Journal of Advances and Scholarly Researches in Allied Education
Vol. 13, Issue No. 1, April-2017, ISSN 2230-7540

An Analysis on Some Algorithms of Fast Runge-Kutta Convolution Quadrature

Seema Rani[1]* Dr. Ashwani Kumar[2]

[1]Research Scholar of OPJS University, Churu, Rajasthan

[2]Associate Professor, OPJS University, Churu, Rajasthan

Abstract – This work addresses the question of the efficient numerical solution of time-domain boundary integral equations with retarded potentials arising in the problems of acoustic and electromagnetic scattering. The convolutional form of the time-domain boundary operators allows to discretize them with the help of Runge-Kutta convolution quadrature. In the work it is shown how this property can be used in the recursive algorithm to construct only a few matrices with the near-field, while for the rest of the matrices the far-field only is assembled. The resulting method allows to solve the three-dimensional wave scattering problem with asymptotically almost linear complexity. The efficiency of the approach is confirmed by extensive numerical experiments.

----------X----------

INTRODUCTION

Our approach is conceptually different from the one used in the study of W. Hackbusch, W. Kress, and S. A. Sauter (2007). We construct the new method based on the algorithm of linear complexity, rather than back substitution of quadratic complexity. This approach allows us to avoid the actual evaluation of the convolution weights, thus enabling the use of fast techniques based on analytic expansions. Computational and storage costs of the improved algorithm scale linearly, up to logarithmic factors.

FAST RUNGE-KUTTA CQ ALGORITHM

Let us come back to the recursive algorithm. Recall that for this algorithm ***O(N)*** Galerkin discretizations of boundary single-layer operators for the Helmholtz equation with decay need to be constructed. A straightforward application of the data-sparse techniques (i.e. FMM and $\mathcal{H}$-matrices) would on its own lead to the algorithm of almost linear complexity. However, a significant drawback of this approach is large constants involved in complexity estimates. Our goal is to design a method that would reduce them.

The data-sparse techniques in question have two main bottlenecks:

- Costly evaluation of singular and nearly singular integrals in the near-field;
- High matrix-vector multiplication costs of the high-frequency FMM.

We overcome the first problem by the use of fast decay of convolution weights $w_n^h(d)$ away from the neighborhood of $d \approx nh$. We show that within the whole recursive algorithm only a few matrices (namely $O(\log N)$) with the near-field need to be constructed, while for the rest we can assemble the far-field only. To motivate this strategy.

In the end of this section we demonstrate that provided that for the approximation of different matrices a choice between $\mathcal{H}$-matrix techniques and the HF FMM is made properly, the problem of high matrix-vector multiplication costs of the IIF FMM ceases to exist.

MOTIVATION

The evaluation of the near-field integrals is commonly done with the help of coordinate transformation techniques. Given the kernel *k(x,y)* of a boundary single layer operator (that maps from $H^{-\frac{1}{2}}(\Gamma)$ to $H^{\frac{1}{2}}(\Gamma)$), the evaluation of

$$\iint_{\pi_i \times \pi_j} k(x,y)\phi_i(x)\phi_j(y)d\Gamma_x d\Gamma_y, \qquad \operatorname{supp}\phi_i = \pi_i,\ \operatorname{supp}\phi_j = \pi_j,$$

with the accuracy sufficient to preserve the stability and convergence of the Galerkin method, requires that the quadrature order scales as $O(\log^3 M)$ if $\operatorname{dist}(\pi_i, \pi_j) = 0, O(\log^4 M)$ if $\operatorname{dist}(\pi_i, \pi_j) = O(\Delta x)$ (nearly singular integrals) and 0 (1) if $\operatorname{dist}(\pi_i, \pi_j) = 1.$ Thus the computation of the near-field (singular and nearly singular integrals) of one matrix is of $O(M \log^4 M)$ complexity. Within the recursive convolution

quadrature algorithm $O(N)$ such matrices need to be assembled, hence resulting in the total complexity $O\left(NM\log^4 M\right)$.

The question of the efficient evaluation of singular and nearly singular integrals was addressed in recent works. Particularly, such integrals were represented as functions of multiple parameters and efficiently computed using interpolation and tensor decomposition techniques. The effect of the application of such techniques on the full $\mathcal{H}$-matrix assembly time was numerically studied. For the Laplace boundary single layer operator on various geometries it was demonstrated that the 50%-70% reduction of the time required for the evaluation of the nearly singular and weakly singular integrals results in the 10%-20% reduction of the total $\mathcal{H}$-matrix assembly time. Given the bound on the ranks of $\mathcal{H}$-matrix r, the rest of the time is spent for the evaluation of $O(rM\log M)$ far-field integrals within the ACA+ procedure of the $\mathcal{H}$-matrix construction. If the evaluation of the far-field is done in a more efficient manner, the gain can be significantly larger. And this is the case for the fast multipole methods.

The precomputation time (i.e. time needed for the construction of the translation operators) for the HF FMM scales as $O(M\log M)$ (assuming $M=O(|\kappa|^2)$ for the wavenumber $\kappa = is$) and the constants involved are significantly smaller than that for the $\mathcal{H}$-matrix assembly. This can be seen in the experiments of D. Brunner, M. Junge, P. Rapp, M. Bebendorf, and L. Gaul (2010), where the IIF FMM precomputation times were reported to be in practice 9-20 times smaller than that for the $\mathcal{H}$-matrix construction. This can be also observed in the numerical experiments in Section 2.3. In the study of M. Fischer (2004), the time to compute the near-field for the HF FMM accelerated Burton-Miller formulation is compared to the time needed to construct the corresponding IIF FMM translation matrices. The results show that for BEM discretizations with 10^3 - HP triangular boundary elements the computation of the near-field is typically order of magnitude slower than the assembly of translation matrices.

However the actual constants depend much on the implementation and the desired accuracy. Nevertheless, for large problems we should be able to see the improvement if we skip constructing the near-field. Asymptotic complexity estimates are improved as well. Indeed, while the application of ACA/ACA+ based $\mathcal{H}$-matrix techniques requires the evaluation of 4-dimensional integrals, for the use of the HF FMM in the far-field only the evaluation of two-dimensional integrals (for the cluster basis) is needed. We perform this step not during the precomputation stage, but when compute matrix-vector products (this allows to avoid storing the cluster basis for all matrices and thus improves memory costs). Therefore the relative improvement in the precomputation time if the near-field is not constructed is even more drastic.

Since in the course of the recursive algorithm, the matrix- vector multiplication with the same matrix block is performed multiple times, it makes sense to precompute the corresponding discretizations of boundary integral operators and keep them in memory, rather than recompute them every time the matrix-vector multiplication is needed. For the matrices that are approximated with the help of the fast multipole method the near-field and translation operators can be stored. If only a small part of matrices has the near-field, the storage costs needed for HF FMM approximated matrices can be affected as well. Given the HF FMM approximation of $\mathbf{V}(s)$, the storage for its far-field part (translation matrices of the FMM) scales as

$$S_{ff}(s) = O(|s|^2 \log M) = O(M \log M),$$

where we assumed $M = O\left(|s|^2\right)$, while for the near-field $S_{nf}(s) = O(M)$.

Hence, as $M \to +\infty$, S_{nf} is smaller than S_{ff} (though only by a logarithmic term). The improvement in the storage costs can be achieved only in the case when the constants in S_{ff} are so small that even for rather large M, $S_{nf} > S_{ff}$. As our numerical experiments in Section 4 show, in practice this is often the case.

The presence of decay, i.e. in the case when $s = s_1 + is_2,\ s_1 > 0$, facilitates the reduction of storage costs. If s_1 is large enough, for such discretizations $\mathbf{V}(s)$ the far-field part

$$S_{ff} \approx 0,$$

see also Figure 1.

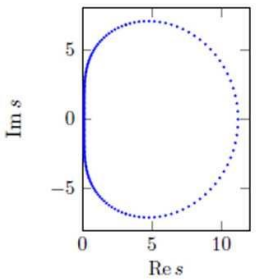

Figure 1: Frequencies s for which we need to construct discretizations of boundary single-layer operators $\mathbf{V}(\frac{s}{h})$; they are computed as eigenvalues of $\Delta(\xi),\ |\xi| = 10^{-\frac{6}{10s}}$. Here h=1. While many frequencies arc located close to the imaginary axis, a significant part of frequencies has $\mathrm{Re}\, s \gg 1$ (high-decay case). A large part of the far-field of the corresponding matrices $\mathbf{V}(\frac{s}{h})$ is negligibly small and

can be a priori ignored when constructing these matrices.

NEAR-FIELD REUSE

Auxiliary Relations on Leaves of a Block-Clustcr Tree -

Before describing our strategy for dealing with the near-field, we introduce two auxiliary relations defined on leaves of a block-cluster tree, namely the near-field d-admissibility and the far-field d-admissibility. Recall that given a cluster τ, the center of its bounding box we denote by c_τ and the diameter of the bounding box by d_τ,.

Definition 5.1. Given $d>0,$ we will call a leaf (τ,σ) near-field d-admissible if $\|c_\tau - c_\sigma\| < d - \frac{d_\tau}{2} - \frac{d_\sigma}{2}$.

Definition 1. Given $D > 0,$ a leaf (τ,σ) is far-field D-admissible if $\|c_\tau - c_\sigma\| < D + \frac{d_\tau}{2} + \frac{d_\sigma}{2}$.

Remark 1. The following properties hold:

(1) If (τ,σ) is near-field d-admissible then $(\forall x \in \Omega_\tau)(\forall y \in \Omega_\sigma),\ \|x-y\| < d.$

(2) If (τ,σ) is not far-field D-admissible then $(\forall x \in \Omega_\tau)(\forall y \in \Omega_\sigma),\ \|x-y\| > D.$

We will denote the set of near-field d-admissible leaves ofa block-cluster tree $\mathcal{T}_{\mathcal{I}\times\mathcal{I}}$ by $\mathcal{L}_d(\mathcal{T}_{\mathcal{I}\times\mathcal{I}})$ and the set of far-field D-admissible leaves by $\mathcal{L}_D^+(\mathcal{T}_{\mathcal{I}\times\mathcal{I}})$.

Remark 2. The following observation is crucial for our algorithm. Recall that $\mathcal{L}_-(\mathcal{T}_{\mathcal{I}\times\mathcal{I}})$ is defined as the set of all non-admissible block-clusters of the block-cluster tree $\mathcal{T}_{\mathcal{I}\times\mathcal{I}}$. Then it. is possible to choose d s.t.

$$\mathcal{L}_-(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}) \subset \mathcal{L}_d(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}). \tag{1}$$

This follows from the definition of the admissibility condition. Namely, given $\eta > 1,$ non-admissible leaves (τ,σ) satisfy

$$\|c_\tau - c_\sigma\| < \frac{\eta}{2}(d_\tau + d_\sigma),$$

where $c_\tau,\ c_\sigma$ are the centers of bounding boxes of $\tau,\ \sigma$ and $d_\tau,\ d_\sigma$ are their diameters. The choice

$$d = \gamma \sup_{(\tau,\sigma)\in\mathcal{L}_-} (d_\tau + d_\sigma), \qquad \gamma \geq \frac{\eta+1}{2} \tag{2}$$

ensures that (1) holds true.

Now we have all the ingredients needed to describe fast Runge-Kutta convolution quadrature.

Main Ideas and Algorithmic Realization-

Consider the matrix-vector product, namely

$$\begin{pmatrix} h_0 \\ h_1 \\ \vdots \\ h_{n-\ell} \end{pmatrix} = \begin{pmatrix} W_\ell^h & W_{\ell-1}^h & \cdots & W_1^h \\ W_{\ell+1}^h & W_\ell^h & \cdots & W_2^h \\ \vdots & & & \\ W_n^h & W_{n-1}^h & \cdots & W_{n-\ell+1}^h \end{pmatrix} \begin{pmatrix} \lambda_0 \\ \lambda_1 \\ \vdots \\ \lambda_{\ell-1} \end{pmatrix} \tag{3}$$

After the discretization in space with the help of the Galerkin method (with trial and test basis functions $(\phi_j(x))_{j=1}^M,$ the above system of equations can be rewritten as:

$$\begin{pmatrix} h_0^j \\ h_1^j \\ \vdots \\ h_{n-\ell}^j \end{pmatrix} = \iint_{\Gamma\times\Gamma} \sum_{k=1}^M T^{\ell,n}(\|x-y\|) \begin{pmatrix} \lambda_0^k \\ \lambda_1^k \\ \vdots \\ \lambda_{\ell-1}^k \end{pmatrix} \phi_k(y)\phi_j(x)d\Gamma_x d\Gamma_y, \qquad j=1,\ldots,M, \tag{4}$$

where

$$h_k^j = \int_\Gamma h_k(x)\phi_j(x)d\Gamma_x, \qquad k=0,\ldots,n-\ell,\ j=1,\ldots,M,$$
$$\lambda_k^j = \int_\Gamma \lambda_k(x)\phi_j(x)d\Gamma_x, \qquad k=0,\ldots,\ell-1,\ j=1,\ldots,M,$$

and $T^{\ell,n}$ is the kernel function

$$T^{\ell,n}(\|x-y\|) = \begin{pmatrix} w_\ell^h(\|x-y\|) & \cdots & w_1^h(\|x-y\|) \\ w_{\ell+1}^h(\|x-y\|) & \cdots & w_2^h(\|x-y\|) \\ \vdots & & \\ w_n^h(\|x-y\|) & \cdots & w_{n-\ell+1}^h(\|x-y\|) \end{pmatrix} \tag{5}$$

Let *d* he chosen as in (2). The double integral in (4) can be split into a sum of two double integrals: one over the leaves of the block-cluster tree belonging to the set $\mathcal{L}_d(\mathcal{T}_{\mathcal{I}\times\mathcal{I}})$ and the other being the remainder. Namely,

$$\iint_{\Gamma\times\Gamma} T^{\ell,n}(\|x-y\|)\phi_j(x)\phi_k(y)d\Gamma_x d\Gamma_y = N_{jk} + F_{jk}, \tag{5.6}$$

$$N_{jk} = \iint_{\Omega_\sigma\times\Omega_\tau,\ (\sigma,\tau)\in\mathcal{L}_d} T^{\ell,n}(\|x-y\|)\phi_j(x)\phi_k(y)d\Gamma_x d\Gamma_y, \tag{5.7}$$

$$F_{jk} = \iint_{\Omega_\sigma\times\Omega_\tau,\ (\sigma,\tau)\in\mathcal{L}_F} T^{\ell,n}(\|x-y\|)\phi_j(x)\phi_k(y)d\Gamma_x d\Gamma_y, \qquad j,k=1,\ldots,M,$$

where $\mathcal{L}_d = \mathcal{L}_d(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}),\ \mathcal{L}_F = \mathcal{L}(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}) \setminus \mathcal{L}_d(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}).$ In this case $\mathcal{F} = (F_{jk})_{k,j=1}^M$ doesnot contain the near-field, since all non-admissible block-clusters belong to $\mathcal{N} = (N_{jk})_{k,j=1}^M.$ Since w_j^h are matrix-valued functions, N_{jk} and F_{jk} are tensors.

First, we demonstrate why such splitting may improve storage and computational costs. The bounds show that, for any given $\epsilon > 0,$ there exists *L*,

$$\left\|w_j^h(\tilde{d})\right\| < \frac{\epsilon}{4\pi\tilde{d}^3}, \text{ for all } j \geq L \text{ and } \tilde{d} < d. \tag{8}$$

Let

$$\Omega_d = \bigcup_{(\sigma,\tau)\in\mathcal{L}_d} \Omega_\sigma \times \Omega_\tau.$$

We assume w.l.o.g. that

$$L < \min(\ell, n-\ell+1). \tag{9}$$

Then some of the elements of the tensor $\mathcal{N}$ are approximately equal to zero. Let us show

this. For $k \geq L$,

$$\left|\iint_{\Omega_d} w_k^h(\|x-y\|)\phi_j(x)\phi_i(y)d\Gamma_x d\Gamma_y\right| < \epsilon\left|\iint_{\Omega_d}\frac{|\phi_j(x)\phi_i(y)|}{4\pi\|x-y\|}d\Gamma_x d\Gamma_y\right| \leq \epsilon\iint_{\Gamma\times\Gamma}\frac{|\phi_j(x)\phi_i(y)|}{4\pi\|x-y\|}d\Gamma_x d\Gamma_y,\ i,j=1,\dots M.$$

Recall that the boundary single-layer operator for the Laplacian is continuous from $L_2(\Gamma) \to L_2(\Gamma)$. Hence, for some $C>0$ that depends only on Γ it holds:

$$\left|\iint_{\Omega_d} w_k^h(\|x-y\|)\phi_j(x)\phi_i(y)d\Gamma_x d\Gamma_y\right| \leq C\epsilon\|\phi_i\|_{L_2(\Gamma)}\|\phi_j\|_{L_2(\Gamma)} = C\epsilon\mu_i\mu_j, \quad i,j=1,\dots,M, \tag{5.10}$$

where

μ_i= meas(supp(ϕ_i)), i, *j* = 1,... *M.*

Then *e* can always be chosen so that up to a desired precision $\mathcal{N}$ can be rewritten as

$$N_{jk} \approx \iint_{\Omega_d} T_L^{\ell,n}(\|x-y\|)\phi_k(y)\phi_j(x)d\Gamma_x d\Gamma_y, \quad k,j=1,\dots,M, \tag{5.11}$$

where

$$T_L^{\ell,n}(\|x-y\|) = \begin{pmatrix} 0 & \cdots & w_{L-1}^h(\|x-y\|) & \cdots & w_2^h(\|x-y\|) & w_1^h(\|x-y\|) \\ 0 & \cdots & 0 & \cdots & w_3^h(\|x-y\|) & w_2^h(\|x-y\|) \\ \vdots & & & & & \\ 0 & \cdots & 0 & \cdots & w_{L-1}^h(\|x-y\|) & w_{L-2}^h(\|x-y\|) \\ 0 & \cdots & 0 & \cdots & 0 & w_{L-1}^h(\|x-y\|) \\ \vdots & & & & & \\ 0 & \cdots & 0 & \cdots & 0 & 0 \end{pmatrix} \tag{12}$$

Hence, to approximate completely the near-field part of the matrix of the system (3), only *O(L)* Galerkin matrices

$$\left(\widetilde{W}_\nu^h\right)_{kj} = \iint_{\Omega_d} w_\nu^h(\|x-y\|)\phi_k(y)\phi_j(x)d\Gamma_x d\Gamma_y, \quad k,j=1,\dots,M,\ \nu=1,\dots,L-1,$$

need to be constructed. In practice we do not assemble these matrices, but rather evaluate the matrix-vector product with $\mathcal{N}$ with the help of either of two procedures we present below. Before describing these procedures, we would like to show that

$$L = O\left(\log\frac{1}{\epsilon}\right)$$

and does not depend on the size of the system. Recall that the diameter of non- admissible clusters

$$d_\tau = O(\Delta x),$$

where Δx is the meshwidth (this is by construction of the admissible block-cluster tree). Hence, by (2), for some $\gamma' > 0$,

$$d = \gamma'\Delta x.$$

Since $\Delta x \approx Ch$, for some $C>0$,

$$d = \tilde{\gamma}h,\ \tilde{\gamma} > 0.$$

Importantly, $\tilde{\gamma}$ is constant and does not depend on *h* and Δx. The estimate on *L* can be obtained, choosing a priori $L \geq \frac{d}{h} = \tilde{\gamma}$. Namely, there exist constants $\delta, G, A, \beta > 0$, s.t.

$$\left\|w_k^h(d')\right\| \leq \frac{G}{d'}(1-\delta)^{k-\frac{d'}{h}}(1+A\delta^\beta)^{\frac{d'}{h}}, \text{ for all } d' < d, \text{ and } k \geq \frac{d}{h}.$$

Then *L* can be estimated from:

$$\frac{G}{d}(1-\delta)^{L-\frac{d}{h}}(1+A\delta^\beta)^{\frac{d'}{h}} < \frac{\epsilon}{4\pi d},$$
$$G(1-\delta)^{L-\tilde{\gamma}}(1+A\delta^\beta)^{\tilde{\gamma}} < \frac{\epsilon}{4\pi}.$$

From this it follows that for a fixed accuracy ϵ

$$L = O\left(\log\frac{1}{\epsilon}\right),$$

Remark 3. Increasing the value of d allows to reuse a pan of the far-field as well.

Remark 4. We do not address here the question how $\epsilon > 0$ has to be chosen to preserve the stability and convergence of the method. A full analysis would require the combination of the estimates of L. Banjai and S. Sauter (2008). In particular, it is shown that the convergence of the sparse DDF2 convolution quadrature is preserved if the convolution weights are cut off with the accuracy e satisfying $\log\frac{1}{\epsilon} = O(\log M) = O(\log N)$. We expect similar estimates to hold for our case as well, since all the errors are linear, and bounds for the errors and operator norms depend on h, Δx polynomially or as powers (positive or negative) of h, Δx.

Next the question of the efficient evaluation of a matrix vector product with the system (5.11) is addressed. We suggest the use of either of two methods.

Remarks on the Application of Data-Sparse Techniques and Paral- lellization

In this section we would like to address several questions on the application of data-sparse techniques, $\mathcal{H}$-matrices and the high-frequency fast multipole method, for approximating Galerkin discretizations of boundary integral operators in the course of the recursive algorithm. Recall that the setup time (i.e. the matrix assembly time) of $\mathcal{H}^2$-matrices that use expansions coming from the HF FMM is much smaller than that of $\mathcal{H}$-matrices. However, the corresponding IIF FMM accelerated matrix-vector multiplications are significantly slower than the matrix-vector multiplications with $\mathcal{H}$-matrices, even for discretizations with about 10° boundary elements .

The structure of the system of equations we need to solve is shown in Figure 5.2.

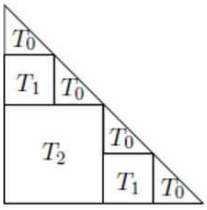

Figure 2: The structure of the matrix of the convolution quadrature system of equations.

The solution of the small triangular system of size *J* (where the matrix to is involved) has to be performed $O\left(\frac{N}{J}\right) = O(N)$ times. Since this operation requires the construction of only a few matrices and performing many matrix-vector multiplications with them, it makes scene to approximate these matrices by $\mathcal{H}$-matrix techniques. Additionally, matrix-vector multiplications with matrix blocks at the lower levels of the recursive algorithm (in the figure these blocks are marked by Ti) need to be performed more often than that with the matrix blocks located at the higher levels of the recursive algorithm (To). Hence, for large problems it is reasonable to employ pure $\mathcal{H}$-matrix approximations in this case. For the rest of the Toeplitz blocks the choice whether $\mathcal{H}$- or $\mathcal{H}+\mathcal{H}^2$ -based approximation is to be used is done. The advantage of the recursive algorithm is its easy parallelizabilitv. The precomputation of Galerkin discretizations of boundary integral operators can be done independently in

FAST CQ ALGORITHM AND ITS COMPLEXITY

In this section the fast Runge-Kutta convolution quadrature algorithm is described. Compared to the conventional recursive algorithm, the multiplication with Toeplitz matrix blocks is replaced by the improved procedure.

We substitute the procedure Multiply for the multiplication of the following matrix- vector product

$$\begin{pmatrix} h_\ell \\ h_{\ell+1} \\ \vdots \\ h_{\ell+n-m} \end{pmatrix} = \begin{pmatrix} W_m^h & W_{m-1}^h & \cdots & W_1^h \\ W_{m+1}^h & W_m^h & \cdots & W_2^h \\ \vdots & & & \\ W_n^h & W_{n-1}^h & \cdots & W_{n-\ell+1}^h \end{pmatrix} \begin{pmatrix} \lambda_p \\ \lambda_{p+1} \\ \vdots \\ \lambda_{p+m-1} \end{pmatrix}$$

by the two procedures.

MultiplyNF $(m,\ n,\ p,\ \ell,\ \lambda,\ \tilde{h})$-performs the matrix-vector multiplication with the near-field:

$$\begin{pmatrix} \tilde{h}_\ell^j \\ \tilde{h}_{\ell+1}^j \\ \cdots \\ \tilde{h}_{\ell+n-m}^j \end{pmatrix} = \sum_{k=1}^{M} N_{jk} \begin{pmatrix} \lambda_p^k \\ \lambda_{p+1}^k \\ \cdots \\ \lambda_{p+m-1}^k \end{pmatrix}, \qquad j = 1, \ldots, M.$$

MultiplyFF $(m,\ n,\ p,\ \ell,\ \lambda,\ \bar{h})$- performs the matrix-vector multiplication with the far-field:

$$\begin{pmatrix} \bar{h}_\ell^j \\ \bar{h}_{\ell+1}^j \\ \cdots \\ \bar{h}_{\ell+n-m}^j \end{pmatrix} = \sum_{k=1}^{M} \iint\limits_{\substack{\Omega_\sigma \times \Omega_\tau, \\ (\sigma,\tau) \in \mathcal{L}_{D_n,F}^+}} \begin{pmatrix} w_m^h(\|x-y\|) & \ldots & w_1^h(\|x-y\|) \\ w_{m+1}^h(\|x-y\|) & \ldots & w_2^h(\|x-y\|) \\ \cdots & & \\ w_n^h(\|x-y\|) & \ldots & w_{n-m+1}^h(\|x-y\|) \end{pmatrix} \times$$

$$\times \begin{pmatrix} \lambda_p^k \\ \lambda_{p+1}^k \\ \cdots \\ \lambda_{p+m-1}^k \end{pmatrix} \phi_j(x)\phi_k(y)d\Gamma_x d\Gamma_y, \qquad j = 1, \ldots, M.$$

Let the parameter *J* be fixed: every system of size smaller than *J* is to be solved directly.

function Solve $(n_0,\ n_1,\ \lambda,\ g)$

if $(n_1 - n_0 \le J)$ then

SolveBasici $(n_0,\ n_1,\ \lambda,\ g)$;

Else

$n_{\frac{1}{2}} = \left\lceil \frac{n_0 + n_{\frac{1}{2}}}{2} \right\rceil$;

Solve $(n_0,\ n_{\frac{1}{2}},\ \lambda,\ g)$;

MultiplyNF $(n_{\frac{1}{2}} - n_0 + 1,\ n_1 - n_0,\ n_0,\ n_{\frac{1}{2}} + 1,\ \lambda,\ h_1)$;

MultiplyFF $(n_{\frac{1}{2}} - n_0 + 1,\ n_1 - n_0,\ n_0,\ n_{\frac{1}{2}} + 1,\ \lambda,\ h_2)$;

$g|_{n_{\frac{1}{2}}+1,\ldots,n_1} = g|_{n_{\frac{1}{2}}+1,\ldots,n_1} - h_1|_{n_{\frac{1}{2}}+1,\ldots,n_1} - h_2|_{n_{\frac{1}{2}}+1,\ldots,n_1}$;

Solve $(n_{\frac{1}{2}} + 1,\ n_1,\ \lambda,\ g)$;

end if

endFunction

Let us disciuss the complexity of this algorithm based on the preliminary estimates. Compared to the conventional recursive algorithm, the new algorithm performs an extra block matrix-vector multiplication with the near-field matrices. The computational complexity of each of matrix-vector multiplication with the near-field matrices is either $O(L \log LM) = O(\log N \log \log NM)$ (if the near-field matrix-vector multiplication with the diagonalization is used) or $O(L^2 M) = O(\log^2 NM)$, see Remark 5.4. Totally, there are $O(N)$ matrix blocks (4), hence the total complexity of the near-field related matrix-vector multiplications is $O(N \log^2 NM)$.

The number of matrix-vector multiplications with the far-field matrices is $O(N \log N)$, while each of this matrix-vector multiplications requires about $O(M \log M)$ operations (here the hidden constant depends on the accuracy of the approximation. Hence, the total complexity of the algorithm is

$$O(N \log NM \log M + N \log^2 NM) = O(NM \log^2 M).$$

The memory costs for the near-field matrices scale linearly, $O(M \log N)$, while for the rest of the matrices as $O(NM \log M)$. As before, for the matrices with the far-field in this complexity estimate there is a hidden constant that depends on the accuracy of the approximation.

The construction times for $\mathcal{H}$-matrices scale as $O(NT_q M \log M)$, where T_q is the com plexity of the evaluation of the integrals in BEM, see also the discussion . Since we use the technique, T_q scales not worse than $O(\log^\alpha M)$, for $\alpha \geq 0$ (in our case $T_q = O(\log^4 M)$). The construction times for $\mathcal{H}^2$-matrices scale as $O(NM \log M)$. The hidden constants in these complexity estimates depend on the accuracy of the matrix approximations.

Combined with the use of data-sparse techniques and the complexity estimates, the computational complexity of the Solve procedure is not worse than $O(NM \log^2 M)$, the time to construct the matrices $O(NM \log M \log^k M)$, for $k > 1$, and the storage costs are $O(NM \log M)$.

CONCLUSION

In this work we built up a quick recursive Runge-Kutta convolution quadrature algorithm for the solution of the wave scattering problem in three dimensions. This method requires the construction of Galerkin discretizations of boundary integral administrators for the Helmholtz equation with rot.

Fast Runge-Kutta convolution quadrature is based on two ingredients: the use of fast data-sparse techniques, namely the high-frequency fast multipole method and H-matrices, and decay properties of Runge-Kutta convolution weights (that are the consequence of the Huygens principle). The use of the data-sparse techniques allows to solve the scattering problem in almost linear time. Exponentially fast decay of convolution weights $w_n^h(d)$ away from the neighborhood of $d \approx nh$ allows to skip constructing the diagonal and near-diagonal matrix blocks for most of the boundary integral operator discretizations, thus avoiding the evaluation of many singular and near-singular BEM integrals.

REFERENCES

Books-

M. Schanz (2001b). Wave Propagation in Viscoelastic and Poroelastic Continua: A Boundary Element Approach, vol 2 of Lecture Notes in Applied Mechanics. Springer-Verlag, Berlin, Heidelberg, New York.

P.W. Partridge, C.A. Brebbia, and L.C.Wrobel (1992). The Dual Reciprocity Boundary Element Method. Computational Mechanics Publication, Southampton.

Sauter, S. A., and Schwab, C. (2013). Boundary Element Methods, vol. 39 of Springer Series in Computational Mathematics. Springer, Berlin.

Research Papers-

Brunner, M. Junge, P. Rapp, M. Bebendorf, and L. Gaul (2010). Comparison of the fast multipole method with hierarchical matrices for the Helmholtz-BEM, CMES: Computer Modeling in Engineering & Sciences, 58, pp. 131-160.

Ch. Lubich and A. Ostermann (1993). Runge-Kutta methods for parabolic equations and convolution quadrature, Mathematics of Computation, 60, pp. 105-131.

L. Banjai and S. Sauter (2008). Rapid solution of the wave equation in unbounded domains, SIAM J. Numerical Analysis, 47 (2008), pp. 227-249.

R. Laliena and F.-J. Sayas (2009). Theoretical aspects of the application of convolution quadrature to scattering of acoustic waves, Numer. Math., 112, pp. 637-678.

W. Hackbusch, W. Kress, and S. A. Sauter (2007). Sparse convolution quadrature for time domain boundary integral formulations of the wave equation by cuto_ and panel-clustering, 29, pp. 113-134.

Thesis/Dissertations –

M. Fischer (2004). The Fast Multipole Boundary Element Method and its Application to Structure-Acoustic Field Interaction, PhD thesis, University of Stuttgart.

Corresponding Author

Seema Rani*

Research Scholar of OPJS University, Churu, Rajasthan

E-Mail –