurate, but Logan could reconstruct the functions with shorter elapsed times. Therefore we conclude that when working in standard-precision Valko's algorithm performed best. However, in multi-precision, the Stehfest algorithm is best as it inverted all the functions with a high degree of accuracy and the shortest elapsed times.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Abate, J. and Valkó, P.P. (2004) Multi-Precision Laplace Transform Inversion. *International Journal for Numerical Methods in Engineering*, **60**, 979-993. https://doi.org/10.1002/nme.995

[2] Logan, J.D. (2013) Transport Modeling in Hydrogeochemical Systems, Volume 15. Springer Science & Business Media, Berlin.

[3] Spiegel, M.R. (1965) Schaum's Outline of Theory and Problems of Laplace Transforms. McGraw-Hill, New York.

[4] Talbot, A. (1979) The Accurate Numerical Inversion of Laplace Transforms. *IMA Journal of Applied Mathematics*, **23**, 97-120. https://doi.org/10.1093/imamat/23.1.97

[5] Stehfest, H. (1970) Algorithm 368: Numerical Inversion of Laplace Transforms [D5]. *Communications of the ACM*, **13**, 47-49. https://doi.org/10.1145/361953.361969

[6] Gaver Jr., D.P. (1966) Observing Stochastic Processes, and Approximate Transform Inversion. *Operations Research*, **14**, 444-459. https://doi.org/10.1287/opre.14.3.444

[7] Wimp, J. (1981) Sequence Transformations and Their Applications. Academic Press, Cambridge.

[8] Crump, K.S. (1976) Numerical Inversion of Laplace Transforms Using a Fourier Series Approximation. *Journal of the ACM* (*JACM*), **23**, 89-96. https://doi.org/10.1145/321921.321931

[9] Davies, A. and Crann, D. (2004) A Handbook of Essential Mathematical Formulae. University of Hertfordshire Press, Hatfield.

[10] Cohen, A.M. (2007) Numerical Methods for Laplace Transform Inversion, Volume 5. Springer Science & Business Media, Berlin.

[11] Epstein, C.L. and Schotland, J. (2008) The Bad Truth about Laplace's Transform. *SIAM Review*, **50**, 504-520. https://doi.org/10.1137/060657273

[12] Kuhlman, K.L. (2012) Comparison of Inverse Laplace Transform Algorithms for Laplace-Space Numerical Approaches. Technical Report, Sandia National Laboratories, Albuquerque.

[13] Defreitas, C.L. and Kane, S.J. (2018) The Noise Handling Properties of the Talbot Algorithm for Numerically Inverting the Laplace Transform. *Journal of Algorithms & Computational Technology*, **13**, 1-14. https://doi.org/10.1177/1748301818797069

[14] Duffy, D.G. (1993) On the Numerical Inversion of Laplace Transforms: Comparison of Three New Methods on Characteristic Problems from Applications. *ACM Transactions on Mathematical Software* (*TOMS*), **19**, 333-359. https://doi.org/10.1145/155743.155788

[15] Narayanan, G.V. and Beskos, D.E. (1982) Numerical Operational Methods for Time-dependent Linear Problems. *International Journal for Numerical Methods in Engineering*, **18**, 1829-1854. https://doi.org/10.1002/nme.1620181207

[16] Davies, B. and Martin, B. (1979) Numerical Inversion of the Laplace Transform: A Survey and Comparison of Methods. *Journal of Computational Physics*, **33**, 1-32. https://doi.org/10.1016/0021-9991(79)90025-1

[17] Le Page, W.R. (1980) Complex Variables and the Laplace Transform for Engineers. Courier Corporation, North Chelmsford, MA.

[18] Dubner, H. and Abate, J. (1968) Numerical Inversion of Laplace Transforms by Relating Them to the Finite Fourier Cosine Transform. *Journal of the ACM* (*JACM*), **15**, 115-123. https://doi.org/10.1145/321439.321446

[19] Simon, R.M., Stroot, M.T. and Weiss, G.H. (1972) Numerical Inversion of Laplace Transforms with Application to Percentage Labeled Mitoses Experiments. *Computers and Biomedical Research*, **5**, 596-607. https://doi.org/10.1016/0010-4809(72)90039-0

[20] Cooley, J.W. and Tukey, J.W. (1965) An Algorithm for the Machine Calculation of Complex Fourier Series. *Mathematics of Computation*, **19**, 297-301. https://doi.org/10.1090/S0025-5718-1965-0178586-1

[21] Cooley, J.W., Lewis, P.A.W. and Welch, P.D. (1970) The Fast Fourier Transform Algorithm: Programming Considerations in the Calculation of Sine, Cosine and Laplace Transforms. *Journal of Sound and Vibration*, **12**, 315-337. https://doi.org/10.1016/0022-460X(70)90075-1

[22] Murli, A. and Rizzardi, M. (1990) Algorithm 682: Talbot's Method of the Laplace Inversion Problems. *ACM Transactions on Mathematical Software* (*TOMS*), **16**, 158-168. https://doi.org/10.1145/78928.78932

[23] Bender, C.M. and Orszag, S.A. (2013) Advanced Mathematical Methods for Scientists and Engineers I: Asymptotic Methods and Perturbation Theory. Springer Science & Business Media, Berlin.