VALIDATION OF NUMERICAL INVERSION METHOD FOR LAPLACE TRANSFORM

Andreea SOIMU, Dorin POPA, Andrei MARINESCU

ICMET Craiova, s_andreea@icmet.ro

***Abstract* – Currently there are several numerical methods for inverting the Laplace transform (ILT) but not all of them are used in electrical engineering. The main difficulty regarding the electric circuits is, among others, to find the transfer function poles represented by high degree polynomials.**
In this paper, we propose a numerical method based on Fourier series for inverting Laplace transforms in the case of transient analysis.
To accelerate the convergence of Fourier series were used acceleration algorithms.
This method has been verified and validated on a program implemented in Matlab.
Two methods are used for the verification and validation of the numerical calculation results. The first method uses a comparison with exact solution, if it has an analytical form and the second method compares the performance of the software developed by the authors with the results published by other authors who have used techniques more or less different.

Keywords: *Laplace transform, numerical methods, verification, validation.*

1. INTRODUCTION

Laplace transforms (LT) are powerful tools in many problems of mathematics, physics, optics, electrical engineering etc.
LT is a method of solving differential equations, which are transformed into algebraic equations.
Analytical or numerical methods can be used to convert the solution obtained from frequency domain in time domain.
ILT is a difficult problem, especially for complex circuits, which is why lately LT use was avoided [1].
The Laplace transform of a function *f(t)*, defined for all real numbers $t \geq 0$, is the function $F(s)$, defined by:

$$F(s) = L\{f(t)\} = \int_{-\infty}^{+\infty} e^{-st} f(t)dt \qquad (1)$$

where s is a complex number: $s = \sigma + i\omega$
The inverse Laplace transform is:

$$f(t) = L^{-1}\{F(s)\} = \frac{1}{2\pi i}\int_{\sigma-i\infty}^{\sigma+i\infty} e^{st} F(s)ds \qquad (2)$$

The main difficulty is to determining the transfer function poles.

This paper proposes that the ILT to be performed by a numerical method based on Fourier series, method which was first proposed by Dubner and Abate [2].
Acceleration algorithms are used to accelerate the convergence of the obtained Fourier series.
ILT method was implemented in Matlab.
Some examples of this method allow verification and validation.

2. NUMERICAL METHOD FOR ITL

The proposed method is based on the Bromwich integral, which enables functions of time (based) on the relationship [3]:

$$f(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{(\sigma+j\omega)t} F(\sigma + j\omega)d\omega \qquad (3)$$

This integral can be written as:

$$f(t) = \frac{2e^{\sigma t}}{\pi}\int_0^{\infty} \mathrm{Re}\left[F(\sigma + j\omega)\right]\cos(\omega t)d\omega =$$

$$= -\frac{2e^{\sigma t}}{\pi}\int_0^{\infty} \mathrm{Im}\left[F(\sigma + j\omega)\right]\sin(\omega t)d\omega \qquad (4)$$

which is a Fourier integral.
Integral approximation by Fourier series resortes to the trapezoidal rule and uses a time discretization error [4].
For a step size h, we obtain:

$$f_h(t) = \frac{e^{A/2}}{2t}\mathrm{Re}\left(F\left(\frac{A}{2t}\right)\right) +$$

$$+ \frac{e^{A/2}}{t}\sum_{k=1}^{n}(-1)^k\,\mathrm{Re}\left(F\left(\frac{A + 2k\pi j}{2t}\right)\right) \qquad (5)$$

After successive transformations [4] and replacement $h = \pi / 2t$ and $b = A / 2t$, we obtain the following form:

$$f(t) = \frac{e^{A/2}}{2t}\sum_{k=-\infty}^{\infty}(-1)^k\,\mathrm{Re}\,F\left(\frac{A + 2k\pi j}{2t}\right)$$

$$-\sum_{k=1}^{\infty} e^{-kA} f\left((2k+1)t\right) \qquad (6)$$

where the first term coincides with the trapezoidal-rule approximation and the second term gives the

discretization error associated with the trapezoidal rule.

If $|f(t)| \le 1$ for all t, then the error is bounded by $|e_d| \le \frac{e^{-A}}{1-e^{-A}}$ which is approximately equal to e^{-A} when e^{-A} is small. Hence, to have at most $10^{-\gamma}$ discretization error, we let $A = \gamma \log 10$.

Since the series from (4) is alternant and therefore convergent is slow, the algorithms for accelerating the convergence should be used.

One of the most elementary acceleration techniques is Euler summation [5].This technique is based on Euler summation, which for an alternating series is very simply described, as the weighted average of the last *m* partial sums of the binomial probability distribution with parameters *m* and *p=1/2*.

If we rewrite equation (5) as the partial sum:

$$s_n(t) = \frac{e^{A/2}}{2t}\mathrm{Re}(F)\left(\frac{A}{2t}\right) + \frac{e^{A/2}}{t}\sum_{k=1}^{n}(-1)^k a_k(t)$$

then the ILT is given by:

$$f(t)_{Euler} = \sum_{k=0}^{m}\binom{m}{k}2^{-m}s_{n+k}(t) \qquad (7)$$

where $a_k(t) = \mathrm{Re}(F)\left(\frac{A+2k\pi i}{2t}\right)$.

Another technique is the epsilon algorithm [6].

The Epsilon algorithm for accelerating the convergence [7] is done by computing a diagonal Padé approximation of this power series.

$$f(t) = \frac{e^{\sigma t}}{T}\sum_{k=0}^{N}\mathrm{Re}(c_k z^k) \qquad (8)$$

where $z = \exp(jk\pi t/T)$ and T=max(t)*2.

To achieve calculations is used an algorithm based on continued fractions:

$$v(z,N) = d_0/(1 + d_1 z/(1 + \ldots + d_{2N} z))$$

The coefficients of the continued fractions are computed using the sequences e_r^i and q_r^i based on the epsilon algorithm [8]:

$$e_0^{(i)} = 0, i = 0,1,\ldots,2N;\ q_1^{(i)} = \frac{c_{i+1}}{c_i} i = 0,1,\ldots 2N-1$$

$$e_r^{(i)} = q_r^{(1+1)} - q_r^{(i)} + e_{r-1}^{(i+1)}, i = 0,1,\ldots,2N-2r, r = 1,2,...,N$$

$$q_r^{(i)} = \frac{q_{r-1}^{(i+1)} e_{r-1}^{(i+1)}}{e_{r-1}^{(i)}}, i = 0,1,\ldots,2N-2r-1, r = 2,3,\ldots,N$$

Then

$$d_0 = c_0, d_{2k-1} = -q_k^{(0)}, d_{2k} = -e_k^{(0)},$$
$$k = 1,2,\ldots,N$$

Then the ILT is:

$$f(t)_{epsilon} = e^{\sigma t}\,\mathrm{Re}[v(z,N)]/T \qquad (9)$$

Based on (7) and (9), the programs were developed in Matlab [9], which compute ILT.

3. VALIDATION OF NUMERICAL METHODS DEVELOPED

Two methods are usually used for the verification and validation of the numerical calculation results.

The first method uses a comparison with exact solution, if it has an analytical form and the second method compares the performance of the software developed by the authors with the results published by other authors who have used techniques more or less different.

For this purpose, the computer program developed in Matlab has been applied for several transfer functions with different data regarding the type phase described by the LT (damped oscillatory, a periodic damping with varying degrees).

3.1. Validation by comparison with exact solution

Consider the following rational function:

$$F(s) = \frac{1/\sqrt{2}}{0.5s^3 + 2.5s^2 + 4s + 2} \qquad (10)$$

which is critic damped impulse with exact solution [1]:

$$f(t) = \sqrt{2}\left[\exp(-t) - (1+t)\exp(-2t)\right] \qquad (11)$$

First the function from (10) was inverted by Fourier series methods according to equation (6), and then the two acceleration algorithms are used, and the obtained results were compared with exact values.

Matlab programs were tested on different levels of discretization error.

Table 1 shows the numerical values of ILT obtained through software programs developed in Matlab and the exact values, and the computing time.

If the function *f(t)* is calculated according to (6), without acceleration algorithms, we obtain a computing time of 0.6 sec and 0.5 sec for 10^{-3} and 10^{-1} discretization error.

Analyzing the results in Table 1, we note a 2-3 fold reduction in computing time, without affecting the accuracy of calculation.

For the 10^{-3} discretization error, the feature given by the exact function overlaps the feature obtained by the proposed method; we note small differences between the two features illustrated in Figures 1 and 2 for a 10^{-1} discretization error.

t[sec]	$f(t)_{Euler}$		$f(t)_{epsilon}$		$f(t)$ Exact value
	Discretization error				
	10^{-3} t_{comp}=0.36s	10^{-1} t_{comp}=0.32s	10^{-3} t_{comp}=0.26s	10^{-1} t_{comp}=0.21s	
0.1	0.0060	0.0073	0.0060	0.0071	**0.0060**
0.2	0.0204	0.0234	0.0203	0.0231	**0.0203**
0.3	0.0388	0.0430	0.0387	0.0413	**0.0387**
0.4	0.0584	0.0630	0.0583	0.0607	**0.0583**
0.5	0.0775	0.0819	0.0774	0.0795	**0.0774**
0.6	0.0947	0.0986	0.0946	0.0966	**0.0946**
0.7	0.1095	0.1129	0.1094	0.1112	**0.1094**
0.8	0.1216	0.1243	0.1215	0.1231	**0.1215**
0.9	0.1309	0.1331	0.1308	0.1323	**0.1308**
1	0.1375	0.1393	0.1375	0.1384	**0.1375**

Table 1: Comparing the results with exact solution

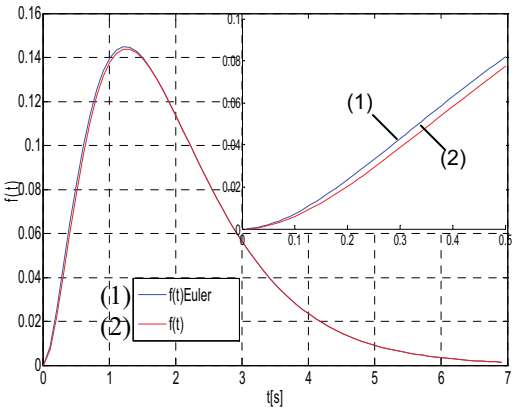

Figure 1: ILT_Euler vs (11)

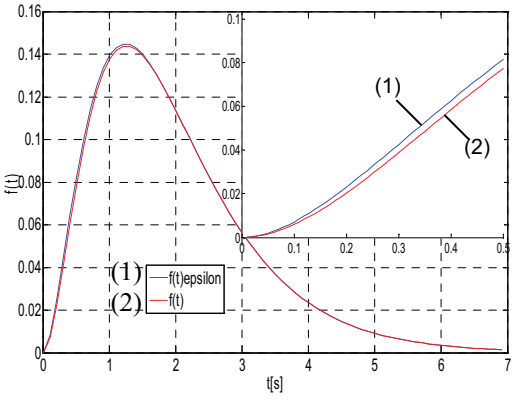

Figure 2: ILT_epsilon vs (11)

In figures 1 and 2 are plotted ILT for function (10) by Fourier series method (curve 1) and ILT exact (curve 2), represented by function (11).
Plotting was done in order to provide an intuitive idea of the accuracy of computation results of the numerical ILT, which shows that the differences are absolutely negligible for technical applications.

3.2. Validation by comparison with published results

By this method, a function is inverted and compare with results of other authors.
The paper [10] provides ILT results for the following function:

$$F(s)=\frac{1400s^2+9520s+13440}{(s^5+25.19s^4+1578.7315s^3+ \\ +15502.14879s^2+24589.8596s)} \qquad (12)$$

which is a poorly damped oscillating impulse.
In Table 2, the results obtained by the proposed method (discretization error is 10^{-3}) are compared to the results provided in [10].

t[s]	$f(t)_{Euler}$ t_{comp}=0.30s	$f(t)_{epsilon}$ t_{comp}=0.27s	$f(t)$ [10]
0.01	0.0652	0.0652	0.0651
0.02	0.2371	0.2371	0.2370
0.03	0.4747	0.4746	0.0745
0.04	0.7342	0.7342	0.7341
0.05	0.9753	0.9754	0.9752
0.1	1.1808	1.1808	1.1807
0.15	0.4367	0.4367	0.4366
0.2	0.4903	0.4903	0.4901
0.25	0.7552	0.7552	0.7552
0.3	0.5805	0.5805	0.5804
0.35	0.4842	0.4842	0.4839
0.4	0.5850	0.5850	0.5849

0.45	0.5816	0.5816	0.5816
0.5	0.5256	0.5256	0.5254
0.6	0.5609	0.5609	0.5607
0.7	0.5402	0.5403	0.5401
0.8	0.5743	0.5743	0.5741
0.9	0.5458	0.5458	0.5457
1	0.5453	0.5453	0.5453

Table 2: Comparing results from different techniques

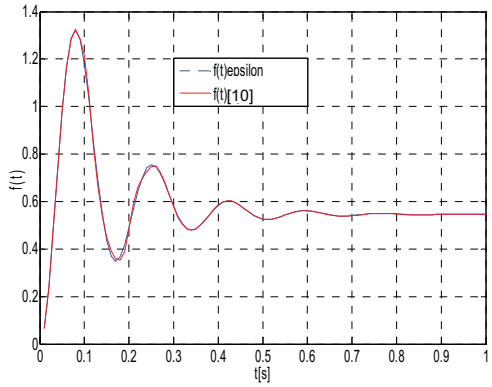

Figure 3: ITL_epsilon vs f(t)[10]

Since the results obtained for ILT with the two acceleration algorithms are comparable, in Figure 3 the ILT with epsilon algorithm was presented, because it requires shorter computing time.

4. CONCLUSIONS

The Laplace transforms inversion by Fourier series method is an efficient computing technique.
The ILT by Fourier series method was tested on several functions with different character in terms of phase-type described by LT (damped oscillatory, a periodic damping with varying degrees).
By using algorithms to accelerate the convergence of series, the computing time is reduced by 2 to 3 times (0.2-0.3 sec against 0.5-0.6 sec)
In the examples described in this paper, the epsilon algorithm proves more efficient results than the Euler algorithm in terms of computation time and accuracy.
Note that when using accelerating algorithms the results technically acceptable can be obtained even when using higher discretization (mesh) errors.
The method will continue to be used for other types of LT in order to determine the applicability limits of the method.

References

[1] A. Marinescu, *Comportarea Transformatoarelor la Supratensiuni de Comutatie*, Editura Tehnica, Bucuresti, 1988.

[2] H. Dubner, J. Abate, *Numerical Inversion of Laplace Transforms by Relating Them to the Finite Fourier Cosine Transform,* JACM, vol. 15, 1968, pp. 115-123.

[3] J.L. Schiff, *The Laplace Transform: Theory and Applications*, Springer, 1999.

[4] J. Abate, W. Whitt, *Numerical Inversion of Laplace Transforms of Probability Distributions*, ORSA Journal on Computing, vol. 7, no. 1, 1996, pp. 36-43.

[5] R.M. Simon, M.T. Stroot, G.H. Weiss *Inversion of Laplace Transforms with Applications to Pertentage labeled Experiments*, Comput. Biomed, pp. 596-607.

[6] F.R. De Hoog, J.H. Knight, A.N. Stokes, *An improved method for Numerical Inversion of Laplace Transforms*, SIAM J. Sci. Stat. Comput., vol. 3, no. 1, 1982, pp. 357-366.

[7] D.G Duffy, *Transform Methods for Solving Partial Differential Equations*, CRC Press LLC, 2000.

[8] http://en.wikipedia.org/wiki/Continued_fraction

[9] A Soimu, *Program pentru inversarea transformatei Laplace prin metoda seriilor Fourier,* ICMET Craiova, 2010.

[10] D. Popa, *O metoda numerica de inversiune a transformatei Laplace a functiilor rationale,* E.E.A. Electrotehnica, vol. 31, no. 2, 1983, pp. 60-63.