Computing 64, 367–380 (2000)

Computing

Incomplete Cross Approximation in the Mosaic-Skeleton Method

E. E. Tyrtyshnikov,* Moscow

Received February 13, 1999; revised October 26, 1999

Abstract

The mosaic-skeleton method was bred in a simple observation that rather large blocks in very large matrices coming from integral formulations can be approximated accurately by a sum of just few rank-one matrices (skeletons). These blocks might correspond to a region where the kernel is smooth enough, and anyway it can be a region where the kernel is approximated by a short sum of separable functions (functional skeletons). Since the effect of approximations is like that of having small-rank matrices, we find it pertinent to say about *mosaic ranks* of a matrix which turn out to be pretty small for many nonsingular matrices.

On the first stage, the method builds up an appropriate mosaic partitioning using the concept of a tree of clusters and some extra information rather than the matrix entries (related to the mesh). On the second stage, it approximates every allowed block by skeletons using the entries of some rather small cross which is chosen by an adaptive procedure. We focus chiefly on some aspects of practical implementation and numerical examples on which the approximation time was found to grow almost linearly in the matrix size.

AMS Subject Classifications: 65F05, 65F30, 65F50.

Key Words: Matrix approximation, low-rank matrices, mosaic ranks, mosaic block partitioning, integral equations, asymptotically smooth functions.

1. Introduction

The mosaic-skeleton method [22, 23] was bred in a simple observation that rather large blocks in very large matrices coming from integral formulations can be approximated accurately by a sum of just few skeletons (some say dyads or rank-one matrices). These blocks might correspond to a region where the kernel is smooth enough, and anyway it can be a region where the kernel is approximated by a short sum of separable functions (in other words, functional skeletons). From a historical point of view, we may note that the earliest mention of low-rank blocks in dense matrices we are aware of was made in [27] (yet in the context entirely different from ours).

The mosaic-skeleton approximations are easy to result in fast approximate matrix-vector multiplication algorithms close by nature to those of multipoles [11,

* Supported by the RFBR Grant 99-01-00017 and Volkswagen-Stiftung Grant VW I/71 493.

16, 17, 20, 21], panel clustering [13], interpolation [2, 3, 18], and, to some extent, wavelet-based approaches [1, 14] (we apologize for not mentioning here many other important works related to these utterly topical fields). All the said techniques involve some hierarchy of interface regions and function approximants. What the mosaic-skeleton method differs in from others is a matrix analysis view on largely the same problem. Such a view can be very useful due to the generality of matrix theory approaches. Since the effect of approximations is like that of having small-rank matrices, we find it pertinent to say about *mosaic ranks* of a matrix which turn to be pretty small for many nonsingular matrices.

From a practical point of view, the mosaic-skeleton method is the only one that works explicitly with the entries of a matrix. It is crucial that it works only with a small part of the entries. These are entries of some cross in every block allowing for a low-rank approximation. First of all, we need to find the list of such blocks (and other blocks as well). On this stage, we fall back to the concept of a tree of clusters due to Hackbusch and Novak [13]. Further development of this concept was recently presented in [12]. Apart from the entries, however, we need some extra information related to the mesh. In this sense, I rather like to say that we discuss a *grey box* solver, in contrast to *black box* solvers, for large dense unstructured matrices (T. Chan told me that he already used this term, luckily in the same sense).

The first stage of the method is discussed in Section 3. Then, in Section 4, we get to the second stage where (allowed) blocks are to be approximated by skeletons. It is done using the entries of some rather small cross in the block. Since only a part of the whole block is involved, we call the approach an incomplete cross approximation. The cross is selected by an adaptive procedure involving more (yet not many) rows and columns step by step. On the whole, the procedure is inspired by the concept of maximal volumes which is of tremendous value in the approximation theory and now adopted to matrix approximation problems. In Section 5, we present numerical examples showing that the solution time depends almost linearly on the matrix size.

In spite of our intention to focus here only on practical aspects, in the next section we begin still with a brief discussion why and when mosaic ranks appear to be small. We discuss estimates on mosaic ranks with a special stress on some important assumptions worthy to be put explicitly (though it is not the case in many papers on compression strategies). We also present some relations between mosaic ranks of matrices and their inverses.

2. Mosaic Ranks

Consider a matrix A of size $m \times n$, and let A_k be a submatrix which is $m_k \times n_k$. Denote by $\Pi(A_k)$ a matrix of the same size as A with zeroes in the positions that are not occupied by A_k. A finite set of submatrices A_k is called a *covering* of A if $A = \sum_k \Pi(A_k)$, and a *mosaic partitioning* if the submatrices have no common entries. If A_k is of rank r_k, then A_k is a sum of k skeletons (rank-one matrices), and the memory for retaining A_k is mem $(A_k) = \min\{r_k(m_k + n_k), m_k n_k\}$. The mosaic

rank associated with the given mosaic partitioning is defined as [23] $\mathrm{mr}(A) = \sum_k \mathrm{mem}(A_k)/(m+n)$. If $m = n$ and A is of rank r, then we can store A using only $2rn$ memory cells. The same holds true if A is a nonsingular matrix and r is its mosaic rank.

We are also interested in the case when A_k are approximated by matrices $\tilde{A}_k$ of rank r_k with relative accuracy ε. If the Frobenius norm is used, then $||A - \sum_k \Pi(\tilde{A}_k)||_F \le \varepsilon\, ||A||_F$. In such cases, we consider *approximate mosaic ranks* (ε-ranks). Further on, if A denotes the whole coefficient matrix in the Galerkin or collocation methods, then $m = n$.

Very large nonsingular matrices coming from integral equations are usually dense and unstructured. Nevertheless, they may be approximated by matrices of low mosaic rank.

For example, consider two typical single layer potential equations related to the Dirichlet boundary value problems for the Laplace ($\Delta w = 0$) and Helmholtz ($(\Delta + k^2)w = 0$) equations in two dimensions. The first one is

$$-\frac{1}{2\pi} \int_{\partial\Omega} \log|x-y| U(y) ds(y) = F(x), \quad x \in \partial\Omega, \tag{1}$$

and the second is

$$\frac{\mathbf{i}}{4} \int_{\partial\Omega} H_0^{(1)}(K|x-y|) U(y) ds(y) = F(x), \quad x \in \partial\Omega, \tag{2}$$

here $ds(y)$ is the arclength element (theory is especially simple if $\partial\Omega$ is an infinitely smooth closed curve cutting the plane into two parts, but more general curves are treated as well in practice). The kernels for the equations are the fundamental solutions for the Laplace and Helmholtz equations in two dimensions. $H_0^{(1)}$ is the Hankel function of order 0 of the first kind. We assume that Ω is such that (1) and (2) have only trivial solution in case $F(x) = 0$.

Let $\partial\Omega$ be an ellipse with half-axes $a = 1$ and $b = 0.5$. We use the Galerkin method with piecewise constant functions. Let us see how approximate mosaic ranks behave as n increases. In Table 1, there are results computed for Eq. (1) with $\varepsilon = 10^{-4}$. We also output the *compression factor* which is the total memory (that would have been needed to keep the original matrix) over the memory used to keep skeletons of the mosaic approximations. In Table 2, there are results computed for Eq. (2) with $\varepsilon = 10^{-3}$; here $K = 1$ and the ellipse parameters are $a = 1$ and $b = 0.25$.

As is seen from Table 1, in matrix-vector multiplications we can work with the nonsingular matrix of order $n = 32768$ as if its classical rank were about 106. It is equal to say that the memory used is equal to 0.65 of n^2.

Table 1. Mosaic ranks for the logarithmic-kernel equation

Matrix order	512	1024	2048	4096	8192	16384	32768
Mosaic rank	63.46	71.49	78.46	86.84	93.60	100.84	107.10
Compression factor	24.79%	13.96%	7.66%	4.24%	2.29%	1.23%	0.65%

Table 2. Mosaic ranks for the Hankel-kernel equation

Matrix order	256	512	1024	2048	4096
Mosaic rank	46.08	53.76	61.44	71.68	81.92
Compression factor	36%	21%	12%	7%	4%

An important observation from the tables is that the compression factor becomes about twice smaller as n increases twice. This means that the memory and arithmetic work for the matrix-vector multiplication manifest *about linear* behavior in n.

It is important to understand why and when this is the case. On the whole, it is sufficient to do this for matrices of the form

$$A_n = [f(x_{in}, y_{jn})], \quad 1 \le i, \quad j \le n,$$

where $f(x, y)$ is a function of x and y from a bounded region S in the μ-dimensional space, and x_{in} and y_{jn} are the nodes of some meshes.

Assumption 1. Let f be *asymptotically smooth* in the sense that there exist $c, d > 0$ and a real number g such that, for $x \neq y$ and all integer $p \ge 0$,

$$|\partial^p f(x, y)| \le c_p |x - y|^{g-p} \tag{3}$$

where

$$c_p \le c d^p p!. \tag{4}$$

Here, ∂^p is any p-order derivative in $y = (y_1, \ldots, y_\mu)$:

$$\partial^p = \left(\frac{\partial}{\partial y_1}\right)^{i_1} \cdots \left(\frac{\partial}{\partial y_\mu}\right)^{i_\mu}, \quad i_1 + \cdots + i_\mu = p.$$

Assumption 2. Let the meshes be *quasi-uniform* in the sense that there are positive constants c_1 and c_2 such that

$$c_1 \frac{\operatorname{mes} S'}{\operatorname{mes} S} n \le \tau_n(S') \le c_2 \frac{\operatorname{mes} S'}{\operatorname{mes} S} n,$$

where S' is any subregion of S and τ_n counts how many nodes of the mesh with n nodes fall into S', and mes is the Riemann measure (thus, S and S' are assumed measurable; we could confine ourselves to those S' which are intersections of a finitely many cubes with S).

Theorem 1. [23] *Under Assumptions 1 and 2, for any $\delta > 0$ there are splittings $A_n = T_n + R_n$ where*

$$\operatorname{mr} T_n = O(\log^{\mu+1} n), \quad ||R_n||_F = O(n^{-\delta}). \tag{5}$$

Note that the concept of asymptotical smoothness in the sense of (3) was introduced by Brandt [2]. However, any proof for compression strategies would be incomplete if we do not say how c_p may grow in p. Thus, (4) should be regarded as an essential complement to (3). (It appeared explicitly, likely first, in [23].)

Theorem 1 can be generalized in several ways. First of all, we can substitute Assumption 1 with the following.

Assumption 1′. Assume that for any y_0 there are $\alpha(p)$ functions $u_i(x, y_0)$ and $v_i(y, y_0)$ such that, whenever $||y - y_0|| < ||x - y_0||$,

$$f(x, y) = \sum_{i=0}^{\alpha(p)} u_i(x, y_0) v_i(y, y_0) + E_p, \tag{6}$$

where

$$|E_p| \le \beta(p) \left(\frac{||y - y_0||}{||x - y_0||} \right)^p. \tag{7}$$

Moreover, for some positive c, d, and γ,

$$\alpha(p) \le c\, p^\gamma, \quad \beta(p) \le c\, d^p. \tag{8}$$

Theorem 2. *Under Assumptions 1′ and 2, for any $\delta > 0$ there are splittings $A_n = T_n + R_n$ where*

$$\operatorname{mr} T_n = O(\log^{\gamma+1} n), \quad ||R_n||_F = O(n^{-\delta}). \tag{9}$$

We omit the proof because, on the whole, it would repeat part of the proof from [23].

In comparison with Theorem 1, the new formulation has at least two advantages. First, with some special expansions other than the Taylor series, it might be proved, sometimes, that $\gamma < \mu$, and this leads to a finer estimate. Second, some oscillatory kernels as that of Eq. (2) are not asymptotically smooth. It is still possible to prove that these kernels fulfil Assumption 1′ [7]. Unfortunately, the mosaic ranks for the Hankel-kernel equation depend on the wave number K linearly; for oscillatory kernels in the 3D case, they depend on K even quadratically [7].

We can not do without any assumption on meshes. However, Assumption 2 seemed to be essential chiefly for the algebraic technique proposed and used in [23], and we always thought that it could be weakened. Now it is done in [7]. More precisely, the lower estimate on $\tau_n(S')$ is no longer in need. Consequently, S could be a closed curve on the plane or a surface in the 3D case [7].

In practice and in our numerical illustrations, matrices may be produced, for example, by the Galerkin method. Note that the above theorems could be extended to the cases of standard boundary or volume finite-element techniques, because "allowed" blocks are approximated by a product of the form PFQ, where F consists of the kernel's values at some mesh (P and Q depend on the test functions), and hence, if F is of low rank, then the same applies to PFQ.

Now, we finish the discussion of theoretical estimates (which is not the main purpose in this paper) and get to practical issues.

3. Mosaic Partitionings

On the first stage of the mosaic-skeleton method, we are to choose a suitable mosaic partitioning. To do this, we do not compute any entries of the given matrix. Instead of this, we rely on some (rather weak indeed) geometrical information.

We require that every entry is associated with two points, x_i and y_j, in the m-dimensional space. We think it is sufficient to have one mesh instead of two ($x_i = y_i$), though we could keep the two if necessary.

We call a *cluster* any subset of nodes $x_1, \ldots, x_n$ furnished with a finite (independent of the number of nodes) set of attributes. In the present algorithms, we use only two attributes: the center c, defined as the mean radius-vector for the nodes, and radius r, defined as the maximal distance between c and the nodes. Prior to forming the list of blocks we construct a *tree of clusters* suggested in [13] and recently developed in [12].

The root of this tree is the cluster containing all the nodes. Apart from the nodes, there are two input parameters in our algorithm: the maximal number of levels $L_{\max}$ and a *separator* for any given cluster (the procedure that subdivides a cluster into several subclusters, if possible). The tree of clusters is described by a list of clusters $\mathscr{T}$ and an integer array $\mathscr{P}$ containing a permutation of $\{1, 2, \ldots, n\}$. Each cluster is identified by its index in the list $\mathscr{T}$. Every item in $\mathscr{T}$ has the following components:

- Level where the cluster belongs.
- Index of the parent cluster.
- Number of kid-clusters.
- Index of a cluster that is followed contiguously by the kid-clusters.
- Number of nodes in the cluster.
- Reference to the position in the permutation array $\mathscr{P}$ which is followed contiguously by indices of the nodes forming the cluster.

Mosaic-Tree Construction. Let $N(\mathscr{T})$ denote the number of items in $\mathscr{T}$ and l the maximal level in the subtree already constructed.

Step 1. Set $N_{\text{beg}} = 0$, $N_{\text{end}} = 1$, and form Item no. 1 in $\mathscr{T}$ (the number of kid-clusters is 0).

Step 2. For every i from $N_{\text{beg}} + 1$ to N_{end}, check if the ith cluster is a leaf (has no kid-clusters). If so, try to split it into subclusters using the given separator. Every subcluster, if any, is successively added to the current end of $\mathscr{T}$. Also, the contiguous indices of $\mathscr{P}$ corresponding to the ith cluster are permuted to make the indices associated with every subcluster run contiguously. Correct the kid-information in Item no. i of $\mathscr{T}$.

Step 3. Increase l by 1 and quit if it is equal to $L_{\max}$. If not, set $N_{\text{beg}} = N_{\text{end}}$, $N_{\text{end}} = N(\mathscr{T})$ and go to Step 2.

The complexity of this algorithm depends on the separator used. If the separation time is linear in the number of nodes of a cluster under separation and the number of subclusters on output is upper bounded uniformly in n, then the working time is $O(L_{\max} n)$.

At present, we tested two methods of separation. The first is strictly motivated by the asymptotical-smoothness property. The second is somewhat heuristic. As we found, both lead to about equal results in practice.

1. Find a (minimal) parallelepiped containing all the nodes of a cluster, subdivide it into 2^{μ} parallelepipeds, and make up subclusters of the nodes fallen into each of them.

2. Let a cluster consist of the nodes $x_1, \ldots, x_k$ with a center c. Find a hyperplane passing through c so that the sum of squared projections of $x_i - c$ onto this hyperplane is maximal. [1] If h is a unit vector normal to the hyperplane in question, then we need to minimize

$$\Phi \equiv \sum_{i=1}^{k} |(x_i - c, h)|^2 = h^T M h, \quad M = \sum_{i=1}^{k} (x_i - c)(x_i - c)^T.$$

It is easy to see that h is the eigenvector of M for its minimal eigenvalue. The hyperplane subdivides the space into two subspaces which accumulate the nodes of two possible subclusters.

When we proceed to preparing the list of blocks, we confine ourselves to considering only those blocks that are associated with pairs of clusters in the tree of clusters (or two of them, in case there are two meshes). We permute the original rows and columns in line with $\mathscr{P}$; then the tree-of-clusters blocks contain contiguous rows and columns.

It is possible to produce many different lists of blocks based on the same tree of clusters. Our final goal is a tree-of-cluster mosaic partitioning minimizing approximate mosaic rank. However, it might be a difficult discrete optimization problem. We use the following heuristic approach: low-rank blocks should be of

[1] This idea of separation was suggested by S. Rjasanow.

maximal possible size. We also assume that, for every pair of clusters, there is an easily computable tag showing if the block is allowed to be compressed or not; for brevity, call the former *allowed* blocks.

Mosaic-List Construction. Let $\mathcal{M}$ be the target list of blocks and $\mathcal{M}_1, \mathcal{M}_2$ auxiliary lists.

Step 1. Put in $\mathcal{M}_1$ the root-cluster block (original matrix).

Step 2. Take up successively the blocks from $\mathcal{M}_1$. If the block is allowed, relegate it immediately to $\mathcal{M}$. If not, consider the clusters a and b defining this block. Add to $\mathcal{M}_2$ all the blocks associated with the subclusters of a and b. If there are no subclusters, move this block to $\mathcal{M}$.

Step 3. Quit if $\mathcal{M}_2$ is empty. Otherwise, substitute $\mathcal{M}_1$ with $\mathcal{M}_2$, empty $\mathcal{M}_2$, and go to Step 2.

This algorithm looks, and is, rather general and may work with various rules for obtaining admissibility tags from the attributes of the couple of clusters tied with a block in question. For our applications, this tag depends on the radii and distance between the centers of clusters. On the base of constructions in [7], it can be shown that the algorithm proposed could yield a mosaic partitioning providing the mosaic-rank estimates of Theorems 1 and 2. In practice, the algorithms of this section consume rather negligible part of the whole time for the mosaic-skeleton method.

4. Incomplete Cross Approximation

On the second stage of the mosaic-skeleton method, we browse in the list of blocks and try to *compress* (approximate by skeletons) every allowed block. It can be done by the Lanczos bidiagonalization method. Although we eventually compute all the entries and the compression time does not behave any close to linear in n, the multiplication time is about linear in n and it might be still useful for some practical problems [10]. Here we propose an entirely different approach.

Let A denote an allowed block of size $m \times n$. If $\operatorname{rank} A = r$ then we can obtain skeletons using any r rows and columns for which the intersection block is nonsingular. The problem is that A is of rank r only up to some small perturbation. We rely on the following (nontrivial) result.

Theorem 3. [8, 9] *Let A and F are $m \times n$, $\operatorname{rank}(A + F) \le r$, and $||F||_2 \le \varepsilon$. Then there exists a cross in A with columns C and rows R, for which, for some $r \times r$ matrix G,*

$$||A - CGR||_2 \le \varepsilon \left(1 + \left(\sqrt{t(r,m)} + \sqrt{t(r,n)}\right)^2\right)$$

with

$$t(r,n) \equiv \max_{U} \min_{P \in \mathscr{M}(U)} \sigma_{\min}^{-1}(P),$$

where U means any r columns of a unitary matrix of order n, $\mathscr{M}(U)$ is the set of all $r \times r$ submatrices in U, and $\sigma_{\min}$ designates the minimal singular value.

It was proved also [9] that

$$t(r,n) \le t_V(r,n) \equiv \sqrt{(r(n-r)+1},$$

though we believe that $t(r,n) \le \sqrt{n}$ (not proved so far). The idea behind the proof of Theorem 3 was the splitting of the singular value decomposition of $A = U_1\Sigma_1 V_1 + U_2\Sigma_2 V_2$, where the first term corresponds to r senior singular triplets, and, then, choosing those submatrices in U_1 and V_1 that have the reciprocal to the minimal singular value majorized by $t(r,m)$ and $t(r,n)$, respectively. The rows chosen in U_1 and columns in V_1 determine the cross at issue.

The choice of the above submatrices in U_1 and V_1 can be performed constructively: it is sufficient to find the submatrices of maximal determinant in modulus (we call this quantity a *volume*) [9].

Unfortunately, the singular value decomposition of A is too heavy a tool to be practical for our purposes. Instead, we would fall back to the concept of maximal volumes.

To give more motivation, recall the role of maximal volumes in the interpolation theory. Assume that Ω is a compact domain in the m-dimensional space. Assume that $f(x)$ is to be interpolated by $\phi(x) = \sum_{l=1}^{k} \alpha_i \phi_l(x)$ in the nodes $x_1, \ldots, x_k \in \Omega$. Then, as is readily verified,

$$\phi(x) = \sum_{l=1}^{k} f(x_l)\, \frac{M_l}{M},$$

where $M = \det\{\phi_i(x_j)\}_{ij=1}^{k}$ and M_l is obtained from M by replacing the lth column by $[\phi_1(x_l), \ldots, \phi_k(x_l)]^T$. If we are allowed to choose the nodes, which is a reasonable choice? A classical result (see [6]) is that a good idea is to maximize $|\det M|$ over $x_1, \ldots, x_k \in \Omega$.

Theorem 4. *Let M maximize $|\det M|$ and $||f||_C \equiv \max_{x\in\Omega} |f(x)|$. Then $||f-\phi||_C \le (1+k)\, E_{\text{best}}$, where $E_{\text{best}} = \inf_{\beta_1,\ldots,\beta_k} ||f - \sum_{l=1}^{k} \beta_l \phi_l||_C$.*

Luckily, this theorem is as profound as elementary allowing for a one-line

Proof: $|f-\phi| \le |f-\phi_{\text{best}}| + |\phi - \phi_{\text{best}}| \le E_{\text{best}} + \sum_{l=1}^{k} |f(x_l) - \phi_{\text{best}}(x_l)|\, \left|\frac{M_l}{M}\right|$,

where ϕ_{best} corresponds to the optimal choice of β_l on which the best uniform bound is attained. A matrix analogue of Theorem 4 is the following.

Theorem 5. *Let A be $m \times n$ and C consist of the first k columns of A. Assume that* rank $C = k$ *and let B be the maximal-volume submatrix in C. Denote by R the rows of A defined by B. Then*

$$||A - CB^{-1}R||_2 \leq (1 + t_V(k,m)) \inf_W ||A - CW||_2. \tag{10}$$

At the same time, if $\mu(A)$ is the maximal entry of A in modulus, then

$$\mu(A - CB^{-1}R) \leq (1 + k) \inf_W \mu(A - CW). \tag{11}$$

Proof: Let $A = [C, A_2]$ and, similarly, $W = [W_1, W_2]$. Then

$$A - CB^{-1}R = (A_2 - CW_2) + CB^{-1}(BW_2 - R).$$

Since B is of maximal volume in C, all the entries of CB^{-1} are not greater than 1 in modulus. Indeed, the postmultiplication of C by any nonsingular matrix does not change the ratio of volumes for any two submatrices. With no loss of generality, let

$$H \equiv CB^{-1} = \begin{bmatrix} & I & \\ h_{k+11} & \cdots & h_{k+1k} \\ \cdots & \cdots & \cdots \end{bmatrix}.$$

If $|h_{k+lj}| > 1$, then H would have a submatrix of volume greater than 1. This submatrix would be I with row j substituted with row $k + l$. Now, (10) are (11) are evident. □

The above proof incorporates an algorithm which we use to find maximal-volume submatrices. On input, we have C and some $\gamma \geq 1$. On output, we obtain a submatrix B whose volume is greater than or equal to the maximal volume divided by γ. With such a B, we modify the estimate (11) to the form

$$\mu(A - CB^{-1}R) \leq (1 + \gamma k) \inf_W \mu(A - CW). \tag{12}$$

Thus, instead of maximal-volume submatrices, we may look for those of sufficiently big volume.

Big-Volume Search.

Step 1. Reduce C to H by column transformations using a complete pivoting.

Step 2. Find the maximal in modulus entry below the kth row of H. Let it have indices $k + l, j$. Quit if it does not exceed γ.

Step 3. Interchange row j and $k + l$ and eliminate nondiagonal zeroes in the upper part of H using column transformations. Go to Step 2.

We are ready to formulate the incomplete cross approximation algorithm. Presently it is a succession of the prescribed number of cross-inflating cycles.

Cross-Inflating Cycle. Given a cross with row indices I and column indices J , find the maximal-volume submatrices in the columns and the rows. Let the former has

column indices $\hat{J}$ and the latter has row indices $\hat{I}$. Then update the cross changing I onto $I \bigcup \hat{I}$ and J onto $J \bigcup \hat{J}$.

Thus, we form the cross adaptively adding to it new rows and columns. Using a sufficiently large cross, we provide an accurate skeleton approximation for the whole block. At the same time, we hope that we finish with a cross small enough.

5. Time Versus Size

An important issue is how we get an initial cross. Sometimes, as in our experiments with the equation (1), it could be almost arbitrary (even of size 1×1 or 2×2). This is not entirely clear yet from the point of theory. However, below we present a practical justification. Consider more information related to Table 1. On Fig. 1 we can see that the approximation time grows almost linearly in n. This is due to the incomplete cross approximation approach. Note also that we symmetrized the mosaic-skeleton approximations and used the conjugate gradients with the circulant preconditioner [4, 5, 25, 26]. We had only 3 iteration to reduce the residual to a factor of 10^{-4}.

The time for iterations was about 30 of the total time, with a tendency to become a smaller part as n increases.

The relative approximation accuracy was set to $\varepsilon = 10^{-4}$ in all experiments. The admissibility tag for a couple of clusters is up whenever $\min r_1, r_2 \leq \alpha\rho$ and $r_1 + r_2 \leq \rho$, where r_1 and r_2 are the radii of clusters, ρ is the distance between their centers, and α is some constant (it was set to 0.3). We opted for the second method

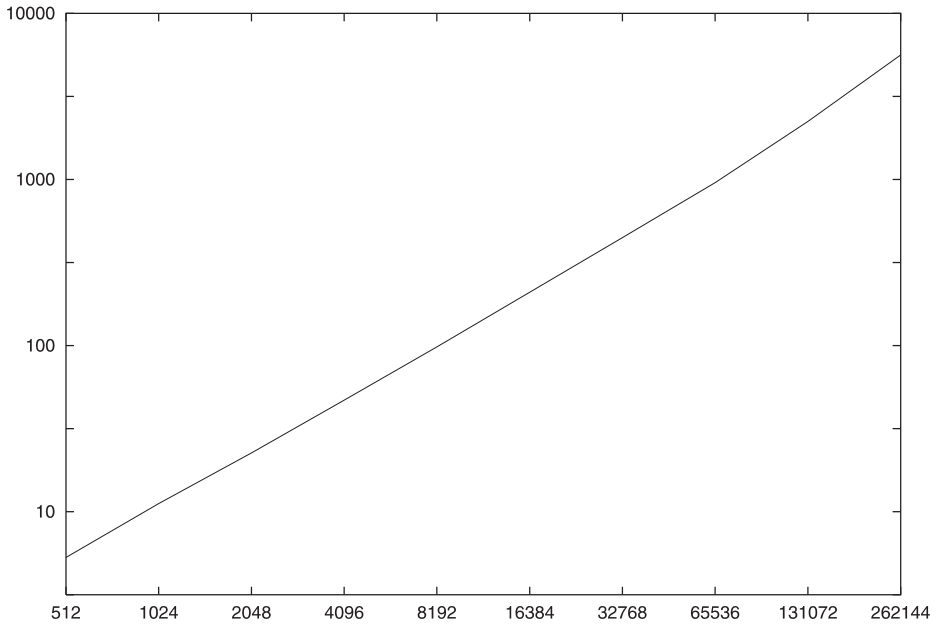

Figure 1. log (TIME) versus log (SIZE)

to separate a cluster. With this, the maximal number of levels in the Mosaic-Tree Construction was chosen empirically as $\log_2 \frac{n}{4}$ (it minimized time yet the minimal mosaic rank might correspond to a larger number of levels). In the Big-Volume Search, we used $\gamma = 1.1$ (however, this parameter could be increased with next to no harm, at least for our examples). In all experiments, the size of initial cross was 2 and the allowed number of the cross-inflating steps was 5. The numerical results confirm that it is a sound choice. However, the proof of its optimality is a subject of further research.

The maximal size we tested was $n = 1\,048\,576$. The computed mosaic rank was 159.72 and the corresponding compression factor was 0.03. We used about 9 Gb of the disc space and 27 283 seconds on the Silicon Graphic workstation in the University of Saarland.

6. Discussion

In this paper, we concentrated on numerical verification of algebraic algorithms for the mosaic-skeleton method. That is why we kept the relative approximation accuracy ε independent of the matrix size. In the context of solving integral equations, this should not be the case (still, for the examples considered, we have made a rather conservative choice for all sizes).

All the same, it is interesting to see how the mosaic ranks would behave depending on ε. As above, take up the logarithmic-kernel equation on an ellipse with half-axes $a = 1$ and $b = 0.5$. In Table 3, there are mosaic ranks and compression factors for $n = 1024$.

Remark that $\varepsilon \leq 10^{-2}$ leads to a poor approximation and should not be used. However, even those ε may be of choice if the fast approximate multiplication algorithm serves as a preconditioner for matrices related to some other operator.

One may also ask what happens with the mosaic ranks when a/b gets larger. In Table 4, the answer is given for $n = 1024$ and $\varepsilon = 10^{-4}$. Here, we vary b with $a = 1$ being fixed. Curiously, the mosaic ranks increase for a while and then start decreasing as $a/b \to \infty$.

The mosaic-skeleton method can be applied also to other types of equations and to 3D problems (see [10]). Application of algebraic algorithms proposed in this paper to such cases is a topic of forthcoming work.

Table 3. Mosaic ranks versus relative approximation accuracy

Relative approx. accuracy	10^0	10^{-1}	10^{-2}	10^{-3}	10^{-5}	10^{-6}	10^{-7}
Mosaic rank	40.51	42.67	52.75	63.93	82.72	91.34	101.05
Compression factor	7.91%	8.33%	10.30%	12.49%	16.16%	17.84%	19.74%

Table 4. Mosaic ranks versus a/b

a/b	1	4	16	64	256	10000
Mosaic rank	67.12	78.34	97.63	84.75	71.78	67.73
Compression factor	13.11%	15.30%	19.07%	16.55%	14.02%	13.23%

Acknowledgements

The author appreciates useful remarks by the referees. Special thanks for the computing facilities go to Prof. Dr. S. Rjasanow.

References

[1] Beylkin, G., Coifman, R., Rokhlin, V.: Fast wavelet transform and numerical algorithms. I. Comm. Pure Appl. Math. *44*, 141–183 (1991).
[2] Brandt, A.: Multilevel computations of integral transforms and particle interactions with oscillatory kernels. Comput. Phys. Comm. *65*, 24–38 (1991).
[3] Brandt, A., Lubrecht, A. A.: Multilevel matrix multiplication and fast solution of integral equations. J. Comput. Phys. *90*, 348–370 (1990).
[4] Chan, R., Sun, H., Ng, W.: Circulant preconditioners for ill-conditioned boundary integral equations from potential equations (to appear).
[5] Chan, T.: An optimal circulant preconditioner for Toeplitz systems. SIAM J. Sci. Stat. Comput. *9*, 766–771 (1988).
[6] Gaier, D.: Vorlesungen über Approximation im Komplexen. Basel-Boston-Berlin: Birkhäuser 1980.
[7] Goreinov, S. A.: Mosaic-skeleton approximations of matrices generated by asymptotically smooth and oscillatory kernels. In: Matrix methods and computations, pp. 41–76. Moscow: INM RAS, 1999. (in Russian)
[8] Goreinov, S. A., Tyrtyshnikov, E. E., Zamarashkin, N. L.: Pseudo-Skeleton Approximations of Matrices. Rep. Russian Acad. Sci. *343*, 151–152 (1995). (in Russian)
[9] Goreinov, S. A., Tyrtyshnikov, E. E., Zamarashkin, N. L.: A Theory of Pseudo-Skeleton Approximations. Linear Algebra Appl. *261*, 1–21 (1997).
[10] Goreinov, S. A., Tyrtyshnikov, E. E., Yeremin, A. Yu.: Matrix-free iterative solution strategies for large dense linear systems. Numer. Linear Algebra Appl. *4*, 273–294 (1997).
[11] Greengard, L., Rokhlin, V.: A fast algorithm for particle simulations. J. Comput. Phys. *73*, 325–348 (1987).
[12] Hackbusch, W.: A Sparse Matrix Arithmetic based on H-Matrices. Part I: Introduction to H-Matrices. Computing *62*, 89–108 (1999).
[13] Hackbusch, W., Novak, Z. P.: On the fast matrix multiplication in the boundary element method by panel clustering. Numer. Math. *54*, 463–491 (1989).
[14] Harten, A.: Multiresolution representation and numerical algorithms: a brief review. ICASE Report 94–59, October 1994.
[15] Martynov, M. S.: Usage of fast multiplication methods for solving integral equation of potential theory. In: Matrix methods and computations, pp. 77–130. Moscow: INM RAS, 1999. (in Russian)
[16] Myagchilov, S. V., Tyrtyshnikov, E. E.: A fast matrix-vector multiplier in discrete vortex method. Russian J. Numer. Anal. Math. Modell. *7*, 325–342 (1992).
[17] Nabors, K., Korsmeyer, F. T., Leighton, F. T., White, J.: Preconditioned, adaptive, multipole-accelerated iterative methods for three-dimensional potential integral equations of the first kind. Dept. of Electrical Eng. and Computer Science, Massachusetts Institute of Technology, 1994.
[18] Nechepurenko, Yu. M.: Fast numerically stable algorithms for a wide class of discrete linear transforms. Preprint 92, OVM RAN, 1985. (in Russian.)
[19] Nikolsky, I. Yu.: Interpolation method for fast approximate multiplication for a matrix generated by a function on a contour. In: Matrix methods and computations, pp. 131–145. Moscow: INM RAS, 1999. (in Russian)
[20] Rokhlin, V.: Rapid solution of integral equations of classical potential theory. J. Comput. Phys. *60*, 187–207 (1985).
[21] Rokhlin, V.: Rapid solution of integral equations of scattering theory in two dimensions. J. Comput. Phys. *86*, 414–439 (1990).
[22] Tyrtyshnikov, E. E.: Mosaic ranks and skeletons. In: Lecture Notes in Computer Science 1196: Numerical analysis and its applications. Proceedings of WNAA-96 (Vulkov, L., et al., ed.), pp. 505–516. Berlin Heidelberg New York Tokyo: Springer, 1996.
[23] Tyrtyshnikov, E. E.: Mosaic–skeleton approximations. Calcolo *33*, 47–57 (1996).
[24] Tyrtyshnikov, E.: Methods for fast multiplication and solution of equations. In: Matrix Methods and Computations, pp. 4–40. Moscow: INM RAS, 1999. (in Russian)

[25] Tyrtyshnikov, E. E.: Optimal and superoptimal circulant preconditioners. SIAM J. Matrix Anal. Appl. *12*, 459–473 (1992).
[26] Tyrtyshnikov, E.: A brief introduction to numerical analysis. Boston: Birkhäuser 1997.
[27] Voevodin, V. V.: On an order-reduction method for matrices arising in the solution of integral equations. In: Numerical Analysis on FORTRAN, pp. 21–26. Moscow: Moscow State University Press, 1979. (in Russian)

E. Tyrtyshnikov
Institute of Numerical Mathematics
Russian Academy of Sciences
Gubkina 8, Moscow 117333
Russia
E-mail: tee@inm.ras.ru