FAST STRUCTURED DIRECT SPECTRAL METHODS FOR DIFFERENTIAL EQUATIONS WITH VARIABLE COEFFICIENTS, I. THE ONE-DIMENSIONAL CASE

JIE SHEN*, YINGWEI WANG*, AND JIANLIN XIA*

Abstract. We study the rank structures of the matrices in Fourier- and Chebyshev-spectral methods for differential equations with variable coefficients in one dimension. We show analytically that these matrices have a so-called low-rank property, not only for constant or smooth variable coefficients, but also for coefficients with steep gradients and/or high variations (large ratios in their maximum-minimum function values). We develop a matrix-free direct spectral solver, which uses only a small number of matrix-vector products to construct a structured approximation to the original discretized matrix A, without the need to explicitly form A. This is followed by fast structured matrix factorizations and solutions. The overall direct spectral solver has $O(N \log^2 N)$ complexity and $O(N)$ memory requirement. Numerical tests for several important but notoriously difficult problems show the superior efficiency and accuracy of our direct spectral solver, especially when iterative methods have severe difficulties in the convergence.

Key words. spectral method, structured matrix, low-rank property, matrix-free direct solver, linear complexity

AMS subject classifications. 65N35, 65F05, 65F30

1. Introduction.

Spectral methods have been extensively used for solving differential equations due to their high order of accuracy, see [13, 3, 25, 26] and the references therein. However, for problems with general variable coefficients, spectral methods lead to full matrices. An effective approach is to use a low-order method (finite differences or finite elements [22, 4, 11, 16]), or integration operator [9, 34], as a preconditioner and to take advantage of the fact that the matrix-vector product from a Fourier- or Chebyshev-spectral discretization can be performed in quasi-optimal complexity. While this approach yields quasi-optimal complexity since lower-order methods on the same grid usually provide optimal preconditioners, it can still be expensive (say, for multiple right-hand sides) and is generally not so robust as compared with direct solvers using finite difference or finite element methods.

The aim of this paper is to develop fast *direct* spectral Galerkin methods for solving differential equations with variable coefficients in one dimension. For the sake of simplicity, we restrict our attention to the following equation:

$$\alpha(x)u - (\beta(x)u_x)_x = f(x), \quad x \in (a,b), \tag{1.1}$$

where $\alpha(x), \beta(x), f(x)$ are given functions and $\alpha(x) \geq 0$, $\beta(x) > 0$. It will be clear from the proofs that the results presented in this paper do not depend on the ellipticity, and are actually applicable to a wide class of differential equations with variable coefficients.

It is well known that when $\alpha(x), \beta(x)$ are constants or polynomials, spectral-Galerkin methods with proper basis functions may lead to well-conditioned sparse linear systems. However, when $\alpha(x), \beta(x)$ are general variable coefficients, a spectral method for (1.1) leads a linear system $A\mathbf{u} = \mathbf{f}$ with A dense and often ill conditioned. When the order of the matrix N is large, the $O(N^3)$ operations and $O(N^2)$ storage

*Department of Mathematics, Purdue University, West Lafayette, IN 47907 (shen7@math.purdue.edu, wywshtj@gmail.com, xiaj@math.purdue.edu). The work of Jie Shen and Yingwei Wang is partially supported by NSF grants DMS-1217066 and DMS-1419053. The research of Jianlin Xia was supported in part by NSF CAREER Award DMS-1255416 and NSF grant DMS-1115572.

required by common dense numerical linear algebra become overwhelmingly expensive. Moreover, the dense matrix A is usually not explicitly available and it is costly to form it. Thus, in traditional spectral methods, iterative methods are often used to solve the resulting linear system, due to the fact that A can be quickly multiplied with vectors. However, when the coefficients have *steep gradients* and/or *large variations or are degenerate*, iterative methods often converge very slowly or do not converge at all.

On the other hand, many scientific and engineering problems often lead to differential equations with variable coefficients with steep gradients and/or large variations, or even degenerate coefficients (i.e., $\beta(x) = 0$ at some points). Examples include interface problems such as Cahn-Hilliard equations or phase-field systems with a concentration dependent mobility, highly anisotropic problems in fluid dynamics and wave scatterings, problems with singular perturbations, problems with interior layers, etc. Hence, it is important to develop robust and accurate solution methods for these situations.

In this work, we seek to quickly solve the linear system $A\mathbf{u} = \mathbf{f}$ resulting from a spectral discretization by a type of so-called structured direct methods based on matrix-vector products. The basic ideas are initially mentioned in [38]. We first study the detailed forms of the intermediate matrices used to build A via a step-by-step decomposition of the spectral methods. Our analysis shows that, in many cases, the dense matrix A enjoys a *low-rank property*, i.e., their off-diagonal blocks have small and nearly bounded (numerical) ranks. For the constant coefficient case, the low-rank property is essentially connected to the properties of the Green functions. We justify such a property for problems with not only constant coefficients, but also variable smooth coefficients and variable coefficients having steep gradients and/or large variations and even degenerate coefficients. This is done for both Fourier- and Chebyshev-Galerkin methods. Algebraic derivations are accompanied by illustrations in terms of some functions which are notoriously difficult to handle with iterative methods.

For certain cases (e.g., constant or smooth coefficients), such a low-rank property can be understood based on the fast decay of the matrix entries away from the diagonal. However, even if the entries decay very slowly or do not decay, such as with steep gradients in the coefficients, we show that the low-rank property still holds. For these cases, a large number of modes are needed to accurately represent the coefficient. A straightforward truncation of the small matrix entries results in an approximate matrix with a large bandwidth and is not very effective. Similarly, the low-rank property is also insensitive to the large ratios in the function values.

Here, A can then be approximated by rank structured matrices, which have already been shown to be very effective in the direct solutions of large linear systems arising from finite difference, finite element, or boundary element discretizations of differential equations [1, 2, 6, 19, 40, 23] and integral equations [20, 17]. One frequently used rank structure is the *hierarchically semiseparable* (HSS) representation [40, 41, 42, 37]. HSS structures provide an effective and reliable way to quickly handle dense matrices with the low-rank property. The factorization and solution with an HSS matrix usually cost only about $O(N)$ flops.

To avoid forming A explicitly, we use an adaptive *matrix-free* scheme [36] to construct an HSS approximation to A, which is built upon randomized techniques [15, 18, 42]. For a pre-specified accuracy, such a construction takes about $O(r \log N)$ matrix-vector products, where r is the maximum off-diagonal (numerical) rank of A

and is small. Since the product of matrix A and a vector in both the Fourier- and the Chebyshev-Galerkin methods can be computed in $O(N \log N)$ flops via FFTs, our total HSS construction cost is only $O(rN \log^2 N)$. (Later, we may use *nearly* $O(N)$ *complexity* to mean $O(N \log^k N)$ complexity for a small integer k). The HSS form can be factorized in about $O(r^2 N)$ flops, and then the linear system solution cost is only $O(rN)$ for each right-hand side. The memory requirement is also $O(rN)$. All these counts are roughly linear in N.

Our methods are thus very attractive for spectral approximations of problems with variable coefficients, especially when the coefficients have steep gradients and/or large variations or are degenerate, and when only matrix-vector products are available.

The remainder of this paper is organized as follows. In Section 2, we construct the detailed matrix forms in Fourier- and Chebyshev-Galerkin methods for elliptic equations with periodic and non-periodic boundary conditions, respectively. We study in Section 3 the low-rank property of the dense matrices obtained in the previous section. In Section 4, we present a fast and stable matrix-free direct solver based on HSS construction, factorization, and solution schemes. We illustrate in Section 5 the performance of the proposed direct spectral solutions via several examples, which demonstrates high accuracy and efficiency. Section 6 contains the conclusions, generalizations and possible directions for future research.

2. Matrix forms for Fourier- and Chebyshev-Galerkin methods. In order to study the hidden low-rank property of the discretized matrices in the Fourier- and Chebyshev-Galerkin methods, we first illustrate the explicit matrix forms in all the intermediate steps, supplementing those given in [25, 26]. This differs from classical spectral methods which primarily evaluate matrix-vector products and ignore the actual matrix structures. Note that the study of the explicit matrices are only to facilitate our analysis in the next section, and they are not actually formed in the implementation due to the matrix-free direct solver in Section 4.

Throughout the paper, we employ the following notation:

- $\{x_j\}$ denotes some collocation points in an interval in one dimension;
- $\boldsymbol{\alpha}$ denotes a vector formed by $\{\alpha(x_j)\}$ for a function $\alpha(x)$ evaluated at the points x_j;
- $\mathcal{D}_{\boldsymbol{\alpha}}$ or $\mathrm{diag}(\boldsymbol{\alpha})$ denotes a diagonal matrix with the entries of the vector $\boldsymbol{\alpha}$ as the diagonal entries;
- $A|_j$ (or $A|_{j,:}$) and $A|_{:,j}$ denote the jth row and column of A, respectively, and can be similarly understood when j is replaced by an index set;
- $A|_{j,k}$ denotes the (j,k) entry of A, and can be similarly understood when j and k are replaced by index sets;
- $\mathrm{rank}(T)$ denotes the rank of a matrix T.

Justifications of the methods in the reviews of this section can be found in [25, 26].

2.1. Fourier-Galerkin method. We consider the problem (1.1) with periodic boundary conditions:

$$\alpha(x)u - (\beta(x)u_x)_x = f(x), \quad x \in (0, 2\pi), \tag{2.1}$$

$$u(0) = u(2\pi), \quad u'(0) = u'(2\pi). \tag{2.2}$$

Let

$$X_N = \left\{ v(x) = \sum_{k=-N/2}^{N/2} \tilde{v}_k \mathrm{e}^{ikx} : \tilde{v}_{-N/2} = \tilde{v}_{N/2} \right\}, \tag{2.3}$$

where, for convenience, N is assumed to be an even number, and odd N can be handled similarly. Let the Fourier collocation points $\{x_j\}_{j=0}^{N-1}$ be

$$x_j = j\frac{2\pi}{N}, \quad j = 0, 1, \ldots, N-1. \tag{2.4}$$

We define the interpolation operator $I_N : \mathcal{C}[0, 2\pi) \longrightarrow X_N$ such that

$$I_N u \in X_N : \ (I_N u)(x_j) = u(x_j), \quad j = 0, 1, \ldots, N-1.$$

It is easy to verify that

$$(I_N u)(x) = \sum_{k=-N/2}^{N/2} \tilde{u}_k \mathrm{e}^{ikx}, \tag{2.5}$$

where $\tilde{u}_k,\ k = -\frac{N}{2}, \ldots, \frac{N}{2}$ are Fourier expansion coefficients that can be obtained from the forward Discrete Fourier Transform (FDFT). Similarly, the backward Discrete Fourier Transform (BDFT) can be used to obtain $u(x_j),\ j = 0, 1, \ldots, N-1$ from $\tilde{u}_k$ [26]. We recall that these transforms can be carried out in $O(N \log N)$ operations [8], in general.

The Fourier-Galerkin method (with numerical integration) for (1.1) is to find $u_N \in X_N$ such that

$$\langle \alpha(x) u_N, \mathrm{e}^{-ikx} \rangle_N - \langle [\beta(x) u_N']', \mathrm{e}^{-ikx} \rangle_N = \langle f, \mathrm{e}^{-ikx} \rangle_N, \quad k = -\frac{N}{2}, \ldots, \frac{N}{2}, \tag{2.6}$$

where the discrete inner product is defined by

$$\langle u, v \rangle_N = \frac{1}{N} \sum_{j=0}^{N-1} u(x_j) \bar{v}(x_j). \tag{2.7}$$

It is easy to show that $\{\mathrm{e}^{-ikx}\}_{k=-N/2}^{N/2}$ is a set of orthonormal basis in the space X_N with respect to the inner product (2.7).

Suppose $\mathbf{u} = (u(x_j))_{j=0}^{N-1}$ is the vector of function values and $\tilde{\mathbf{u}} = (\tilde{u}_k)_{k=-N/2}^{N/2}$ is the vector of the Fourier expansion coefficients. Then the transformations can be done via

$$\tilde{\mathbf{u}} = F\mathbf{u}, \quad \mathbf{u} = F^* \tilde{\mathbf{u}}.$$

where the matrices F and F^* represent FDFT and BDFT respectively. Besides, the coefficients of the expansion of the first derivative $u'(x)$ are given by

$$\tilde{\mathbf{u}}^{(1)} = i\mathbf{k} .* \tilde{\mathbf{u}},$$

where the right-hand side involves a componentwise multiplication, and the vector $\mathbf{k}$ is given by

$$\mathbf{k} = (0, 1, \ldots, \frac{N}{2}, -\frac{N}{2}, -\frac{N}{2} + 1, \ldots, -1).$$

Hence, the general procedure for solving the problem (2.6) in the matrix forms is as follows.

1. Computing the right-hand side.
   Evaluate $f(x)$ at the points $\{x_j\}_{j=0}^{N-1}$ in (2.4) to get the vector $\mathbf{f} = (f(x_j))_{j=0}^{N-1}$.
   Use FFT to obtain the Fourier expansion coefficient vector $\tilde{\mathbf{f}} = (\tilde{f}_k)_{k=-N/2}^{N/2}$.
2. Solving in frequency space.
   Let $\boldsymbol{\alpha} = (\alpha(x_j))_{j=0}^{N-1}$, $\boldsymbol{\beta} = (\beta(x_j))_{j=0}^{N-1}$. The variational form (2.6) leads to the linear system

$$A\tilde{\mathbf{u}} = \tilde{\mathbf{f}}, \quad A = F\mathcal{D}_{\boldsymbol{\alpha}}F^* + \mathcal{D}_{\mathbf{k}}F\mathcal{D}_{\boldsymbol{\beta}}F^*\mathcal{D}_{\mathbf{k}}. \tag{2.8}$$

   The matrix A is positive definite since $\alpha(x), \beta(x) > 0$. Solve this for the vector of unknowns $\tilde{\mathbf{u}} = (\tilde{u}_k)_{k=-N/2}^{N/2}$ in (2.5).
3. Transforming back to physical space.
   Obtain the vector of function values $\mathbf{u} = (u(x_j))_{j=0}^{N-1} = F^*\tilde{\mathbf{u}}$.

The main expense of this procedure is to solve the linear system (2.8) to get $\tilde{\mathbf{u}}$.

REMARK 1. We can also derive A by representing the multiplication of the Fourier series for the coefficients and $u(x)$ as an operator on the coefficients. The two matrices in the sum in (2.8) are then approximations of the Fourier multiplication operators.

2.2. Chebyshev-Galerkin method. Next, consider the problem (1.1) with Dirichlet boundary conditions:

$$\alpha(x)u - (\beta(x)u_x)_x = f(x), \quad x \in (-1, 1), \tag{2.9}$$

$$u(-1) = u(1) = 0. \tag{2.10}$$

Denote $\omega(x) = (1 - x^2)^{-1/2}$ and $\tilde{X}_N = \{v \in P_N : v(-1) = v(1) = 0\}$, where P_N is the set of all polynomials of degrees less than or equal to N. Then, the Chebyshev-Galerkin method (with numerical integration) is to find $u_N \in X_N$ such that

$$\langle \alpha u_N, v_N\rangle_{N,\omega} - \langle (\beta u_N')', v_N\rangle_{N,\omega} = \langle f, v_N\rangle_{N,\omega}, \quad \forall v_N \in \tilde{X}_N, \tag{2.11}$$

where $\langle\cdot\rangle_{N,\omega}$ is the discrete inner product relative to the Chebyshev-Gauss-Lobatto quadrature.

Denote

$$\phi_k(x) := T_k(x) - T_{k+2}(x),$$

where T_k is the Chebyshev polynomial of degree k. It is easy to know that the function $\phi_k(x)$ defined here satisfies the boundary condition (2.10). Hence, the space of the basis functions can be chosen as

$$X_N = \operatorname{span}\{\phi_k(x)\}_{k=0}^{N-2}. \tag{2.12}$$

Suppose the Chebyshev expansion of $u(x)$ with finitely many terms is

$$u(x) = \sum_{n=0}^{N} \tilde{u}_n T_n(x) = \sum_{k=0}^{N-2} \bar{u}_k \phi_k(x).$$

Let $\tilde{\mathbf{u}} = (\tilde{u}_n)_{n=0}^{N}$ and $\bar{\mathbf{u}} = (\bar{u}_k)_{k=0}^{N-2}$ be the column vectors of expansion coefficients. In order to implement (2.11), we need to perform the following tasks.

1. Transforming between the basis functions $\{\phi_k\}_{k=0}^{N-2}$ and the Chebyshev polynomials $\{T_n\}_{n=0}^{N}$. The transforms between $\tilde{\mathbf{u}}$ and $\bar{\mathbf{u}}$ can be performed by

$$\tilde{\mathbf{u}} = A_1 \bar{\mathbf{u}} \quad \text{with}$$

$$A_1 = \begin{pmatrix} 1 & & \\ 0 & \ddots & \\ -1 & \ddots & 1 \\ & \ddots & 0 \\ & & -1 \end{pmatrix}_{(N+1)\times(N-1)}.$$

2. Differentiation in spectral space.
Recall the differentiation relation of the Chebyshev polynomials

$$T_n'(x) = 2n \sum_{\substack{k=0 \\ k+n:\ \text{odd}}}^{n-1} \frac{1}{c_k} T_k(x), \tag{2.13}$$

where $c_0 = 2$ and $c_k = 1$ for $k \geq 1$. Suppose $u'(x) = \sum_{n=0}^{N} \tilde{u}_n' T_n(x)$, and let $\tilde{\mathbf{u}}' = (\tilde{u}_n')_{n=0}^{N}$ be the vector of the expansion coefficients of the first derivative. Following (2.13), we have

$$\tilde{\mathbf{u}}' = A_2 \tilde{\mathbf{u}}, \quad \text{with}$$

$$A_2 = \begin{pmatrix} 0 & 1 & 0 & 3 & 0 & 5 & \cdots & 0 \\ & 0 & 4 & 0 & 8 & 0 & \cdots & 2(N+1) \\ & & 0 & 6 & 0 & 10 & \cdots & 0 \\ & & & 0 & 8 & 0 & \cdots & 2(N+1) \\ & & & & 0 & 10 & \ddots & 0 \\ & & & & & \ddots & \ddots & \vdots \\ & & & & & & \ddots & 2(N+1) \\ & & & & & & & 0 \end{pmatrix}_{(N+1)\times(N+1)}. \tag{2.14}$$

More specifically, A_2 has the following form:

$$A_2|_{j,j+1:N} = \begin{cases} (1, 0, 3, 0, 5, \ldots, 0), & j = 1, \\ 2(j, 0, j+2, 0, j+4, \ldots, 0), & j > 1,\ j\text{: odd}, \\ 2(j, 0, j+2, 0, \ldots, N+1), & j > 1,\ j\text{: even}. \end{cases} \tag{2.15}$$

3. Inner product with the basis functions.
For the inner product vector of the solution $\hat{\mathbf{u}} = \{\hat{u}_k\}_{k=0}^{N-2}$, where $\hat{u}_k = \langle u(x), \phi_k(x) \rangle_{N,\omega}$, we have

$$\hat{\mathbf{u}} = A_3 \tilde{\mathbf{u}}, \quad \text{with}$$

$$A_3 = \begin{pmatrix} \pi & 0 & -\pi/2 & & & \\ & \pi/2 & 0 & -\pi/2 & & \\ & & \ddots & \ddots & \ddots & \\ & & & \pi/2 & 0 & -\pi/2 \end{pmatrix}_{(N-1)\times(N+1)}.$$

4. Forward discrete Chebyshev transform (FDCT) and backward discrete Chebyshev transform (BDCT).
   It is well known that the Chebyshev collocation points $\{x_j\}_{j=0}^N$ are

$$x_j = \cos(\frac{\pi j}{N}), \quad j = 0, 1, \ldots, N. \tag{2.16}$$

   The transforms between the spectral space $\tilde{\mathbf{u}}$ and the physical space $\mathbf{u} = (u(x_j))_{j=0}^N$ can be performed by

$$\mathbf{u} = \tilde{F}\tilde{\mathbf{u}}, \quad \tilde{\mathbf{u}} = \tilde{F}^*\mathbf{u}, \tag{2.17}$$

   where $\tilde{F}$ and $\tilde{F}^*$ are the FDCT and BDCT matrices, respectively. FDCT and BDCT can be performed with $O(N \log N)$ operations via FFTs, in general.

Hence, a detailed procedure for solving the problem (2.11) in the matrix forms is:

1. Computing the right-hand side.
   Compute the Fourier collocation points $\{x_j\}_{j=0}^N$ defined in (2.16), and evaluate the right-hand function $f(x)$ at these points to get the vector $\mathbf{f} = (f(x_j))_{j=0}^N$. Then use FDCT to obtain the Chebyshev expansion coefficients vector $\tilde{\mathbf{f}} = (\tilde{f}_n)_{n=0}^N$. Finally, take the inner product with the basis functions $\{\phi_k\}_{k=0}^{N-2}$ to get $\hat{\mathbf{f}} = \{\hat{f}_k\}_{k=0}^{N-2}$:

$$\hat{\mathbf{f}} = A_3\tilde{F}^*\mathbf{f}.$$

2. Solving in frequency space.
   Let $(I_N u)(x) = \sum_{k=0}^{N-2} \bar{u}_k\phi_k(x)$, where $\bar{\mathbf{u}} = (\bar{u})_{k=0}^{N-2}$ are the unknowns. Let $\boldsymbol{\alpha} = (\alpha(x_j))_{j=0}^N$ and $\boldsymbol{\beta} = (\beta(x_j))_{j=0}^N$ be the vectors of function values. Then the variational form (2.11) leads to the following linear system:

$$A\bar{\mathbf{u}} = \hat{\mathbf{f}}, \quad A = A_3(\tilde{F}^{-1}\mathcal{D}_{\boldsymbol{\alpha}}\tilde{F} - A_2\tilde{F}^{-1}\mathcal{D}_{\boldsymbol{\beta}}\tilde{F}A_2)A_1, \tag{2.18}$$

   where A is $(N-1)\times(N-1)$. Solve this equation for the expansion coefficients $\bar{\mathbf{u}}$.
3. Transforming back to physical space.
   The function value vector $\mathbf{u} = (u(x_j))_{j=0}^N$ can be obtained by

$$\mathbf{u} = \tilde{F}A_1\bar{\mathbf{u}}.$$

In Section 4, we will show how to solve (2.18) efficiently to get $\bar{\mathbf{u}}$.

3. Low-rank property for problems with variable coefficients. We seek to quickly solve the systems (2.8) and (2.18) by taking advantage of a hidden low-rank property. That is, we show the off-diagonal blocks of A have small numerical ranks for these cases:

1. constant coefficients (included for completeness);
2. smooth variable coefficients;
3. variable coefficients with steep gradients and/or large variations.

3.1. Preliminaries. For notational convenience, we let N be the size of A (instead of $N-1$ as in the previous section). For simplicity, we focus on the following type of off-diagonal blocks, as often used in the studies of rank structures [6, 32].

DEFINITION 3.1 (Off-diagonal blocks). *Let A be an $N \times N$ matrix. Partition A into the following block 2×2 form such that the diagonal blocks are square matrices:*

$$A = \begin{array}{c} i \\ N-i \end{array} \left(\begin{array}{c|c} A_{11} & A_{12} \\ \hline A_{21} & A_{22} \end{array} \right).$$

$A_{12} \equiv A|_{1:i,i+1:N}$ and $A_{21} \equiv A|_{i+1:N,1:i}$ are called the off-diagonal blocks of A.

That is, an off-diagonal block touches but does not include the main diagonal, and consists of either the top-right or bottom-left part of A. We are interested in the numerical ranks of these blocks. The following definition is frequently used by researchers (see, e.g., [41]).

DEFINITION 3.2 (Numerical ranks and low-rank property). *The numerical rank of a matrix is the number of its singular values larger than a given tolerance τ. This can be similarly defined for a relative tolerance τ (relative to the largest singular value). A matrix has the low-rank property if all its off-diagonal blocks have ranks or numerical ranks that are much smaller than N. (Typically, the ranks are bounded independent of N, or only increase very slowly, such as $O(\log N)$.)*

It is straightforward to show that, if two matrices A and B of the same size have the low-rank property with the maximum rank of all the off-diagonal ranks bounded by r, then $A + B$ and AB also have the low-rank property, with the maximum rank of all their off-diagonal ranks bounded by $2r$. This will be used often later. In the following subsections, we study the low-rank property of the matrices A in (2.8) and (2.18).

3.2. Low-rank property for constant coefficients. We briefly consider the low-rank property of A when $\alpha(x) \equiv \alpha,\ \beta(x) \equiv \beta$ are constants.

For the case of the Fourier-Galerkin method, the linear system (2.8) is simply a diagonal system of the equations

$$(\alpha + k^2\beta)\tilde{u}_k = \tilde{f}_k, \quad k = -\frac{N}{2}, \ldots, \frac{N}{2}. \tag{3.1}$$

That is, A is a diagonal matrix, and the solution to (2.8) is trivially

$$\tilde{u}_k = \frac{\tilde{f}_k}{\alpha + k^2\beta}, \quad k = -\frac{N}{2}, \ldots, \frac{N}{2}.$$

Next, consider the Chebyshev-Galerkin method. Let α and β be two non-negative constants in (2.9). Then the matrix A in (2.18) has the form

$$A = A_3 \left(\alpha I - \beta A_2^2\right) A_1 = \alpha A_3 A_1 - \beta A_3 A_2^2 A_1, \tag{3.2}$$

which is dense. Since A_1 and A_3 are tridiagonal matrices, their off-diagonal ranks are bounded by 1. The matrix A_2 is upper triangular, and can be shown to have the low-rank property as follows, even though its nonzero entries away from the diagonal *do not decay*. In fact, according to (2.15), $\operatorname{rank}(A_2|_{1:j,j+1:N}) = 2$ for any integer j satisfying $1 \leqslant j \leqslant N$. Thus, we can show that A_3A_1 and $A_3A_2^2A_1$ have off-diagonal rank bounds 2 and 6, respectively. The matrix A in (3.2) thus has an off-diagonal rank bound 8. However, we can further improve such a bound by inspecting the actual forms of A_3A_1 and $A_3A_2^2A_1$.

THEOREM 3.3. *For any integer n satisfying $1 \leqslant n \leqslant N$,*

$$\operatorname{rank}(A|_{1:n,n+1:N}) \leq 4, \quad \operatorname{rank}(A|_{n+1:N,1:n}) \leq 4.$$

Proof. The actual forms of A_3A_1 and $A_3A_2^2A_1$ can be found in [24]. A_3A_1 is a banded matrix with its half bandwidth 2, so it has off-diagonal ranks bounded by 2.

Any off-diagonal block of $A_3A_2^2A_1$ is either a zero block or has the form

$$(A_3A_2^2A_1)|_{1:j,k} = \begin{cases} 4\pi(0,2,0,4,\ldots,0,j-1,0)^T, & k=j+1,j+3,\ldots, \\ 4\pi(1,0,3,0,\ldots,j-2,0,j)^T, & k=j+2,j+4,\ldots, \end{cases}$$

if n is odd, or

$$(A_3A_2^2A_1)|_{1:j,k} = \begin{cases} 4\pi(1,0,3,0,\ldots,j-1,0)^T, & k=j+1,j+3,\ldots, \\ 4\pi(0,2,0,4,\ldots,0,j)^T, & k=j+2,j+4,\ldots, \end{cases}$$

if n is even. Thus, we have

$$\operatorname{rank}((A_3A_2^2A_1)|_{1:j,j+1:N}) \le 2.$$

The desired result then follows from the addition and multiplication of matrices with the low-rank property. ∎

3.3. Low-rank property for "smooth" variable coefficients: Fourier case. For variable coefficients $\alpha(x)$ and $\beta(x)$, the matrix A in (2.8) for the Fourier case is usually dense. However, we show below that if $\alpha(x)$ and $\beta(x)$ are sufficiently "smooth" in the sense that they can be approximated by Fourier series with small finite numbers of terms, then A has the low-rank property.

THEOREM 3.4. *If $\alpha(x)$ can be approximated by an r-term Fourier series within a tolerance τ, then the numerical rank (with respect to the tolerance $O(N\tau)$) of any off-diagonal block of the following matrix is bounded by r:*

$$C \equiv F\mathcal{D}_{\boldsymbol{\alpha}}F^* \quad (\text{or } F^*\mathcal{D}_{\boldsymbol{\alpha}}F),$$

where $\boldsymbol{\alpha} = (\alpha(x_j))_{j=0}^{N-1}$, $\mathcal{D}_{\boldsymbol{\alpha}} = \operatorname{diag}(\alpha)$, and F represents FDFT.

Proof. C is a circulant matrix with the first column

$$c = F\boldsymbol{\alpha}. \tag{3.3}$$

Assume $c = (c_1,\ldots,c_N)^T$. Let $\hat{C}$ be a Hermitian circulant matrix with the first column $\hat{c} = (c_1,\ldots,c_r,0,\ldots 0,c_{N-r+1},\ldots,c_N)^T$. Clearly, $\hat{C}_{ij}$ can be obtained from C by setting $\hat{C}_{ij} = 0$ for $r < |i-j| < N-r$. See Figure 3.1 for an illustration of the nonzero pattern of $\hat{C}$. Clearly, any off-diagonal block of $\hat{C}$ is zero except the lower left and the upper right corners, which are two $r \times r$ triangular matrices. Thus,

$$\operatorname{rank}(\hat{C}|_{1:n,n+1:N}) \le 2r, \quad \operatorname{rank}(\hat{C}|_{n+1:N,1:n}) \le 2r \quad (1 \leqslant n \leqslant N). \tag{3.4}$$

Since $\alpha(x)$ can be approximated by an r-term Fourier series within the tolerance τ, we have $|c_j| = O(\tau)$ for any $j > r$. That is,

$$|C_{ij}| = O(\tau) \quad \text{for} \quad r < |i-j| < N-r.$$

(This can be understood with the aid of Figure 3.1, where the entries of C corresponding to the blank spaces are of magnitudes $O(\tau)$.) Thus, the off-diagonal block $C|_{1:n,n+1:N}$ satisfies

$$\begin{aligned} ||C|_{1:n,n+1:N} - \hat{C}|_{1:n,n+1:N}||_2 &\le \sqrt{n(N-n)} \max_{0<j-i<N-r} |C_{ij}| \\ &\le \frac{N}{2} \max_{0<j-i<N-r} |C_{ij}| = O(N\tau). \end{aligned} \tag{3.5}$$

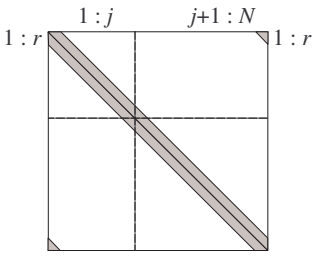


FIG. 3.1. *Nonzero pattern of* $\hat{C}$.

Let σ_{r+1} be the $(r+1)$-st singular value of $C|_{1:n,n+1:N}$ (when all its singular values are ordered from the largest to the smallest). It is known that (see, e.g., [10, Section 3.2.3], with a trivial extension)

$$\sigma_{r+1} = \min_{\mathrm{rank}(\tilde{C})\le r} ||C|_{1:n,n+1:N} - \tilde{C}||_2.$$

(3.4) and (3.5) then mean

$$\sigma_{r+1} \le ||C|_{1:n,n+1:N} - \hat{C}|_{1:n,n+1:N}||_2 = O(N\tau).$$

Therefore, the numerical rank of $C|_{1:n,n+1:N}$ is at most r with respect to the tolerance $O(N\tau)$. Similarly, the result holds for $C|_{n+1:N,1:n}$. ∎

Note that C is also the kernel of a discrete integral transform. Thus, the low-rank property is closely related to the idea of the fast multipole method (FMM) [12, 14].

REMARK 2. σ_{r+1} is usually much smaller than $O(N\tau)$ since the bound in (3.5) often overestimates $||C|_{1:n,n+1:N} - \hat{C}|_{1:n,n+1:N}||_2$.

REMARK 3. Clearly, the 2-norm condition number of $F\mathcal{D}_{\boldsymbol{\alpha}}F^*$ is

$$\kappa(\tilde{F}\mathcal{D}_{\boldsymbol{\alpha}}\tilde{F}^*) = \kappa(\mathcal{D}_{\boldsymbol{\alpha}}) = \frac{\max_i |\alpha(x_i)|}{\min_i |\alpha(x_i)|}, \tag{3.6}$$

where we assume $\min_i |\alpha(x_i)| \neq 0$. Thus, the ratio of the largest and the smallest values in (3.6) measures how much $\alpha(x)$ varies, and also the conditioning of the matrix. If certain $\alpha(x_i)$ is extremely large or small, or if $\alpha(x)$ varies dramatically, then the condition number in (3.6) is large. This may cause difficulties for iterative methods to converge. On the other hand, Theorem 3.4 indicates that the rank structure is actually a much more robust way of studying the numerical solution.

LEMMA 3.5. *If C has the low-rank property, then so do $\Lambda_1 C$, $C\Lambda_2$, $\Lambda_1 C\Lambda_2$, and $\hat{\Lambda}_1 C\hat{\Lambda}_2$, where Λ_1 and Λ_2 are diagonal matrices, $\hat{\Lambda}_1$ and $\hat{\Lambda}_2$ are anti-diagonal matrices, and all the matrices are $N \times N$.*

Proof. It is straightforward to verify the result for $\Lambda_1 C$, $C\Lambda_2$, and $\Lambda_1 C\Lambda_2$ based on diagonal scaling. $\hat{\Lambda}_1 C\hat{\Lambda}_2$ can be rewritten as $(\hat{\Lambda}_1\Pi)(\Pi C\Pi)(\Pi\hat{\Lambda}_2)$, where Π is the anti-identity matrix, $\hat{\Lambda}_1\Pi$ and $\Pi\hat{\Lambda}_2$ are diagonal matrices, and $\Pi C\Pi$ has the low-rank property since its off-diagonal blocks are the same as those of C with appropriate permutations. ∎

We are now ready to present the following result.

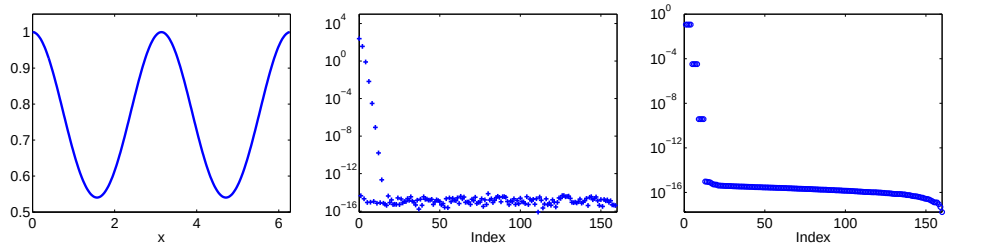
THEOREM 3.6. *If $\alpha(x)$ and $\beta(x)$ can be approximated by r-term Fourier series within a tolerance τ, then the numerical rank of the matrix A (with respect to the tolerance $O(N\tau)$) in (2.8) is of order $O(r)$.*

Proof. Theorem 3.4 means that $F\mathcal{D}_{\boldsymbol{\alpha}}F^*$ and $F\mathcal{D}_{\boldsymbol{\beta}}F^*$ have the low-rank property. The conclusion then follows from Lemma 3.5. ∎

To illustrate the result of the fundamental Theorem 3.4, let us consider the following function:

$$\alpha_1(x) = \cos(\sin x), \tag{3.7}$$

which is plotted in Figure 3.2(i). The function $\alpha_1(x)$ is smooth, and is evaluated at the Fourier collocation points $\{x_j\}_{j=0}^{N-1}$ defined in (2.4). The decay of the Fourier expansion coefficients can be observed in Figure 3.2(ii). To check the low-rank property, we plot the singular values for an off-diagonal block $(F\mathcal{D}_{\boldsymbol{\alpha}_1}F^*)|_{1:\frac{N}{2},\frac{N}{2}+1:N}$ in Figure 3.2(iii). These singular values also decay quickly.



(i) Plot of $\alpha_1(x)$ (ii) Entries of $|F\boldsymbol{\alpha}_1|$ (iii) Off-diagonal singular values

FIG. 3.2. *Plots of $\alpha_1(x) = \cos(\sin x)$ in (3.7), absolute values of the Fourier coefficients $F\boldsymbol{\alpha}_1$, and singular values of the off-diagonal block $(F\mathcal{D}_{\boldsymbol{\alpha}_1}F^*)|_{1:\frac{N}{2},\frac{N}{2}+1:N}$, where $N = 320$.*

Moreover, when N doubles, the numerical rank of $(F\mathcal{D}_{\boldsymbol{\alpha}_1}F^*)|_{1:\frac{N}{2},\frac{N}{2}+1:N}$ remains small and bounded. Even if we use increase the accuracy τ from 10^{-6} to 10^{-12}, the numerical ranks only increase slightly. See Table 3.1. This indicates the feasibility of using a rank structured approximation to solve large problems with high accuracies.

TABLE 3.1
Numerical ranks of the off-diagonal block $(F\mathcal{D}_{\boldsymbol{\alpha}_1}F^*)|_{1:\frac{N}{2},\frac{N}{2}+1:N}$ for $\alpha_1(x)$ in (3.7) with different sizes N and tolerances τ.

N		20	40	80	160	320	640	1280
Numerical rank	$\tau = 10^{-6}$	8	8	8	8	8	8	8
	$\tau = 10^{-12}$	10	12	12	12	12	12	12

3.4. Low-rank property for "smooth" variable coefficients: Chebyshev case. Next, we consider A in (2.18) for the Chebyshev case with variable coefficients $\alpha(x)$ and $\beta(x)$. The matrix A has a form more complicated than in the Fourier case. However, we can still show that A has the low-rank property. This is mainly based on the following result. Here, a function is "smooth" if it can be approximated by a Chebyshev series with a small finite number of terms.

THEOREM 3.7. *If $\alpha(x)$ can be approximated by an r-term Chebyshev series within a tolerance τ, then the numerical rank (with respect to the tolerance $O(N\tau)$) of any off-diagonal block of the following matrix is bounded by $O(r)$:*

$$\tilde{C} \equiv \tilde{F}\mathcal{D}_{\boldsymbol{\alpha}}\tilde{F}^* \quad (\textit{or } \tilde{F}^*\mathcal{D}_{\boldsymbol{\alpha}}\tilde{F}),$$

where $\boldsymbol{\alpha} = (\alpha(x_j))_{j=0}^{N}$, $\mathcal{D}_{\boldsymbol{\alpha}} = \mathrm{diag}(\boldsymbol{\alpha})$, *and* $\tilde{F}$ *represents FDCT.*

Proof. One way is to use the relationship between FDCT and FDFT. In fact, $\tilde{F}$ is real and symmetric. For simplicity, assume F and $\tilde{F}$ have sizes $2N$ and $N+1$, respectively. Clearly,

$$\tilde{F} = A_6 F A_5,$$

where

$$A_5 = \begin{pmatrix} \mathrm{diag}(1, \frac{1}{2}, \ldots, \frac{1}{2}, 1) \\ \begin{pmatrix} 0 & & & 1 \\ \vdots & & & \\ & & \frac{1}{2} & \\ \vdots & & & \\ 0 & \frac{1}{2} & & \end{pmatrix} \end{pmatrix}_{2N\times(N+1)}, \quad A_6 = \begin{pmatrix} I & 0 \end{pmatrix}_{(N+1)\times 2N}.$$

Also let

$$\hat{D} \equiv A_5 \mathcal{D}_{\boldsymbol{\alpha}} A_5^T, \quad \hat{C} \equiv F\hat{D}F^*.$$

Then

$$\tilde{C} = A_6 \hat{C} A_6^T = \hat{C}|_{1:N+1,1:N+1}. \tag{3.8}$$

Since $\mathrm{diag}(\frac{1}{2}, \ldots, \frac{1}{2}, 1) = \frac{1}{2}(I + \mathrm{diag}(0, \ldots, 0, 1))$, we have

$$\hat{D} = \frac{1}{4}\begin{pmatrix} 4\boldsymbol{\alpha}_0 & & \\ & \mathcal{D}_{\boldsymbol{\alpha}|_{1:N}} & \mathcal{D}_{\boldsymbol{\alpha}|_{1:N}} J_N \\ & \mathcal{D}_{\boldsymbol{\alpha}|_{N:-1:1}} J_N & \mathcal{D}_{\boldsymbol{\alpha}|_{N:-1:1}} \end{pmatrix} + E_2,$$

where $\boldsymbol{\alpha}|_{N:-1:1}$ represents $\boldsymbol{\alpha}|_{1:N}$ in its reverse ordering, J_N is the $N \times N$ anti-identity matrix, and E_2 is an appropriate matrix of rank no more than 2. Let

$$H = \begin{pmatrix} 1 & \\ & J_{2N} \end{pmatrix}, \quad G = F\,\mathrm{diag}(\mathcal{D}_{\boldsymbol{\alpha}|_{0:N,N:-1:1}})F^*. \tag{3.9}$$

Then we can rewrite $\hat{D}$ and $\hat{C}$ as

$$\begin{aligned} \hat{D} &= \frac{1}{4}(\mathrm{diag}(\mathcal{D}_{\boldsymbol{\alpha}|_{0:N,N:-1:1}}) + \mathrm{diag}(\mathcal{D}_{\boldsymbol{\alpha}|_{0:N,N:-1:1}})H) + \frac{1}{2}\mathrm{diag}(\boldsymbol{\alpha}_0, 0, \ldots, 0) + E_2 \\ &= \frac{1}{4}(\mathrm{diag}(\mathcal{D}_{\boldsymbol{\alpha}|_{0:N,N:-1:1}}) + \mathrm{diag}(\mathcal{D}_{\boldsymbol{\alpha}|_{0:N,N:-1:1}})H) + E_3, \\ \hat{C} &= \frac{1}{4}(G + F\,\mathrm{diag}(\mathcal{D}_{\boldsymbol{\alpha}|_{0:N,N:-1:1}})HF^*) + FE_3F^*, \end{aligned}$$

respectively, where E_3 is an appropriate matrix of rank no more than 3. According to the property of the FDFT matrix,

$$HF^* = F^*H.$$

Thus,

$$\begin{aligned} \hat{C} &= \frac{1}{4}(G + F\,\mathrm{diag}(\mathcal{D}_{\boldsymbol{\alpha}|_{0:N,N:-1:1}})F^*H) + FE_3F^* \\ &= \frac{1}{4}(G + GH) + FE_3F^*. \end{aligned} \tag{3.10}$$

Since $\alpha(x)$ can be approximated by an r-term Chebyshev series in x within a tolerance τ, it can also be approximated by an $O(r)$-term Fourier series in θ (with the change of variable $x = \cos\theta$)) within a tolerance $O(\tau)$. Thus, Theorem 3.4 applied to G in (3.9) yields the off-diagonal low-rank property of G. The leading $(N+1)\times(N+1)$ diagonal block of GH looks like

$$(GH)|_{1:N+1,1:N+1} = G|_{1:N+1,1:N+1}H|_{1:N+1,1:N+1} + G|_{1:N+1,N+2:2N}H|_{N+2:2N,1:N+1}. \tag{3.11}$$

Since $H|_{1:N+1,1:N+1}$ has only one nonzero entry as in (3.9), the first term on the right-hand side of (3.11) has rank 1. Since $G|_{1:N+1,N+2:2N}$ has numerical rank (with respect to the tolerance $O(N\tau)$) bounded by $O(r)$, so does the second term on the right-hand side of (3.11). Therefore, the numerical rank of $(GH)|_{1:N+1,1:N+1}$ is bounded by $O(r)$. According to (3.8) and (3.10), the numerical rank of $\tilde{C}$ is bounded by $O(r)$. ∎

Therefore, the following result on the low-rank property of A can be naturally obtained based on Theorem 3.7.

THEOREM 3.8. *If $\alpha(x)$ and $\beta(x)$ can be approximated by an r-term Chebyshev series within a tolerance τ, then the numerical rank (with respect to the tolerance $O(N\tau)$) of any off-diagonal block of the matrix A in (2.18) is bounded by $O(r)$.*

REMARK 4. Although the results in Theorems 3.4 and 3.7 can be understood based on the fast decay in the entries of C and $\tilde{C}$ away from the diagonal, respectively, this may not be the case for the entries of A in the above theorem. In fact, for A in the Chebyshev case (2.18), the entries away from the diagonal barely decay.

We can similarly illustrate the low-rank property of $\tilde{F}\mathcal{D}_{\boldsymbol{\alpha}}\tilde{F}^*$. For example, for

$$\alpha_2(x) = \mathrm{e}^x,$$

the off-diagonal numerical ranks with $\tau = 10^{-6}$ and 10^{-12} are bounded by 3 and 5, respectively for N up to 1280, and they stay almost the same when N increases. This indicates the robustness of the rank structure with respect to both τ and N.

3.5. Low-rank property for variable coefficients with steep gradients. The studies in the previous subsections indicate the existence of the low-rank property in problems with constant or smooth coefficients. In particular, as pointed out in Remark 4, the smoothness of $\alpha(x)$ leads to the fast decay of the entries of C in Theorem 3.4 and $\tilde{C}$ in Theorem 3.7 away from the diagonal. However, we emphasize that the decay in the entries is *only sufficient but not necessary* for our structured solver to work, which just requires the low-rank property. (A_2 in (2.14) is such an example with no decay away from the diagonal.)

In fact, if the variable coefficient $\alpha(x)$ has steep gradients and the assumptions in Theorems 3.4 and 3.7 are not satisfied, such a decay is usually very slow. However, the low-rank property still holds if $\alpha(x)$ can be split into the sum of a smooth piece $\alpha_{\mathrm{s}}(x)$ and a residual piece $\alpha_{\mathrm{r}}(x)$:

$$\alpha(x) = \alpha_{\mathrm{s}}(x) + \alpha_{\mathrm{r}}(x), \tag{3.12}$$

with $\alpha_{\mathrm{r}}(x)$ being essentially zero except in a finite number of small intervals where the function exhibits steep gradients (see, e.g., Figure 3.3(i)).

To fix the idea, we consider below the Fourier case, although the argument also applies to the Chebyshev case. Since the off-diagonal numerical rank bound of $F\mathcal{D}_{\boldsymbol{\alpha}_{\mathrm{s}}}F^*$ is small, we only have to show that the off-diagonal numerical rank bound of $F\mathcal{D}_{\boldsymbol{\alpha}_{\mathrm{r}}}F^*$

is also small. Intuitively, the low-rank property of $F\mathcal{D}_{\boldsymbol{\alpha}_\mathrm{r}}F^*$ can be understood as follows. Suppose $\alpha_\mathrm{r}(x)$ has localized steep gradients in small subintervals (a_j, b_j) of width $O(\epsilon)$, $j = 1, \ldots, r_1$. Let N be the number of Fourier-collocation points, hence the size of the matrix $F\mathcal{D}_{\boldsymbol{\alpha}_\mathrm{r}}F^*$, such that

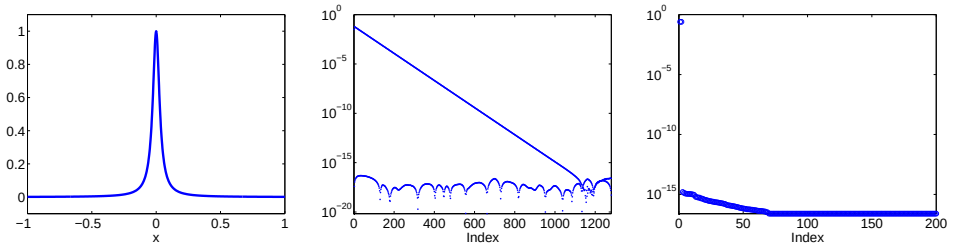
$$\|\alpha_\mathrm{r} - I_N\alpha_\mathrm{r}\|_{L^\infty} \le \tau, \tag{3.13}$$

where $I_N\alpha_\mathrm{r}$ is the Fourier interpolation of α_r, and τ is a given accuracy. It is clear that this N is also the approximate number of Fourier coefficients needed for the N-term truncated Fourier-series to have error bounded by τ (see, e.g. Figure 3.3(ii)). It is well-known that one only needs to have a fixed number (r_2) of Fourier-collocation points within each (a_i, b_i) to fully resolve the steep gradient. Hence, the total number of non-zero values of $I_N\alpha_\mathrm{r}$ at these N collocation points is bounded by r_1r_2, and the total number of collocation points needed for (3.13) in the whole interval $[0, 2\pi)$ is $N = O(\frac{2\pi}{\epsilon})$ which is very big for small ϵ. Thus, the off-diagonal numerical rank of $F\mathcal{D}_{\boldsymbol{\alpha}_\mathrm{r}}F^*$ is bounded by r_1r_2 (see, e.g., Figure 3.3(iii)), which is much smaller than N and is independent of τ. In fact, we observe from Table 3.2 that, for the functions $\alpha_i(x)$ $(i = 3, 4, 5)$ plotted in Figs. 3.3-3.5, the off-diagonal numerical ranks of these matrices do not increase with N.

To illustrate the low-rank property with steep gradients, we look at the following examples:

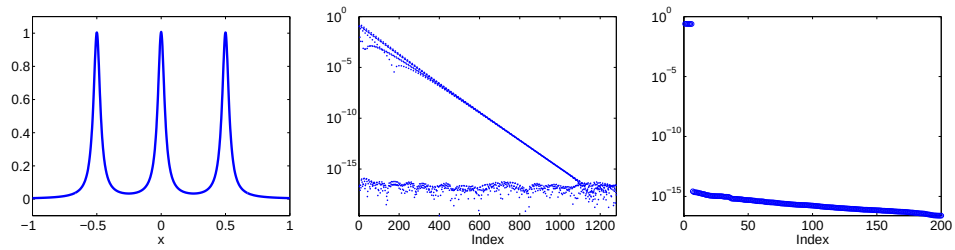
$$\begin{aligned}
\alpha_3(x) &= \frac{1}{1000x^2+1}, \quad x \in (-1, 1),\\
\alpha_4(x) &= \frac{1}{1000(x-0.5)^2+1} + \frac{1}{1000x^2+1} + \frac{1}{1000(x+0.5)^2+1}, \quad x \in (-1, 1),\\
\alpha_5(x) &= -\frac{1}{2}\left(\tanh(100\cdot(2x-3\pi)) + \tanh(-100\cdot(2x-\pi))\right), \quad x \in (0, 2\pi).
\end{aligned}$$

Note that α_3 and α_4 are not periodic but α_5 can essentially be considered as periodic. Hence, we consider the Chebyshev approximations to the functions $\alpha_3(x)$ and $\alpha_4(x)$, and the Fourier approximation to the function $\alpha_5(x)$. We demonstrate the rank structure of the matrices $\tilde{F}\mathcal{D}_{\boldsymbol{\alpha}_3}\tilde{F}^*$, $\tilde{F}\mathcal{D}_{\boldsymbol{\alpha}_4}\tilde{F}^*$ and $F\mathcal{D}_{\boldsymbol{\alpha}_5}F^*$. We observe from Figures 3.3–3.5 and Table 3.2 that the Chebyshev/Fourier expansion coefficients of these functions decay very slowly, but the off-diagonal singular values of these matrices decay quickly, indicating the low-rank property for all the cases.



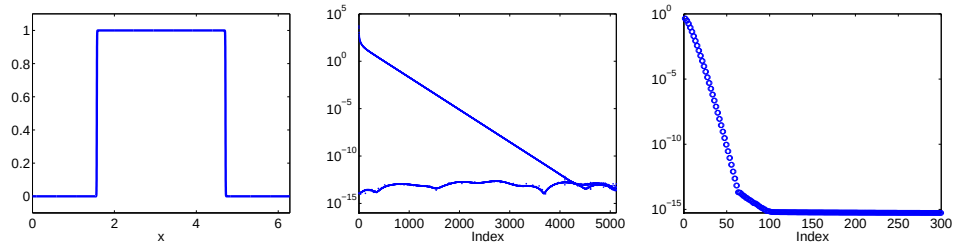
(i) Plot of $\alpha_3(x)$ (ii) Entries of $|\tilde{F}\boldsymbol{\alpha}_3|$ (iii) Off-diagonal singular values

FIG. 3.3. *Plots of* $\alpha_3(x) = \frac{1}{1000x^2+1}$, *absolute values of the Chebyshev coefficients* $\tilde{F}\boldsymbol{\alpha}_3$, *and the first 200 singular values of the off-diagonal block* $(\tilde{F}\mathcal{D}_{\boldsymbol{\alpha}_3}\tilde{F}^*)|_{1:\frac{N}{2},\frac{N}{2}+1:N}$, *where* $N = 1280$.



(i) Plot of $\alpha_4(x)$ (ii) Entries of $|\tilde{F}\boldsymbol{\alpha}_4|$ (iii) Off-diagonal singular values

FIG. 3.4. *Plots of* $\tilde{\alpha}_4(x) = \frac{1}{1000(x-0.5)^2+1} + \frac{1}{1000x^2+1} + \frac{1}{1000(x+0.5)^2+1}$, *absolute values of the Chebyshev coefficients* $\tilde{F}\boldsymbol{\alpha}_4$, *and the first 200 singular values of the off-diagonal block* $(\tilde{F}\mathcal{D}_{\boldsymbol{\alpha}_4}\tilde{F}^*)|_{1:\frac{N}{2},\frac{N}{2}+1:N}$, *where* $N = 1280$.



(i) Plot of $\alpha_5(x)$ (ii) Entries of $|F\boldsymbol{\alpha}_5|$ (iii) Off-diagonal singular values

FIG. 3.5. *Plots of* $\alpha_5(x) = -\frac{1}{2}\left(\tanh(100\cdot(2x-3\pi)) + \tanh(-100\cdot(2x-\pi))\right)$, *absolute values of the Fourier coefficients* $F\boldsymbol{\alpha}_5$, *and the first 300 singular values of the off-diagonal block* $(F\mathcal{D}_{\boldsymbol{\alpha}_5}F^*)|_{1:\frac{N}{2},\frac{N}{2}+1:N}$, *where* $N = 10240$.

TABLE 3.2
Numerical ranks of the off-diagonal block $(\tilde{F}\mathcal{D}_{\boldsymbol{\alpha}_i}\tilde{F}^*)|_{1:\frac{N}{2},\frac{N}{2}+1:N}$ *for* $\alpha_i(x)$, $i = 3, 4$ *and* $(F\mathcal{D}_{\boldsymbol{\alpha}_5}F^*)|_{1:\frac{N}{2},\frac{N}{2}+1:N}$ *with different sizes* N *and tolerances* τ.

N		80	160	320	640	1280	2560	5120	10240
Numerical rank (with $\alpha_3(x)$)	$\tau = 10^{-6}$	2	2	2	2	2	2	2	2
	$\tau = 10^{-12}$	2	2	2	2	2	2	2	3
Numerical rank (with $\alpha_4(x)$)	$\tau = 10^{-6}$	6	6	6	6	6	6	6	6
	$\tau = 10^{-12}$	6	6	6	6	6	6	6	6
Numerical rank (with $\alpha_5(x)$)	$\tau = 10^{-6}$	20	24	28	30	32	32	32	32
	$\tau = 10^{-12}$	30	38	46	54	56	56	56	56

4. HSS structure and matrix-free direct HSS solver. To take advantage of the low-rank property of the coefficient matrices A in the linear systems (2.8) and (2.18), we employ the HSS structure [5, 41] and a matrix-free direct HSS solver [36]. The solver uses some adaptive randomized sampling techniques [15, 35, 42], so that only matrix-vector products are needed to construct an HSS approximation to A, which is then used for the fast solution. These are briefly reviewed in this section.

4.1. HSS representation. In the HSS representation for an $N \times N$ dense matrix A, its off-diagonal blocks are hierarchically represented or approximated by low-rank

forms. For example, a two-level HSS form looks like

$$A = \begin{pmatrix} D_3 & U_3 B_3 V_6^T \\ U_6 B_6 V_3^T & D_6 \end{pmatrix},$$

where

$$D_3 = \begin{pmatrix} D_1 & U_1 B_1 V_2^T \\ U_2 B_2 V_1^T & D_2 \end{pmatrix}, \quad D_6 = \begin{pmatrix} D_4 & U_4 B_4 V_5^T \\ U_5 B_5 V_4^T & D_5 \end{pmatrix},$$
$$U_3 = \begin{pmatrix} U_1 R_1 \\ U_2 R_2 \end{pmatrix}, \quad V_3 = \begin{pmatrix} V_1 W_1 \\ V_2 W_2 \end{pmatrix}, \quad U_6 = \begin{pmatrix} U_4 R_4 \\ U_5 R_5 \end{pmatrix}, \quad V_6 = \begin{pmatrix} V_4 W_4 \\ V_5 W_5 \end{pmatrix}.$$

This can be conveniently understood with the aid of a postordered full binary tree $\mathcal{T}$ called HSS tree. That is, a two-level tree $\mathcal{T}$ defined, with its bottom or first level nodes being labeled $1, 2, 4, 5$, and second level nodes $3, 6$ (see Figure 4.1). Then each node j is associated with the matrices D_j, U_j, R_j, etc. (called *HSS generators*). Thus, the generators define a data-sparse representation of A.

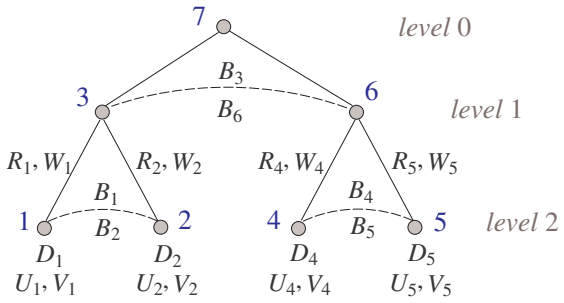


FIG. 4.1. *An HSS tree example.*

Later, for the coefficient matrices A we consider, we do not distinguish between the original matrix and its HSS approximation.

REMARK 5. In Section 3, the off-diagonal blocks $A|_{1:i,i+1:N}$ are inspected (see Definition 3.1). Notice that this is done for all $i = 1, 2, \ldots, N$. These off-diagonal blocks have overlaps and their rank structures have internal correlations or dependencies. The HSS representation provides a natural way to describe such dependencies. It has been well studied that if $A|_{1:i,i+1:N}$, $i = 1, 2, \ldots, N$ are (numerically) low-rank, then A can be represented (or approximated) by an HSS form [5, 37, 41]. In fact, the size of the B_j generators reflects the rank of a certain off-diagonal block $A|_{1:i,i+1:N}$ corresponding to node j and its sibling. Thus, the low-rank property in Section 3 is sufficient to justify the use of HSS approximations here.

4.2. Matrix-free HSS construction. To construct an HSS approximation to a dense matrix A with the low-rank property, traditional methods use the explicit compression of the HSS blocks [41]. Here, since we can quickly compute the product of A and A^* with vectors, we employ a matrix-free HSS construction method proposed in [36].

First, we recall that matrix-vector products for the Fourier case (2.8) and the Chebyshev case (2.18) can generally be computed in $O(N \log N)$ flops using DFT and DCT.

Second, for a low-rank block Φ, if we can quickly compute its product with vectors, then randomized sampling [15, 18, 42] can be used to compress Φ into a low-rank form. In particular, an adaptive randomized strategy in [15, 36] is employed here. The basic idea is to convert the compression of Φ into that of $Y \equiv \Phi X$, where X is a skinny random matrix. Then Y is used to extract a basis matrix for the column space of Φ. The number of random vectors in X is adaptively decided according to a desired accuracy τ.

Finally, the fast matrix-vector multiplication for A can be combined with adaptive randomized sampling to quickly construct an HSS approximation. For the off-diagonal blocks Φ at each level of the hierarchical partition of A, a small number of multiplications of A and A^* with random vectors are used to extract the column and row bases of Φ, respectively. This provides the HSS generators after all the $O(\log N)$ levels are done. When all the off-diagonal blocks are approximated, one more level of matrix-vector products can be used to approximate the diagonal blocks of A.

Assume r is the largest (numerical) rank of all the off-diagonal blocks of A, then the overall cost of this construction is $O(rN \log^2 N)$ and the storage required for the HSS form is $O(rN)$. In this paper, r is dynamically computed for a given relative tolerance τ in the adaptive randomized sampling.

4.3. HSS factorization and solution. After the HSS form is obtained, a fast ULV-type factorization method in [42] (a modified version of [5]) can be used to quickly factorize the matrix. We briefly describe the primary steps of the factorization as follows:

1. Introducing zeros into an off-diagonal block row by QR factorizing its column basis.
2. Partially eliminating the diagonal block.
3. Merging the remaining blocks with those resulting from the step associated with the sibling node of the HSS tree.
4. Forming a smaller HSS matrix with the remaining blocks and repeat.

The factorization costs $O(r^2 N)$ flops. Then, the solution stage for (2.8) and (2.18) costs $O(rN)$ flops. As discussed in the previous section, r is nearly a constant for the approximation tolerance $O(N\tau)$. Therefore, the overall matrix-free HSS solver costs roughly $O(N)$ flops. Similarly, the total memory requirement is about $O(N)$.

5. Numerical results. In this section, we present some numerical experiments to illustrate the efficiency and the accuracy of our fast structured direct spectral methods. We test some important classes of coefficient functions, especially problems with steep gradients and/or high variations, and with even degenerate coefficients. For each case, a large N is required to resolve the problem, due to the "singularities" in the numerical solutions. The code is in Matlab, and is also compared with backslash (the Matlab backslash function). All the tests are carried out on a Thinkpad T430s laptop with 4GB RAM and an Intel i7 core at 2.9GHz. For convenience, we use the following notation in the experiments:

- N means the size of the linear system;
- κ means the 2-norm condition number of the matrix;
- $\mathbf{e} = \frac{\|\tilde{\mathbf{u}}-\mathbf{u}\|_2}{\|\mathbf{u}\|_2}$ is the relative error of the numerical solution $\tilde{\mathbf{u}}$ when the analytical solution is available;
- $\mathbf{r} = \frac{\|A\tilde{\mathbf{u}}-b\|_2}{\|b\|_2}$ is the relative residual of the linear system, i.e. (2.8) or (2.18); $\mathbf{r}$ (original) and $\mathbf{r}$ (n_{it}) mean the residual before and after n_{it} steps of iterative refinement;

- in our fast structured direct spectral methods, $n_{\rm mv}$ is the number of matrix-vector products used in the matrix-free HSS construction, and time$_f$ is the time of the matrix factorization;

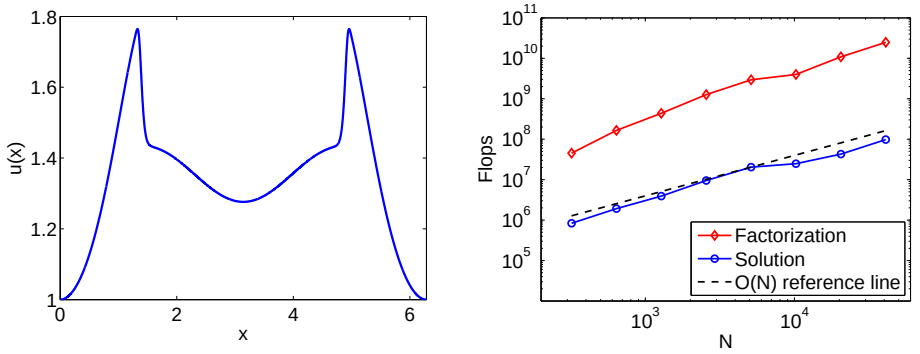
EXAMPLE 1. We first solve a problem with square-wave coefficients and periodic boundary conditions via the fast structured Fourier-Galerkin method. Consider the problem (2.1)–(2.2) with

$$\alpha(x) = \cos(\sin(x)),\ \beta(x) = -\frac{1}{2}\left(\tanh(\gamma(2x-3\pi)) + \tanh(-\gamma(2x-\pi))\right),\ f(x) = K,$$

where γ and K are constants.

As shown in Figure 3.3, $\beta(x)$ is a "square-wave"-like function, which leads to two "cusp points" in the solution (see, e.g., Figures 5.1(i) and 5.2 below). The problem is ill conditioned, so that iterative methods have difficulties converging even for small N. Finding an effective preconditioner is not simple either, especially since A is not explicitly formed.

To see the complexity of the fast structured method clearly, we show the factorization and solution flops in Figure 5.1(i). $\tau = 10^{-7}$ is used here (and also in the later examples). The results roughly follow the $O(N)$ complexity line.



(ii) Numerical solution (based on $N = 5120$) (i) Factorization and solution complexity

FIG. 5.1. *Example 1: Performance of the fast structured solver for* $\gamma = 10$.

Additional results are reported in Table 5.1, including the condition numbers of the matrices, the number $n_{\rm mv}$ of matrix-vector products needed by the fast solver to construction the HSS approximations, and the accuracies. When N doubles, $n_{\rm mv}$ increases slowly, which is consistent with the discussions in Section 4.2. The fast solver produces approximate solutions with reasonable accuracies, followed by few steps of iterative refinement to reach high accuracies. Later, we only show the accuracies after iterative refinement.

We would also like to mention that it is impractical to use regular dense direct solvers, which cost $O(N^3)$ flops to factorize the matrix with $O(N^2)$ storage. We compare the CPU time of our fast structured solver and the Matlab backslash in Table 5.2. For smaller problem sizes, we run the tests 10 times and report the average HSS construction, factorization, and solution time. Similarly, for the Matlab backslash, we measure the average time for forming the dense matrix, computing the LU factorization, and performing the triangular solution. It can be observed that the time of the

TABLE 5.1
Example 1: $n_{\rm mv}$ and accuracies of the fast structured solver, where $\gamma = 10$.

N	κ	$n_{\rm mv}$	$\mathbf{r}$ (original)	$\mathbf{r}$ ($n_{\rm it}$)
320	$4.28e4$	308	$3.51e-9$	$1.07e-14$ (2)
640	$1.73e5$	544	$1.35e-8$	$2.48e-14$ (2)
1280	$6.97e5$	544	$1.65e-8$	$5.10e-14$ (2)
2560	$2.79e6$	416	$3.38e-7$	$9.76e-14$ (3)
5120	$1.12e7$	520	$7.19e-7$	$2.12e-13$ (3)
10240		660	$6.95e-6$	$3.47e-13$ (4)
20480		880	$1.28e-5$	$7.73e-11$ (5)

fast structured solver scales roughly linearly in N, while the Matlab backslash quickly becomes much slower, although the latter is highly optimized. We note that forming and storing the dense matrix in dense direct solvers has other serious drawbacks, which are avoided in our structured direct solver.

TABLE 5.2
Example 1: Comparison of the time (in seconds) between our fast structured solver and Matlab backslash, where $\gamma = 10$.

N	Fast structured			Matlab backslash		
	HSS construction	Factorization	Solution	Matrix formation	Factorization	Solution
320	0.230	0.027	0.006	0.047	0.009	0.001
640	0.527	0.089	0.014	0.359	0.045	0.004
1280	0.806	0.180	0.018	3.023	0.302	0.016
2560	2.156	0.472	0.039	22.488	2.183	0.063
5120	5.349	1.179	0.089	178.622	15.952	0.247
10240	8.918	2.666	0.154	4806.939	244.303	1.807

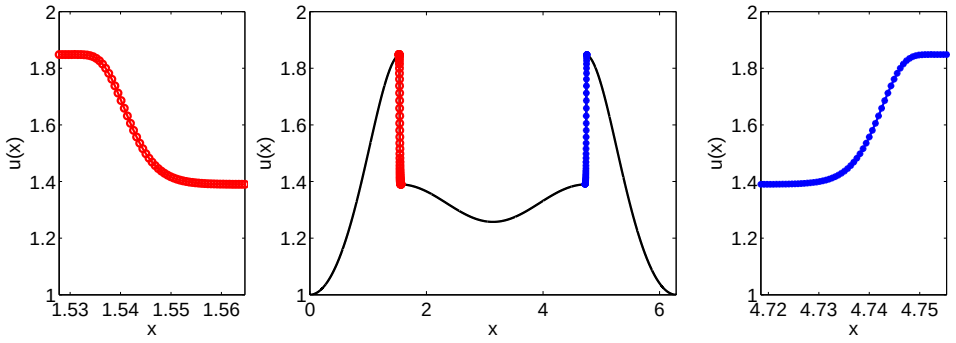
Furthermore, we consider the extreme case $\gamma = 100$ corresponding to Figure 3.3. The efficiency and the accuracy of our method are shown in Table 5.3. The numerical solution is illustrated in Figure 5.2, which has two steep thin interior layers near the points $x = \pi/2$ and $x = 3\pi/2$.

TABLE 5.3
Example 1: Numerical results of the fast structured solver for the extreme case $\gamma = 100$.

N	κ	$n_{\rm mv}$	Time$_f$	$\mathbf{r}$ ($n_{\rm it}$)
2560	$3.02e6$	700	0.69	$3.23e-13$ (4)
5120	$1.21e7$	980	1.78	$7.28e-13$ (5)
10240		1020	6.95	$3.00e-11$ (5)
20480		1080	17.90	$2.55e-09$ (6)

In addition, since our method is a direct solver, the HSS construction, factorization, and solution costs are independent of the right-hand side. The iterative refinement step may depend on the right-hand side. However, the total number of refinement steps is very small for either a constant right-hand side or a random one, and the difference is nearly negligible.

EXAMPLE 2. Next, we apply the fast structured Chebyshev-Galerkin method to the problem (2.9)–(2.10) with delta-like coefficients and Dirichlet boundary conditions,



(i) Zoomed in near $\pi/2$ (ii) Overall solution (iii) Zoomed in near $3\pi/2$

FIG. 5.2. *Example 1: Numerical solution of the fast solver for the extreme case* $\gamma = 100$*, where* $N = 10240$.

where

$$\alpha(x) = \mathrm{e}^x, \quad \beta(x) = \frac{1}{ax^2+1},$$
$$f(x) = 1.$$

The shape of $\beta(x)$ for large a is similar to the delta function. See Figure 5.3 for various a. When a is large, iterative methods can barely converge. For example, for $a = 1000$, CG fails to converge even for small problems. In fact, even if we somehow manage to extract the Jacobi preconditioner (which may not be practical due to the unavailability of A), the accuracy of CG is still very low.

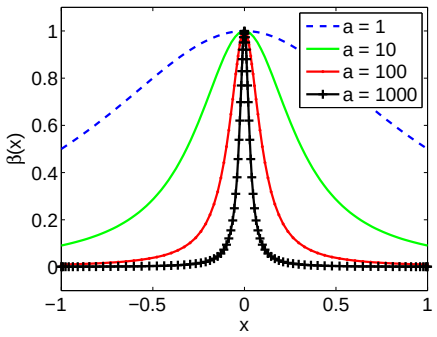
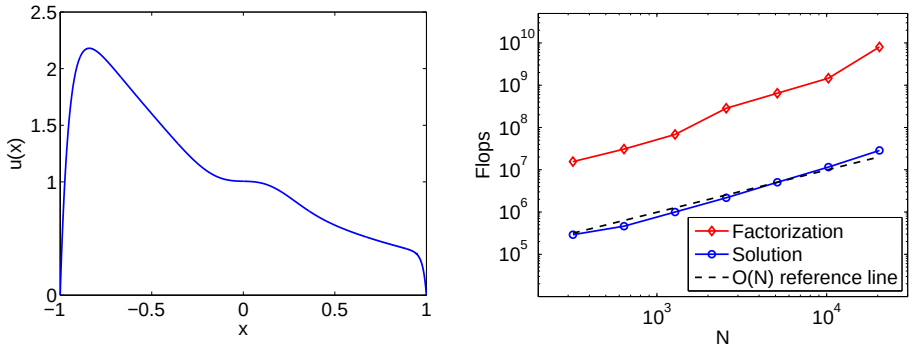


FIG. 5.3. *Example 2: The coefficient function* $\beta(x)$ *for various* a.

On the other hand, our structured solver obtains an accurate solution (Figure 5.4(i)) with nearly linear complexity (Figure 5.4 (ii)). The results such as n_{mv} and the accuracies are shown in Table 5.4, which illustrates the significant advantages of our solver.

Furthermore, we consider the extreme case $a = 10^{10}$. For $N = 5120$, A has condition number 9.3×10^{10} and is severely ill conditioned. However, it only takes $n_{\mathrm{mv}} = 308$ steps to construct an HSS approximation with a small numerical rank. The HSS form is used to solve the problem to reach an accuracy of $\mathbf{r} = 6.27 \times 10^{-12}$



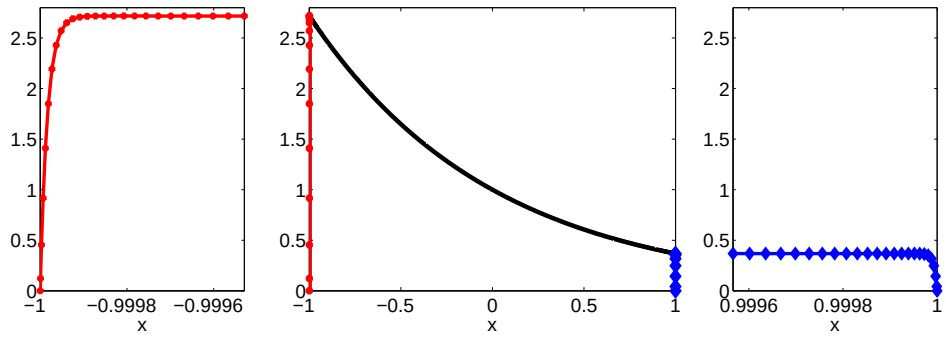
(i) Numerical solution (based on $N = 2560$) (ii) Factorization and solution complexity

FIG. 5.4. *Example 2: Performance of the fast structured solver for $a = 1000$.*

TABLE 5.4
Example 2: Numerical results of the fast structured solver for $a = 1000$.

N	κ	n_{mv}	Time_f	$\mathbf{r}$ (n_{it})
320	$9.18e5$	168	0.02	$3.87e-14$ (4)
640	$4.19e6$	168	0.01	$9.19e-14$ (3)
1280	$1.82e7$	187	0.03	$1.71e-13$ (2)
2560	$7.70e7$	165	0.07	$5.97e-13$ (4)
5120	$3.19e8$	200	0.18	$1.21e-12$ (3)
10240		240	0.37	$2.23e-12$ (4)

after few steps of iterative refinements. See Figure 5.5 for the numerical solution, which has steep thin layers near the two endpoints $x = \pm 1$.



(i) Zoomed in near -1 (ii) Overall solution (iii) Zoomed in near 1

FIG. 5.5. *Example 2: Numerical solution of the fast solver for the extreme case $a = 10^{10}$, where $N = 5120$.*

EXAMPLE 3. Next, we consider the following problem with degenerate coefficients and Neumann boundary conditions:

$$e^x u - (\sin^2(2\pi x)u_x)_x = 1, \quad x \in (-1,1),$$
$$u'(-1) = u'(1) = 0.$$

The general procedure of our fast structured Chebyshev-Galerkin method for solving the problem is the same as what has been shown in Section 2.2, except that the basis functions should be chosen as follows due to the Neumann boundary conditions:

$$\psi_k(x) = T_k(x) - \left(\frac{k}{k+2}\right)^2 T_{k+2}(x).$$

The coefficient $\beta(x) = \sin^2(2\pi x)$ degenerates at multiple points. See Figure 5.6. which leads to the "jumps" in the solution (see, e.g., Figure 5.7(i)). We observe that the solution has many interior layers located at the zeroes of $\beta(x)$. Such a scenario can happen, for example, for problems with internal layers and for Cahn-Hilliard equations with concentration dependent mobility. In this case, the ratio of the maximum value to the minimum is infinity. Iterative methods such as CG (with possibly the Jacobi preconditioner) fail to converge.

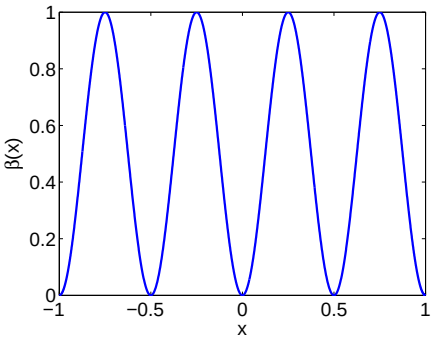
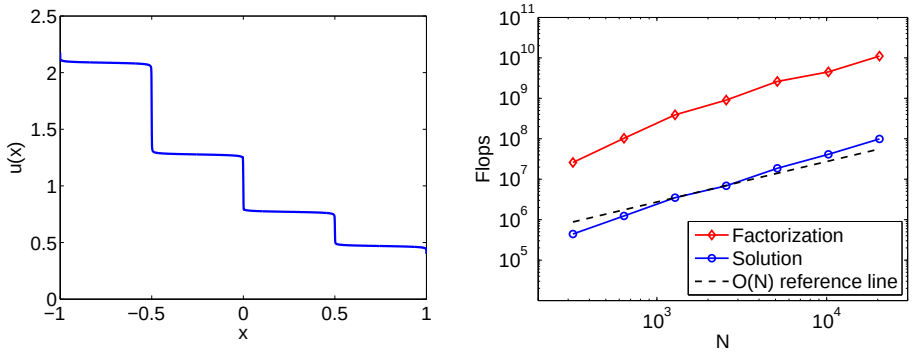


FIG. 5.6. *Example 3: The coefficient function $\beta(x)$.*

Fortunately, the corresponding system matrix still has the low-rank property and can be solved efficiently by our fast structured method. We achieved both high accuracies and nearly linear complexity just like in the previous examples for such a difficult problem. See Figure 5.7 and Table 5.5 for the detailed reports.

TABLE 5.5
Example 3: Numerical results of the fast structured solver.

N	κ	n_{mv}	Time_f	$\mathbf{r}$ (n_{it})
320	$1.46e07$	264	0.01	$4.28e-13$ (2)
640	$2.40e08$	352	0.03	$1.94e-12$ (2)
1280	$3.92e09$	440	0.09	$1.01e-11$ (2)
2560	$6.38e10$	440	0.18	$5.19e-11$ (2)
5120	$1.04e12$	460	0.37	$3.27e-10$ (3)
10240		460	0.75	$6.45e-10$ (4)



(i) Numerical solution (based on $N = 2560$) (ii) Factorization and solution complexity

FIG. 5.7. *Example 3: Performance of CG and complexity of our fast structured solver.*

EXAMPLE 4. Finally, we consider a problem where the analytical solution is known. We solve the problem (2.1)–(2.2) with

$$\alpha(x) = \beta(x) = \cos(\sin(x)),$$

and $f(x)$ is chosen such that the exact solution is

$$u(x) = \mathrm{e}^{1+\sin(kx)},$$

where k is an integer.

For $k = 10$ and 100, the numerical results are shown in Table 5.6. When N increases, the relative error and the residual approach the desired accuracy. The performance of the solver in terms of the efficiency is similar to that in the previous examples. Note that for $k = 100$, the solution is highly oscillatory. However, the method can still achieve high accuracy for reasonably large N. As N increases, the relative error $\mathbf{e}$ decays exponentially. The computation time is only slightly larger than that for $k = 10$.

REMARK 6. The amplification of the HSS approximation error by the condition number may affect the accuracy. In fact, even standard direct solvers have similar amplification (of the numerical errors) by the condition number. In the field of direct solutions, such amplification is usually acceptable in practice.

6. Concluding remarks. We developed fast structured direct spectral Galerkin methods for 1D differential equations with variable coefficients. First, we gave systematic analysis of a low-rank property in Fourier- and Chebyshev-spectral methods. Then we presented a fast matrix-free algorithm based on HSS structures and randomized sampling for the solutions of the linear systems arising from spectral discretizations. The solver typically achieves an asymptotic computational complexity of about $O(N)$. The key to the success of the methods lies on two ingredients: (i) the low-rank property of the underlying matrices; (ii) the fast matrix-free HSS construction based on matrix-vector products. Both the theory and the numerical experiments support our claims regarding the accuracy and the efficiency of our method.

We observe that the proofs are purely algebraic and do not depend on the ellipticity. Hence, the method is expected to be applicable to some other types of

TABLE 5.6
Example 4: Performance of the structured solver.

N	n_{mv}	Original accuracy		After iterative refinement			Time (s)
		e	**r**	n_{it}	**e**	**r**	
80	68	$5.80e+00$	$1.79e-08$	2	$5.80e+00$	$2.11e-17$	$2.09e-02$
160	92	$8.44e-04$	$3.07e-10$	2	$8.44e-04$	$2.10e-17$	$3.84e-02$
320	100	$1.39e-07$	$6.92e-09$	2	$7.42e-14$	$1.35e-16$	$1.08e-01$
640	120	$4.90e-08$	$6.52e-09$	2	$9.00e-15$	$1.84e-17$	$1.86e-01$
1280	140	$1.24e-07$	$9.41e-09$	2	$1.38e-14$	$9.70e-17$	$4.28e-01$

(i) $k = 10$

N	n_{mv}	Original accuracy		After iterative refinement			Time (s)
		e	**r**	n_{it}	**e**	**r**	
160	104	$5.80e+02$	$1.37e-09$	1	$5.80e+02$	$6.90e-17$	$3.69e-02$
320	132	$4.65e+00$	$6.23e-11$	2	$4.65e+00$	$5.20e-17$	$8.87e-02$
640	120	$7.21e-03$	$6.28e-11$	2	$7.21e-03$	$2.62e-17$	$1.87e-01$
1280	140	$7.13e-07$	$1.02e-10$	2	$7.04e-07$	$5.99e-17$	$4.32e-01$
2560	160	$4.00e-08$	$3.01e-11$	2	$4.16e-12$	$3.34e-17$	$9.77e-01$

(ii) $k = 100$

differential equations with variable coefficients in one-dimension. Examples include $u_t = -(a(x)u_x)_x + F(u)$ and $-(a(x)u_x)_x + F(u) = 0$, where $F(u)$ is a certain function that may be nonlinear in u. In such cases, the approximation of the resulting discretized matrix involves the sum of HSS matrices and narrow banded matrices (with possible diagonal scaling). Moreover, the method is particularly effective for problems with steep gradients and/or large variations, and even problems with degenerate coefficients.

This methodology can be potentially extended to the following cases:

- *Initial-boundary-value problems.* After implicit or semi-implicit methods for temporal discretizations, the time-dependent evolution equations, such as heat equation, Burgers equation, Navier-Stokes equations etc., would become elliptic equations about spatial variables like (1.1) at each time step. Besides, since the number of time steps is usually very large, a large number of solutions to a linear system like (2.8) or (2.18) with the same coefficient matrix but different right-hand sides are required. In such situations, the fast direct solver shown in Section 4 is much more efficient than iterative solvers.
- *High-order equations and coupled systems of second-order equations.* Combining the method shown in [7], one can likely construct fast direct spectral Galerkin solvers for higher order problems, such as biharmonic equations, Cahn-Hilliard systems, and plate blending problems of elasticity.

We restricted our attention to the spectral Galerkin method for one-dimensional problems with rigorous analysis and ample numerical examples. Actually, the theoretical and numerical frameworks presented in this paper are essential for extension to multi-dimensional problems and spectral collocation methods. We are currently working on the following situations:

- *Tensor-product based spectral methods.* For separable equations in multi-dimensions, the solution process can be reduced to solving a sequence of one-dimensional problems [24, 26]. Here the bottleneck is the HSS eigenvalue decomposition and eigenmatrix-vector multiplication along each direction.

For certain cases, we can use a recent superfast HSS eigensolver in [33]. Based on the tensor products and the HSS eigenvalue decomposition along each direction, we can find a structured eigenvalue decomposition of the overall discretized matrix. The eigenmatrix is in a structured form and can be applied to a vector in nearly linear complexity. This leads to the solution of the overall problem in nearly linear complexity. The details will appear in [27].

- *Multi-layer structured solvers for multi-dimensional problems.* For two and three dimensional problems, a simple HSS structure is generally not practical. Recently, a multi-layer structure is designed in [39]. That is, if the discretized matrix is approximated by an outer (non-compact) HSS form, its generators are further structured. This leads to a multi-layer structured form that can be quickly factorized.
- *Sparse spectral methods for higher dimensional problems.* Shen, et al. proposed sparse spectral methods for some higher dimensional problems, where the appropriate intermediate dense submatrices also enjoy the low-rank property [28, 30, 31]. It is hopeful that the fast direct solver based on rank-structured matrices could be generalized to these higher-dimensional cases.
- *Fast structured spectral collocation methods.* Although the differential matrices in spectral collocation methods are totally dense, they are numerically observed to also enjoy the low-rank property, fortunately. It indicates that we can first construct the HSS approximation to the differential matrices and store them in advance, and then the linear systems resulting from collocation discretization of differential equations could be solved quickly [29].

Our preliminary results indicate that it is, in principle, possible to extend our structured solvers to these situations, although their efficient implementations are by no means trivial.

Finally, we would like to mention an interesting solver proposed in [21] very recently. Their strategy is based on a totally different approach — the tau-formulation, and on an essential assumption that the variable coefficients can be accurately approximated by low-order polynomials. The decay of the entries would yield a solver with $O(m^2N)$ complexity, where m is the number of Chebyshev coefficients needed to resolve the variable coefficients. For smooth coefficients, m is very small and a straightforward truncation to a banded approximation is much simpler than the construction of structured approximations. However, for coefficients with steep gradients, a large m is needed. Our direct solver yields nearly linear computational complexity independent of such m, and does not assume or rely on any fast decay of the entries of A in (2.8) or (2.18). More specifically, our complexity is $O(rN\log^2 N) + O(r^2N)$, where r is small even for coefficients with steep gradients or variations. Note that the cost $O(rN\log^2 N)$ is for the matrix-free HSS construction. (In contrast, the solver in [21] for smooth coefficients forms the entries of A explicitly and then truncate.) Our matrix-free construction aims at more general cases (especially with steep gradients or variations in the coefficients) so that we do not need to explicitly form the matrix entries.

Acknowledgements. We thank Yuanzhe Xi for his help with the numerical tests. The authors also thank the anonymous referees for valuable comments and suggestions which led to a significant improvement of the presentation.

REFERENCES

[1] M. Bebendorf, *Efficient inversion of Galerkin matrices of general second-order elliptic differential operators with nonsmooth coefficients*, Math. Comp., 74 (2005), pp. 1179–1199.
[2] M. Bebendorf and W. Hackbusch, *Existence of H-matrix approximants to the inverse FE-matrix of elliptic operators with L^∞ coefficients*, Numer. Math., 95 (2003), pp. 1–28.
[3] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang, *Spectral Methods: Fundamentals in Single Domains*, Springer-Verlag, Berlin-Heidelberg, 2006.
[4] C. Canuto and A. Quarteroni, *Preconditioner minimal residual methods for Chebyshev spectral calculations*, J. Comput. Phys., 60 (1985), pp. 315–337.
[5] S. Chandrasekaran, P. Dewilde, M. Gu, and T. Pals, *A fast ULV decomposition solver for hierarchically semiseparable representations*, SIAM J. Matrix. Anal. Appl., 28 (2006), pp. 603–622.
[6] S. Chandrasekaran, P. Dewilde, M. Gu, and N. Somasunderam, *On the numerical rank of the off-diagonal blocks of Schur complements of discretized elliptic PDEs*, SIAM J. Matrix Anal. Appl., 31 (2010), pp. 2261–2290.
[7] F. Chen and J. Shen, *Efficient spectral-Galerkin methods for systems of coupled second-order equations and their applications*, J. Comp. Phys., 231 (2012), pp. 5016–5028.
[8] J. W. Cooley and J. W. Tukey, *An algorithm for the machine calculation of complex Fourier series*, Math. Comput., 19 (1965), pp. 297–301.
[9] E. A. Coutsias, T. Hagstrom, J. S. Hesthaven, and D. Torres, *Integration preconditioners for differential operators in spectral τ-methods*, in Proceedings of the Third International Conference on Spectral and High Order Methods, Houston, TX, 1996, pp. 21???38.
[10] J. W. Demmel, *Applied numerical linear algebra*, SIAM, Philadelphia, PA, (1997).
[11] M. O. Deville and E. H. Mund, *Chebyshev pseudospectral solution of second-order elliptic equations with finite element preconditioning*, J. Comput. Phys., 60 (1985), pp. 517–533.
[12] A. Dutt and V. Rokhlin, *Fast fourier transforms for nonequispaced data*, SIAM J. Sci. Comput., 14 (1993), pp. 1368–1393.
[13] D. Gottlieb and S. A. Orszag, *Numerical Analysis of Spectral Methods: Theory and Applications*, no. 26 in CBMS-NSF Regional Conference Series in Applied Mathematics, SIAM, Philadelphia, 1977.
[14] L. Greengard and V. Rokhlin, *A fast algorithm for particle simulations*, J. Comp. Phys., 73 (1987), pp. 325–348.
[15] N. Halko, P. G. Martinsson, and J. A. Tropp, *Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions*, SIAM Review, 53 (2011), pp. 217–288.
[16] S. D. Kim and S. V. Parter, *Preconditioning Chebyshev spectral collocation method for elliptic partial differential equations*, SIAM J. Numer. Anal., 33 (1996), pp. 2375–2400.
[17] W. Y. Kong, J. Bremerb, and V. Rokhlin, *An adaptive fast direct solver for boundary integral equations in two dimensions*, Appl. Comput. Harmon. Anal., 231 (2011), pp. 346–369.
[18] L. Lin, J. Lu, and L. Ying, *Fast construction of hierarchical matrix representation from matrix-vector multiplication*, J. Comput. Phys., 230 (2011), pp. 4071–4087.
[19] P. G. Martinsson, *A fast direct solver for a class of elliptic partial differential equations*, J. Sci. Comput., 38 (2009), pp. 316–330.
[20] P. G. Martinsson and V. Rokhlin, *A fast direct solver for boundary integral equations in two dimensions*, J. Comp. Phys., 205 (2005), pp. 1–13.
[21] S. Olver and A. Townsend, *A fast and well-conditioned spectral method*, SIAM Review, 55 (2013), pp. 462–489.
[22] S. A. Orszag, *Spectral methods for complex geometries*, J. Comp. Phys., 37 (1980), pp. 70–92.
[23] P. Schmitz and L. Ying, *A fast direct solver for elliptic problems on general meshes in 2D*, J. Comp. Phys., 231 (2012), pp. 1314–1338.
[24] J. Shen, *Efficient spectral-Galerkin method II: Direct solvers for second- and fourth-order equations using Chebyshev polynomials*, SIAM J. Sci. Comput., 16 (1995), pp. 74–87.
[25] J. Shen and T. Tang, *Spectral and High-Order Methods with Applications*, Science Press of China, Beijing, 2006.
[26] J. Shen, T. Tang, and L.-L. Wang, *Spectral Methods: Algorithms, Analysis and Applications*, no. 41 in Springer Series in Computational Mathematics, Springer-Verlag, Berlin-Heidelberg, 2011.
[27] J. Shen, J. Vogel, Y. Wang, and J. Xia, *Fast structured direct spectral methods for differential equations with variable coefficients: two dimensional case*, in preparation, (2015).
[28] J. Shen and L.-L. Wang, *Sparse spectral approximations of high-dimensional problems based on hyperbolic cross*, SIAM J. Numer. Anal., 48 (2010), pp. 1087–1109.
[29] J. Shen, Y. Wang, and J. Xia, *Fast structured spectral collocation methods*, in preparation,

(2014).
[30] J. Shen and H. Yu, *Efficient spectral sparse grid methods and applications to high dimensional elliptic problems*, SIAM J. Sci. Comput., 32 (2010), pp. 3228–3250.
[31] J. Shen and H. Yu, *Efficient spectral sparse grid methods and applications to high dimensional elliptic problems II: Unbounded domains*, SIAM J. Sci. Comput., 34 (2012), pp. A1141–A1164.
[32] R. Vandebril, M. Van Barel, G. Golub, and N. Mastronardi, *A bibliography on semiseparable matrices*, Calcolo, 42 (2005), pp. 249–270.
[33] J. Vogel, J. Xia, S. Cauley, and V. Balakrishnan, *Superfast divide-and-conquer method and perturbation analysis for structured eigenvalue solutions*, SIAM J. Sci. Comput., submitted, 2015.
[34] L.-L. Wang, M. D. Samson, and X. Zhao, *A well-conditioned collocation method using a pseudospectral integration matrix*, SIAM J. Sci. Comput., 36 (2014), pp. A907–A929.
[35] F. Woolfe, E. Liberty, V. Rokhlin, and M. Tygert, *A fast randomized algorithm for approximation of matrices*, Appl. Comput. Harmon. Anal., 25 (2008), pp. 335–366.
[36] Y. Xi, J. Xia, and R. Chan, *A fast randomized eigensolver with structured ldl factorization update*, SIAM J. Matrix Anal. Appl., 35 (2014), pp. 974–996.
[37] J. Xia, *On the complexity of some hierarchical structured matrix algorithms*, SIAM J. Matrix Anal. Appl., 33 (2012), pp. 388–410.
[38] J. Xia, *Randomized sparse direct solvers for large discretized PDEs*, Presentation in the International Conference on Mathematical Modeling, Analysis and Computation, Xiamen, China, July 2012, (2012).
[39] J. Xia, *Multi-layer hierarchically semiseparable structures*, GMIG Report, Purdue University, (2015).
[40] J. Xia, S. Chandrasekaran, M. Gu, and X. S. Li, *Superfast multifrontal method for large structured linear systems of equations*, SIAM J. Matrix Anal. Appl., 31 (2009), pp. 1382–1411.
[41] J. Xia, S. Chandrasekaran, M. Gu, and X. S. Li, *Fast algorithms for hierarchically semiseparable matrices*, Numer. Linear Algebra Appl., 17 (2010), pp. 953–976.
[42] J. Xia, Y. Xi, and M. Gu, *A superfast structured solver for Toeplitz linear systems via randomized sampling*, SIAM J. Matrix Anal. Appl., 33 (2012), pp. 837–858.