Optimization and Limitations of a Multi-Level Adaptive-Order Fast Multipole Algorithm for Acoustical Calculations

Ralf Burgschweiger[1], Ingo Schäfer[2], Martin Ochmann[1], Bodo Nolte[2]

[1] *Beuth Hochschule für Technik Berlin, University of Applied Sciences, Faculty II, Research Group Computational Acoustics, Luxemburger Str. 10, 13353 Berlin, Germany*

[2] *Federal Armed Forces Underwater Acoustics and Marine Geophysics Research Institute, Klausdorfer Weg 2-24, 24148 Kiel, Germany*

Abstract: The Multi-Level Fast Multipole Method (MLFMM) allows the computation of acoustical problems based on the Boundary Element Method (BEM) where the discretized models of the corresponding structures may consist of a huge number of elements. The required calculation time and the memory requirements are much less when compared with conventional methods because the algorithm uses a level-based composition of the potentials from different point sources to acoustic multipoles, which highly accelerates the computation of the matrix-vector-products required for iterative solvers. A multi-level single-order variation of the algorithm developed during a previous research project was extended to a multi-level adaptive-order version, which was analyzed and optimized with respect to quality, performance and parallelization issues. The insights gained will be presented using different test cases and the results achieved will be compared with analytical solutions and results based on conventional BEM- and FEM-based calculations.
Keywords: Fast Multipole Method, Optimization, Numerical Methods

1. Introduction

The Multi-Level Fast Multipole Method (MLFMM) describes a fast algorithm to accelerate the matrix-vector product which is required for the iterative solution of BEM-based calculations without ever assembling the complete matrix. The method is suited for big problems where the interactions between huge numbers of source (N_x) and destination points (N_y) must be considered.

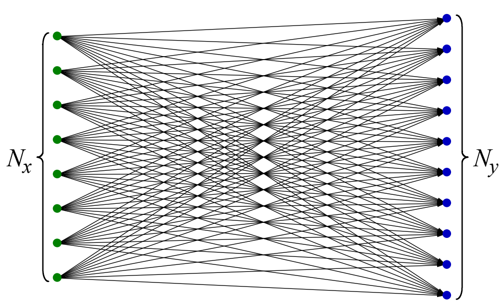


Figure 1 - direct interactions (conventional BEM)

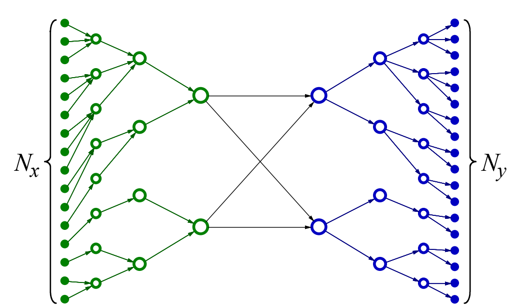


Figure 2 - cluster based interactions (MLFMM)

The decrease of the number of interactions when using the multi-level version of the algorithm can be identified by comparing Figure 1 and Figure 2.

[1] burgi@beuth-hochschule.de, ochmann@beuth-hochschule.de
[2] ingo5schaefer@bwb.org, bodonolte@bwb.org

The effects of all individual sources x_i within a source cluster C_x of radius R are summarized in a multipole source at z_x. Their potential is transformed to a remote target cluster C_y centered at z_y and distributed to the target points y_j (Figure 3).

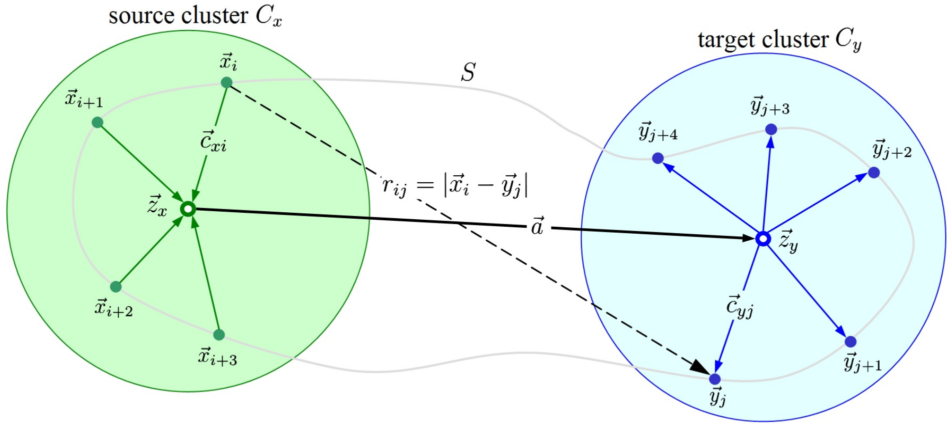


Figure 3 - Decomposition of the path between source and destination points of two clusters (based on [Sak4], p. 347, Fig. 12.9)

For each part of the path between source and destination points a corresponding transfer function (h,$\mu^M$and f) is used as shown in Figure 4, the full details and formulations may be found in [Sak4].

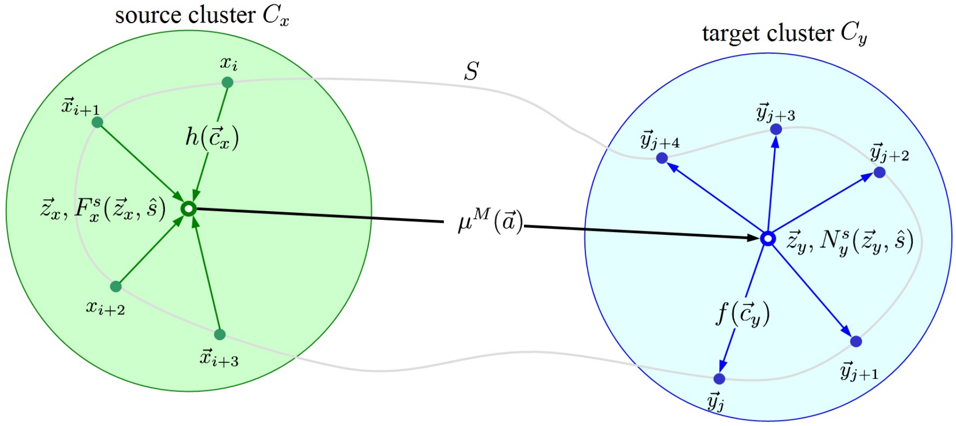


Figure 4 - path dependent transfer functions h, μ^M and f

One if the most critical functions here is the translation operator μ^M which "transfers" the multipole potential from a source cluster (the far field signature F_s) to the center of a target cluster with the near field signature N^s. This operator can be represented as a truncated series (Eq. (1)) with the maximum order set to O_{mp} ("multipole order"):

$$\mu^M(\vec{a},\hat{s}) = \frac{1}{4\pi}\sum_{l=1}^{O_{mp}}(2l+1)\,\mathrm{i}^l\,h_l(k\,|\vec{a}|)\,P_l(\hat{s}\cdot\hat{a}) \tag{1}$$

with	$\vec{a}$	distance vector between cluster centers
	$\hat{s}$	set of vectors on the unit sphere
	k	wave number
	$h_l(k\,|\vec{a}|)$	Hankel function
	$P_l(\hat{s}\cdot\hat{a})$	Legendre polynomials

There are many articles in literature dealing with the determination of the parameters used [Sak6, Yas7] and the specialties of the Hankel function h_l and Legendre polynoms P_l, but this would exceed the framework of this paper.

First we implemented a fixed-order version (Vs. 1.x) of the MLFMM algorithm [Bur1, Bur2] based on [Sak4] to check the limits with regard to element numbers and frequency. Then the algorithm was extended to an adaptive-level version (Vs. 2.x) using [Che5, Sak6, Che8] where the multipole order is defined as a function $O_{mp}(k, |\vec{a}|)$ and calculated at runtime. In addition, other optimizations to reduce the solution time were made, especially within the precalculation of geometry-based values.

2. Examples and results

2.1 Rigid sphere

To test the hardware dependent limits of the implemented MLFMM algorithm, a simple rigid sphere (r = 0.5 m, Figure 5) placed in water and hit by a plane wave using a frequency of 1 kHz was discretized with resulting element numbers between 1.000 and 5 millions.

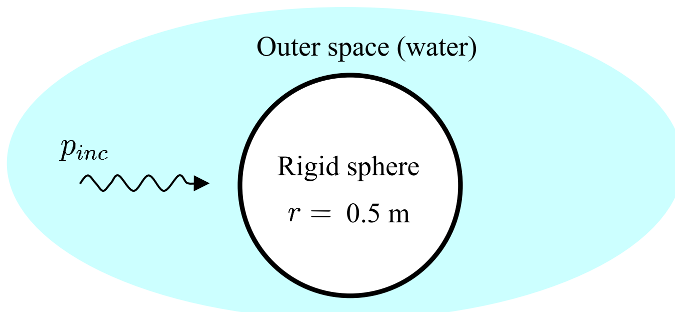


Figure 5 - rigid sphere

The resulting solving times of the fixed-order MLFMM algorithm (Vs. 1.020 and 1.030) are shown in Table 1 and Figure 6, using a fixed multipole order of O_{mp} = 6 and an iteration error of $e_{iter} \leq 10^{-10}$. All calculations ran on a 12 core workstation with 48 GB RAM, thus limiting the number of unknowns to ≈ 64,000 for a full matrix.

Table 1 - solving times for the rigid sphere using different methods

number of elements	direct solver	iterative solver (GMRES)		iterative solver w/MLFMM, Vs. 1.030			
N_{elem}	Δt_{solve}	Δt_{solve}	N_{iter}	Δt_{solve}	N_{iter}	Δt_{MVP1}	$\oslash\Delta t_{MVP}$
1k	0.20 s	0.16 s	12	0.64 s	13	0.093 s	0.045 s
2.5k	1.23 s	0.73 s	11	0.62 s	12	0.141 s	0.042 s
5k	7.44 s	3.03 s	11	2.06 s	13	0.328 s	0.142 s
10k	43.99 s	11.92 s	11	2.59 s	12	0.562 s	0.180 s
20k	310.75 s	52.56 s	11	7.63 s	13	1.202 s	0.530 s
50k	4,352.43 s	329.37 s	11	15.13 s	12	3.433 s	1.053 s
100k	n.a.	11,710.95* s	11	31.87 s	12	5.975 s	2.331 s
200k	n.a.	46,635.42* s	11	57.16 s	12	12.823 s	3.989 s
500k	n.a.	n.a.		138.26 s	12	28.205 s	9.869 s
1M	n.a.	n.a.		373.61 s	12	68.500 s	27.596 s
2M	n.a.	n.a.		561.82 s	12	114.973 s	40.332 s
5M	n.a.	n.a.		1,922.26 s	13	315.325 s	133.179 s

(a full matrix was rebuild for each matrix-vector product for results marked with an *)

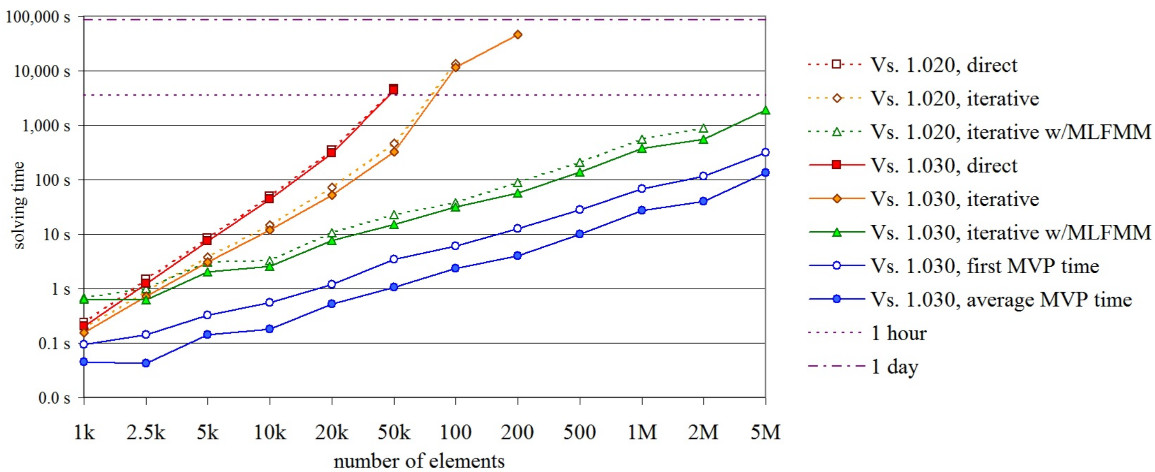


Figure 6 - solving times for the rigid sphere using different methods

These solution times were significantly reduced using the adaptive-order version (Table 2, Figure 7).

Table 2 - solving times for the rigid sphere using different MLFMM versions

elements	iterative solver w/MLFMM, Vs. 2.030				multipole order	
N_{elem}	Δt_{solve}	N_{iter}	Δt_{MVP1}	$\oslash\Delta t_{MVP}$	$O_{mp,min}$	$O_{mp,max}$
1k	0.29	13	0.071	0.015	6	10
2.5k	0.42	13	0.120	0.024	6	10
5k	0.91	12	0.319	0.047	6	10
10k	1.33	12	0.560	0.067	6	10
20k	2.57	12	1.115	0.127	6	10
50k	7.38	12	4.037	0.291	6	10
100k	12.79	12	6.557	0.543	6	10
200k	25.88	12	13.630	1.083	6	10
500k	63.74	12	33.053	2.661	6	10
1M	134.61	12	71.360	5.491	6	10
2M	257.12	12	140.226	10.100	6	10
5M	856.73	12	532.744	25.987	6	10

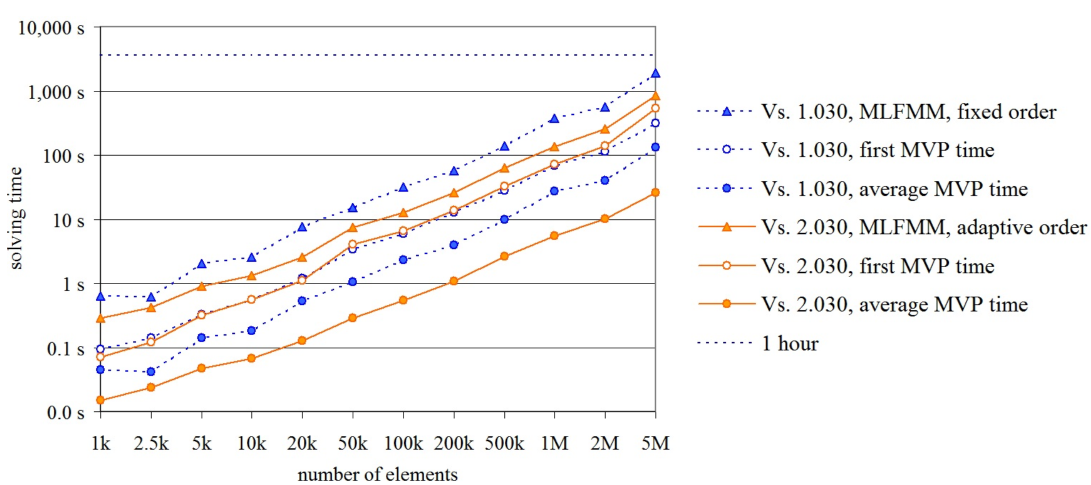
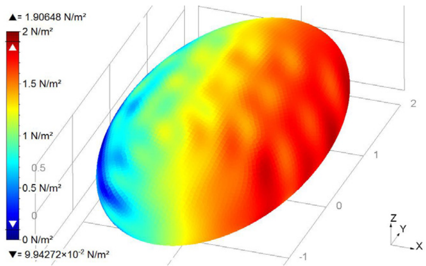


Figure 7 - solving times for the rigid sphere using different MLFMM versions

2.2 Ellipsoid

Another problem with the use of the fixed-order algorithm is that the solution due to the error in the matrix-vector products does not match the expected solution, although the iterative error was below a given value. To illustrate this problem, a rigid ellipsoid (2 × 4 × 1 m, N_{elem} = 17,300) placed in water is hit by a plane wave with at an incident angle of 30° using a frequency of f = 2.5 kHz.

Figure 8 shows the expected result using a conventional matrix-based BEM calculation.

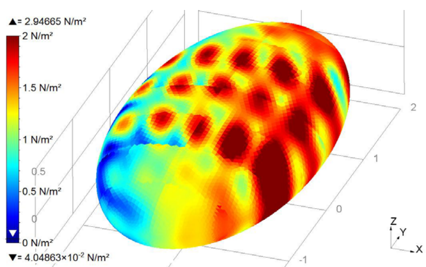


GMRES, matrix based
N_{iter}: 70
e_{iter}: $\leq 10^{-6}$
Δt_{solve}: 37.47 s (Windows)
30.90 s (LINUX)

direct solver, matrix based
Δt_{solve}: 139.50 s (Windows)
131.39 s (LINUX)

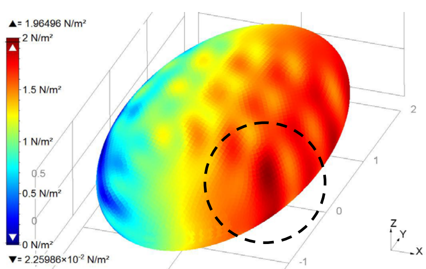
Figure 8 - Ellipsoid, $\text{abs}(p_{surf})$, matrix-based

This model is difficult for the MLFMM due to the different dimensions which require high multipole orders for correct results due to larger cluster distances. Figure 9 and Figure 10 show the results when using the fixed-order MLFMM algorithm.



GMRES with MLFMM
fixed multipole order
N_{iter}: 67
e_{iter}: $\leq 10^{-6}$
O_{mp}: 6
Δt_{solve}: 11.06 s (Windows)
8.67 s (LINUX)
Big differences, not usable!

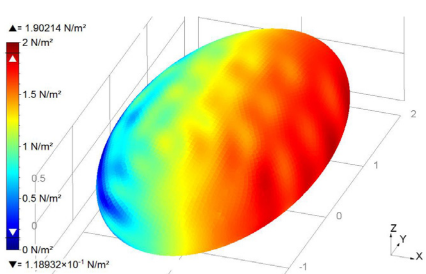
Figure 9 - Ellipsoid, $\text{abs}(p_{surf})$, MLFMM, fixed order of O_{mp} = 6



GMRES with MLFMM
fixed multipole order
N_{iter}: 71
e_{iter}: $\leq 10^{-6}$
O_{mp}: 15
Δt_{solve}: 36.93 s (Windows)
27.56 s (LINUX)
Some visible differences

Figure 10 - Ellipsoid, $\text{abs}(p_{surf})$, MLFMM, fixed order of O_{mp} = 15

The dependence of the quality on the multipole order O_{mp} is clear. The adaptive-level version gives the best performance and quality (Figure 11).



GMRES with MLFMM
adaptive multipole order
N_{iter}: 70
e_{iter}: $\leq 10^{-6}$
O_{mp}: 6…55
Δt_{solve}: 33.45 s (Windows)
28.17 s (LINUX)
Good agreement and performance!

Figure 11 - Ellipsoid, $\text{abs}(p_{surf})$, MLFMM, adaptive order O_{mp} = 6 ... 55

3. Conclusions and future work

The results presented show that the adaptive and performance-optimized version of the MFLMM algorithm gives better quality and faster results.

At higher frequencies, the MLFMM shows significant qualitative differences, since the error in the method-based matrix-vector product forms a stronger effect. Further investigations to optimize the code for higher multipole expansion orders are necessary.

Likewise, a suitable preconditioner should be used to achieve a better convergence of the iterative method. First results were published in [Och3], but these were not really successful, so further work is needed here.

Acknowledgments

This project was partially supported from 01/2010-12/2011 by the Institute for Applied Research Berlin I.S. (Institut für angewandte Forschung Berlin e.V.).

IFAF
Institut für angewandte Forschung Berlin
BERLIN

References and links

[Bur1] R. Burgschweiger, I. Schäfer and M. Ochmann, ”A Multi-Level Fast Multipole Algorithm (MLFMM) for calculating the Sound scattered from Objects within Fluids”, Proceedings of the 20th International Congress on Acoustics (ICA 2010), Sydney, Australia (2010)

[Bur2] R. Burgschweiger and M. Ochmann, ”A Multi-Level Fast Multipole BEM-Method for computing the sound field in rooms”, Proceedings of the Forum Acusticum, Aalborg, Denmark (2011)

[Och3] M. Ochmann, R. Burgschweiger and C. Steuck, ”Numerical experiments for testing the convergence of the acoustical Fast Multipole Method”, Proceedings of the 1st EAA Congress on Sound & Vibration (EuroRegio), Ljubljana (2010)

[Sak4] T. Sakuma, S. Schneider and Y. Yasuda, “Fast Solution Methods”, chap. 12, in ”Computational Acoustics of Noise Propagation in Fluids”, Springer Verlag, p. 333ff (2008)

[Che5] H. Cheng, L. Greengard and V. Rokhlin, “A Fast Adaptive Multipole Algorithm in Three Dimensions”, Journal of Computational Physics, Vol. 153, p. 468-498 (1999)

[Sak6] T. Sakuma and Y. Yasuda, “Fast Multipole Boundary Element Method for Large-Scale Steady-State Sound Field Analysis, Part I: Setup and Validation”, Acta Acustica united with Acustica, Vol. 88, p. 513–525 (2002)

[Yas7] Y. Yasuda and T. Sakuma, “Fast Multipole Boundary Element Method for Large-Scale Steady-State Sound Field Analysis, Part II: Examination of Numerical Items”, Acta Acustica united with Acustica, Vol. 89, p. 28-38 (2003)

[Che8] H. Cheng, W. Crutchfield, Z. Gimbutas, L. Greengard, J. Ethridge, J. Huang, V. Rokhlin, N. Yarvin and J. Zhao, “A wideband fast multipole method for the Helmholtz equation in three dimensions”, Journal of Computational Physics, Vol. 216, p. 300-325 (2006)

More information can be found at the website of our research group "computational acoustics":
http://projekt.beuth-hochschule.de/ca