A divide-and-conquer algorithm for seismic data approximation by the Laguerre series

A divide-and-conquer algorithm for seismic data approximation by the Laguerre series

Andrew V Terekhov

Institute of Computational Mathematics and Mathematical Geophysics, Ac Lavrentieva ave., 6, Novosibirsk, 630090, Russia
Novosibirsk State Technical University, K. Marksa ave., 20, Novosibirsk, 630073, Russia

E-mail: andrew.terekhov@mail.ru

Abstract. An algorithm of the Laguerre transform for approximating functions on large intervals is proposed. The idea of the considered approach is that the calculation of improper integrals of rapidly oscillating functions is replaced by a solution of an initial boundary value problem for the one-dimensional transport equation. It allows one to successfully avoid the problems associated with the stable implementation of the Laguerre transform. A divide-and-conquer algorithm based on shift operations made it possible to significantly reduce the computational cost of the proposed method. Numerical experiments have shown that the methods are economical in the number of operations, stable, and have satisfactory accuracy for seismic data approximation.

1. Introduction

Consider Laguerre functions, which are defined as [1]

$$l_m(t) = e^{-t/2} L_m(t), \quad t \geq 0, \tag{1}$$

where $L_m(t)$ is the Laguerre polynomial of degree m. The Laguerre functions are a complete orthonormal system in $L_2[0,\infty)$, which guarantees that for any function $f(t) \in L_2[0,\infty)$ there is a Laguerre expansion

$$f(t) \approx \eta \sum_{m=0}^{n} \bar{a}_m l_m(\eta t), \quad \bar{a}_m = \int_0^\infty f(t) l_m(\eta t) dt, \quad t \geq 0, \tag{2}$$

where $\eta > 0$ is a scaling parameter.

It is well-known that, owing to the high performance and stability of the algorithm of fast Fourier transform, it has been widely used in many branches of computational mathematics, whereas no algorithm for the Laguerre transform having comparable efficiency has been developed so far. Although general methods of fast polynomial transforms were proposed long ago [2], they are of theoretical rather than practical importance. This is because they use numerically unstable efficient procedures of multiplying matrices V and V^{T} by a vector, where $V \in \mathbb{R}^{n\times n}$ is an ill-conditioned Vandermonde matrix [3]. For instance, fast multiplication by the matrix V can be performed by using an algorithm [4] whose computational complexity is of the order of $O(n \log^2 n)$ operations. Unfortunately, this method is unstable, since one of its

stages includes a recursive use of the operation of polynomial division. Multiplication of the matrix V^{T} by a vector can be reduced to solving systems of linear algebraic equations with a Vandermonde matrix with an operation count of the order of $O(n\log^2 n)$ [5, 6]. This approach also cannot be recommended for practical use due to its numerical instability.

The condition number for Laguerre functions is greater than that for the other classical orthogonal polynomials. Therefore, the problem of stability of fast algorithms for the Laguerre transform is probably one of the most difficult ones. By now, fast transforms have been developed for Chebyshev, Legendre, and Hermite polynomials [7, 8, 9]. In these cases the arithmetic complexity of the algorithms is of the order of $O(n\log n)$ or $O(n\log^2 n)$ operations. Fast algorithms of changing from one orthogonal polynomial basis specified by a three-term recurrence relation to another one have also been developed [10]. In paper [11], an algorithm for fast polynomial transforms based on an approximate factorization of the matrices V or V^{T} was proposed. In some cases the authors managed to decrease the computational costs to a level of $O(n\log n)$ arithmetic operations. However, the computational complexity may vary widely for various orthogonal polynomials and expansion interval lengths. Also, the algorithm becomes efficient in comparison to the direct method of multiplying a matrix by a vector, for $n \geq n_0$, where n_0 is of the order of several thousand.

The Laguerre transform (2) has been used in various fields of mathematical simulation to solve acoustics and elasticity equations [12, 13], Maxwell and heat conduction equations [14, 15], and spectroscopy problems. These problems are of considerable interest for mathematical modeling and spectral analysis methods. In this paper, a new fast method to calculate Laguerre series coefficients is proposed for the approximation of functions on large intervals.

2. Expansion algorithms

2.1. Main formulas

Let us consider Algorithm 1 of expanding the function $f(t)$ in a Laguerre series. To expand a function by formula (3), about $O(nN_x)$ arithmetic operations are needed, where n is the number of expansion terms of the Laguerre series and N_x is the number of harmonics of the auxiliary Fourier series. Approximation of the function for longer intervals calls for specifying larger values of n and N_x, which makes the Laguerre transform inefficient. To decrease the calculation time when performing the Laguerre transform, we consider an algorithm of the "divide and conquer" type. The general idea of this approach is that at the first stage the initial problem is divided into independent subproblems (figure 1) with much less computational costs needed for their solution. At the second stage the solution to the initial problem is assembled from the solutions to the subproblems.

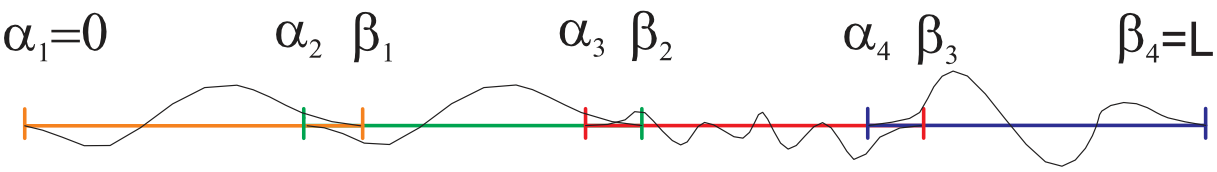

Figure 1. Decompositions of the initial approximation interval into four overlapping subintervals.

Algorithm 1 to approximate a function $f(t)$ on the interval $t \in [0, L]$ by a Laguerre series:

(i) For given $n > 20$ and η, calculate σ_n

$$\sigma_n = 4n - \frac{10 l_n(4n)}{l'_n(4n)} = 4n - \frac{10 l_n(4n)}{\frac{1}{2} l_n(4n) - \sum_{k=0}^{n-1} l_k(4n)}$$

and set a new approximation interval $[0, \max(L, \sigma_n/\eta)]$.

(ii) Calculate $\tilde{f}(k_j) = \mathrm{DFT}\{f(t_i)\}$ for a discrete set of frequencies, $k_j = \frac{2\pi}{\sigma_n/\eta}(j - \frac{N_x}{2})$, $j = 0, 1, ..., N_x - 1$.

(iii) Calculate the expansion coefficients of series (2) by the formula

$$\bar{a}_m \approx \sum_{j=0}^{N_x-1} \tilde{f}(k_j) \left[\left(-\frac{\eta}{2} - \mathrm{I}k_j \right)^m / \left(\frac{\eta}{2} - \mathrm{I}k_j \right)^{m+1} \right], \; m = 0, 1, ..., n. \tag{3}$$

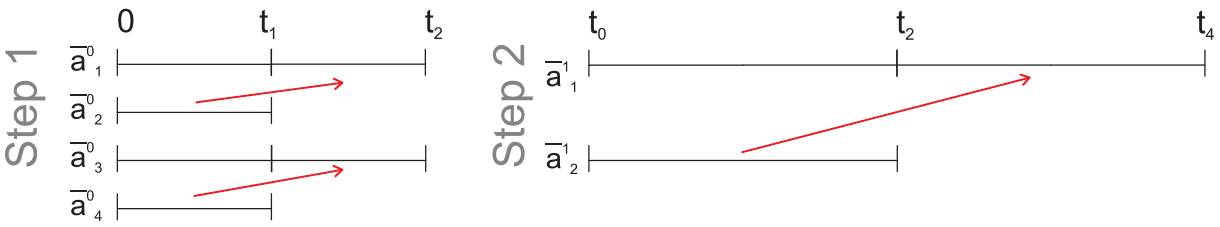

Figure 2. Approximation construction scheme for the function $f(t)$ with precalculated approximations for the local functions $f_i(t)$.

For the calculation scheme in figure 2 every shift doubles the minimum number of the Laguerre series terms. This can be explained by the fact that the Laguerre functions of the m-th order oscillate on the interval $0 < t < 4m$. Therefore, before making a shift the sequence of coefficients of the Laguerre series must be added by zeros (zero padding). After making the shift the zero values of the added expansion coefficients will take nonzero values. For larger values of n and N_x the computational complexity of Algorithm 2 will be of the order $O(nN_x/p + n \log_2 n \log_2 p)$ vs. $O(nN_x)$, where p is the number of subintervals. The first term is the costs to approximate the local functions $f_i(t)$, $t \in [0, \Delta t_i]$, and the second one is the costs to perform a series of shift operations to transform the local expansion coefficients to the expansion coefficients for the initial function $f(t)$. However, this algorithm has the following shortcoming: the number of Laguerre series coefficients to approximate the local functions $f_i(t)$ on the subintervals $[0, \Delta t_i]$ depends not only on the lengths of the subintervals, but also on the smoothness of the functions $f_i(t)$. Taking into account that in solving practical problems the function to be approximated may have low smoothness, the convergence of the series may be not high. This results in the fact that at the same accuracy the total number of expansion coefficients for the local problems for Algorithm 2 will be greater than the number of expansion coefficients when using Algorithm 1. Thus, the division will require additional computational costs, which can be estimated in computational experiments.

3. Numerical experiments

To estimate the accuracy of the approximation and the efficiency of the methods being proposed, let us perform a series of computational experiments to approximate functions on intervals

Algorithm 2 to approximate a function $f(t)$ on an interval $t \in [0, L]$ by a Laguerre series:

(i) Decompose the approximation interval $t \in [0, L]$ into $p = 2^s$ overlapping subintervals of lengths $\Delta t_i = \beta_i - \alpha_i$ (Fig. 1). In this case the function must smoothly tend to zero on the subinterval boundaries in the buffer zones so that the sum of the two local functions remains equal to the value of the function being approximated.

(ii) The local function $f_i(t)$ specified on the subinterval with number i is expanded in a Laguerre series on the auxiliary interval $[0, \Delta t_i]$ by Algorithm 1.

(iii) Shift the local functions by changing from the interval $[0, \Delta t_i]$ to the subinterval $[\alpha_i, \beta_i]$. This is done by a series of shifts of the function $f_i(t)$ using the scheme presented in Fig. 2, which gives an example of four subintervals. The process of assembly consists of $\log_2 p$ steps, where p is the number of subintervals. Hence, two steps will be needed for the example being considered. At the first step the sequences of the Laguerre series coefficients for the local functions $f_2(t)$ and $f_4(t)$ are supplemented by zeroes to double the number of expansion coefficients.

(iv) Then the thus expanded series are shifted using the procedures $\mathbb{S}\left\{\bar{a}_{n/2}; \alpha_2\right\}$ and $\mathbb{S}\left\{\bar{a}_{n/2}; \alpha_4 - \alpha_3\right\}$, where

$$\mathbb{S}\left\{\bar{a}_m; \tau\right\} = \sum_{j=0}^{m} \left(\bar{a}_{m-j} - \bar{a}_{m-j-1}\right) l_j(\eta\tau) \ , \bar{a}_{-1} \equiv 0.$$

To calculate this linear convolution one can use algorithms based on the fast Fourier transform [16]. After this the corresponding coefficients of the first and second series and of the third and fourth series are added pairwise. This results in two intervals of larger lengths. At the second step this process is used for the new second series, and after it is shifted by $\mathbb{S}\left\{\bar{a}_n; \alpha_3\right\}$ the expansion coefficients of the first and second series are added. Thus, all local functions will be shifted to their initial positions with respect to the variable t, and the thus obtained series will approximate the initial function $f(t)$ with some accuracy.

of various lengths. The numerical procedures to calculate the Laguerre coefficients will be performed with double precision. All algorithms have been implemented as Fortran-2008 programs using BLAS and FFTW libraries of the Intel Math Kernel Library. The calculations have been made on Intel Xeon E5-2680v3 processors with a frequency of 2500 MHz and 192Gb RAM.

The approximation error is estimated by the formula

$$\varepsilon = \left(\sum_{i=1}^{s} \left[f(t_i) - \sum_{j=1}^{n} \bar{a}_j l_j(\eta t_i) \right]^2 / \sum_{i=1}^{s} f^2(t_i) \right)^{1/2}, \tag{4}$$

where $f(t_i)$ is the function to be expanded in a Laguerre series, which is specified on a set of values $t_i \in [0, L], i = 1, 2, ...s,\ t_1 = 0,\ t_s = L$.

In testing of Algorithm 2, table 1 and figures. 3 present the calculation times and accuracy estimates in the approximation of all 152684 seismic traces for the Sigsbee model (figure 4). For the interval $[0, 12]$ the numbers of coefficients of the series were $n = 4096$ and 8192, and for the intervals $[0, 60]$ and $[0, 120]$ the numbers of coefficients of the series were specified as $n = 32768$ and 65536, respectively. Initial seismic traces for the Sigsbee model were specified for $t \in [0, 12]$. To obtain the time series for the intervals $[0, 60]$ and $[0, 120]$, the initial trace

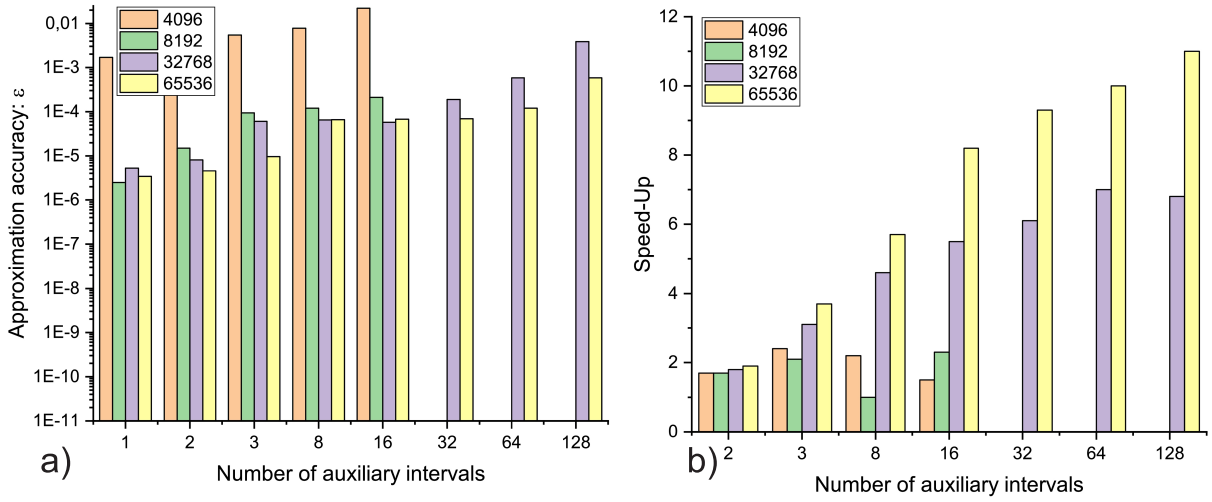

Figure 3. a) Estimates of accuracy of Algorithm 2, b) ratio between calculation time for Algorithm 1 and calculation time for Algorithm 2 (Table 1)

	L $= 12s$, $n = 4096$		L $= 12s$, $n = 8192$		L $= 60s$, $n = 32768$		T $= 120s$, $n = 65536$	
SubInt	Init	Step_{1+2}	Init	Step_{1+2}	Init	Step_{1+2}	Init	Step_{1+2}
1	1.7	29	3.9	52	77	878	3417	3450
2	0.3	16	1.1	28	19	474	1725	1794
3	5.2e-2	12	0.2	25	4.8	275	827	917
8	1.1e-2	13	4.0e-2	26	1.2	187	468	617
16	8.8e-2	19	9.1e-2	40	2.5	159	248	412
32	-	-	-	-	4.2E-2	91	125	360
64	-	-	-	-	9.6E-3	100	231	331
128	-	-	-	-	1.7E-3	112	225	299

Table 1. Estimates of calculation time and accuracy of Algorithm 2. The number of auxiliary intervals p versus: (*Init*) preparatory calculation time needed to calculate the local matrix; (*Step 1*) calculation time of local approximation; (*Step 2*) calculation time of the sequence of shifts for constructing the global approximation;

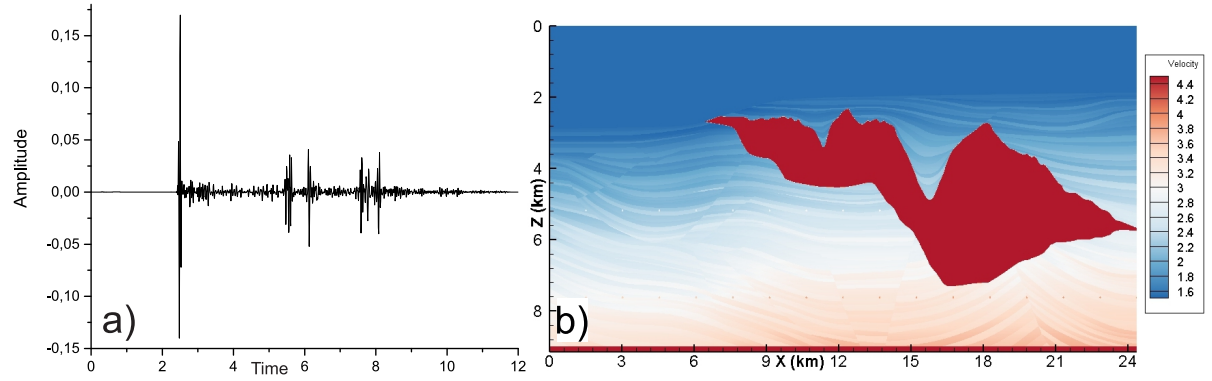

Figure 4. a) First trace from seismograms for b) the Sigsbee velocity model

was supplemented by four or eleven identical copies of the initial signal, respectively. One can see from the data presented that, although Algorithm 2 does not belong to the class of fast algorithms, it allows a slight decrease in the calculation time, especially for large time intervals. It follows from table 1 and figures 3 the approximation accuracy ϵ decreases as the number of auxiliary intervals increases. This is caused, first, by the presence of auxiliary buffers, in which multiplication by an exponentially decreasing factor is made for the Laguerre spectrum of a local function not to be infinite because of the discontinuities of the function values on the boundaries of the subintervals. On each subinterval the local function is approximated by a Laguerre series with a number of coefficients of n/p, where p is the number of subintervals. However, n/p expansion coefficients may be insufficient to approximate a nonsmooth local function, which results in loss in approximation accuracy. Nevertheless, if one has to approximate a time series with an accuracy of the order of $\epsilon = 10^{-3} \div 10^{-5}$, it is recommended to use Algorithm 2.

To multiply the matrix by a vector (see formula (3)), a numerical procedure from BLAS MKL library was used. Taking into account high degree of optimization of the BLAS procedure for a specific processor model, the calculation of Laguerre coefficients is performed very fast. At the same time, optimization of the algorithm of fast Fourier transform is a more complicated problem. This decreases the degree of vectorization of the calculations at the second step of Algorithm 2. As a result, the speedup of Algorithm 2 also decreases. If an internal Fortran procedure, such as "matlmul", had been used for matrix multiplication instead of that from BLAS library, the speedup coefficient of Algorithm 2 would have been much larger (although the total calculation time also increases), since the computational costs of the "matmul" function are several times greater than those of the procedure from the BLAS MKL library.

4. Conclusions

In this paper, a new approach has been developed to decrease the computational costs in making the Laguerre transform for large approximation intervals by solving the transport equation. The test calculations have confirmed that all developed algorithms can be used both with single and double real precision without loss of numerical stability. Thus, if a large set of functions is approximated by a Laguerre series (for instance, in solving problems of seismic prospecting), the above-mentioned algorithms allow saving the calculation time considerably.

5. Acknowledgements

The numerical implementation of proposed algorithms was carried out under state contract with ICMMG SB RAS (0251-2021-0004), the study of proposed algorithms was financially supported by RFBR and Novosibirsk region (Project No. 20-41-540003).

References

[1] *NIST Digital Library of Mathematical Functions* http://dlmf.nist.gov/, Release 1.1.0 of 2020-12-15 f. W. J. Olver, A. B. Olde Daalhuis, D. W. Lozier, B. I. Schneider, R. F. Boisvert, C. W. Clark, B. R. Miller, B. V. Saunders, H. S. Cohl, and M. A. McClain, eds.
[2] Gohberg I and Olshevsky V 1994 *Journal of Complexity* **10** 411 – 427
[3] Pan V 2016 *SIAM Journal on Matrix Analysis and Applications* **37** 676–694
[4] Borodin A and Moenck R 1974 *Journal of Computer and System Sciences* **8** 366 – 386
[5] Gohberg I and Olshevsky V 1994 *Linear Algebra and its Applications* **202** 163 – 192
[6] Pan V, Sadikou A, Landowne E and Tiga O 1993 *Computers & Mathematics with Applications* **25** 25 – 30
[7] Alpert B and Rokhlin V 1991 *SIAM Journal on Scientific and Statistical Computing* **12** 158–179
[8] Hale N and Townsend A 2016 *IMA Journal of Numerical Analysis* **36** 1670–1684
[9] Leibon G, Rockmore D N, Park W, Taintor R and Chirikjian G S 2008 *Theoretical Computer Science* **409** 211 – 228
[10] Bostan A, Salvy B and Schost E 2010 *Linear Algebra and its Applications* **432** 249 – 258
[11] O'Neil M, Woolfe F and Rokhlin V 2010 *Applied and Computational Harmonic Analysis* **28** 203 – 226
[12] Fatyanov A G and Terekhov A V 2011 *J. Comp. Phys.* **230** 1992–2003

[13] Mikhailenko B G 1999 *Applied Mathematics Letters* **12** 105–110
[14] Mikhailenko B G and Mastryukov A F 2008 *Russian Geology and Geophysics* **49** 621–627
[15] Colton D and Wimp J 1984 *Complex Variables, Theory and Application: An International Journal* **3** 397–412
[16] Nussbaumer H J 1982 *Fast Fourier Transform and Convolution Algorithms* (Springer-Verlag)