Chapter 1
Numerical homogenization via approximation of the solution operator

A. Gillman, P. Young, and P.G. Martinsson

Abstract The paper describes techniques for constructing simplified models for problems governed by elliptic equations involving heterogeneous media. Examples of problems under consideration include electro-statics and linear elasticity in composite materials, and flows in porous media. A common approach to such problems is to either up-scale the governing differential equation and then discretize the up-scaled equation, or to construct a discrete problem whose solution approximates the solution to the original problem under some constraints on the permissible loads. In contrast, the current paper suggests that it is in many situations advantageous to directly approximate the *solution operator* to the original differential equation. Such an approach has become feasible due to recent advances in numerical analysis, and can in a natural way handle situations that are challenging to existing techniques, such as those involving, *e.g.* concentrated loads, boundary effects, and irregular micro-structures. The capabilities of the proposed methodology is illustrated by numerical examples involving domains that are loaded on the boundary only, in which case the solution operator is a boundary integral operator such as, *e.g.*, a Neumann-to–Dirichlet operator.

1.1 Introduction

1.1.1 Background

The purpose of this report is to draw attention to a number of recent developments in computational harmonic analysis that may prove helpful to the construction of simplified models for heterogeneous media. We consider problems modeled by elliptic PDEs such as electrostatics and linear elasticity in composite materials, and Stokes' flow in porous media.

Many different solution approaches have been proposed for the type of problems under consideration. A classical technique that works relatively well in situations where there is a clear separation of length-scales is to derive so called *homogenized equations* which accurately model the macro-scale behavior of the constitutive equations without fully resolving the micro-structure. The homogenized equations can sometimes be derived analytically, but they are typically obtained from numerically solving a set of equations defined on a *Representative Volume Element* (RVE). An unfortunate aspect of this approach is that its accuracy is held hostage to many factors that are outside of the control of the modeler. Phenomena that tend to lead to less accurate solutions include:

1. Concentrated loads.
2. Boundaries, in particular non-smooth boundaries.
3. Irregular micro-structures.

The accuracy cannot readily be improved using generic techniques, but a number of strategies for developing coarse-grained models for specific situations have been developed. The most popular ones

appear to be variations of finite element methods in which a discretization on the macro-scale is constructed by solving a set of local problems defined on a representative collection of patches of fully resolved micro-structure.

We contend that it is in many situations advantageous to approximate the *solution operator*, rather than the *differential operator*. For the elliptic problems under consideration in this paper, the solution operator takes the form of an integral operator with the Green's function of the problem as its kernel. That such operators should in principle allow compressed representations has been known for some time (at least since [4]), but efficient techniques for actually computing them have become available only recently.

To illustrate the viability of the proposed techniques, we demonstrate how they apply to a couple of archetypical model problems. We first consider situations in which the micro-structure needs to be fully resolved and a coarse-grained model be constructed computationally. We show that this computation can be executed efficiently, and that once it has been, the reduced model allows for very fast solves, and is highly accurate even in situations that are challenging to existing coarse-graining methods. We then show that the proposed methods can fully exploit the simplifications possible when an accurate model of the material can be derived from computations on an RVE.

1.1.2 Mathematical problem formulation

While the ideas described are applicable in a broad range of environments, we will for expositional clarity focus on scalar elliptic boundary value problems defined on some regular domain $\Omega \subset \mathbb{R}^2$ with boundary Γ. Specifically, we consider Neumann problems of the form

$$\begin{cases} -\nabla \cdot \big(a(x) \cdot \nabla u(x)\big) = 0, & x \in \Omega, \\ u_n(x) = f(x), & x \in \Gamma, \end{cases} \tag{1.1}$$

where $a : \Omega \to \mathbb{R}^{2\times 2}$ is a matrix-valued function that varies "rapidly" (on the length-scale of the micro-structure), and where $u_n(x)$ denotes the normal derivative of u at $x \in \Gamma$. Our objective is to rapidly construct $u|_\Gamma$, from a given boundary function f. We are interested both in the situation where we are allowed a pre-computation involving some given function a, and in the situation in which a is specified probabilistically.

Some of our numerical work will focus on the special case where (1.1) represents a two-phase material. To be precise, we suppose that Ω can be partitioned into two disjoint "phases," $\bar{\Omega} = \bar{\Omega}_1 \cup \bar{\Omega}_2$, and that there exist constants a_1 and a_2 such that

$$a(x) = \begin{cases} a_1\, I, & x \in \Omega_1, \\ a_2\, I, & x \in \Omega_2, \end{cases}$$

where I is the identity matrix. We further suppose that $\bar{\Omega}_2$ is wholly contained inside Ω, and let Γ_{int} denote the boundary between Ω_1 and Ω_2, see Figure 1.1. Then (1.1) can more clearly be written

$$\begin{cases} -a_1 \Delta u(x) = 0, & x \in \Omega_1, \\ -a_2 \Delta u(x) = 0, & x \in \Omega_2, \\ [u](x) = 0, & x \in \Gamma_{\text{int}}, \\ [a\, u_n](x) = 0, & x \in \Gamma_{\text{int}}, \\ u_n(x) = f(x), & x \in \Gamma, \end{cases} \tag{1.2}$$

where for $x \in \Gamma$, $[u](x)$ and $[a\, u_n](x)$ denote the jumps in the potential and in the *flow* $-a(x)\nabla u(x)$ in the normal direction, respectively.

While the current paper concerns only situations modeled by equations of the types (1.1) and (1.2), the methodology extends to more general elliptic differential equations, see Section 1.5.

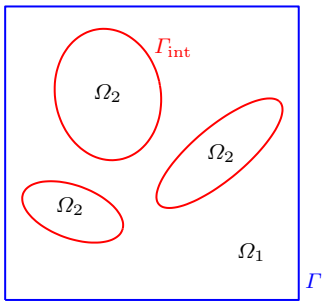


Fig. 1.1 A two phase domain.

1.1.3 Coarse-graining of the differential operator (homogenization)

A classical technique [2, 14] for handling a problem such as (1.1) with a rapidly varying coefficient function a is to construct a functions a_{hom} that varies on the macroscale only (or may even be constant) such that the solution u is in some sense approximated by the solution u_{hom} to

$$\begin{cases} -\nabla \cdot \big(a_{\text{hom}}(x) \cdot \nabla u_{\text{hom}}(x)\big) = 0, & x \in \Omega, \\ \partial_n \, u_{\text{hom}}(x) = f(x), & x \in \Gamma. \end{cases} \tag{1.3}$$

The derivation of an equation such as (1.3) typically relies on fairly strong assumptions on separation of length-scales, rendering this technique problematic in situations involving boundary effects, concentrated loads, multiple or undifferentiated length-scales, *etc.* A common technique for ameliorating these difficulties is to preserve a piece of the fully resolved micro-structure near the boundary, or the concentrated load, and then to "glue" the two models together.

Another common approach is to forego the construction of a coarse-grained continuum model and construct an equation involving a discretized differential operator whose solution in some sense captures the macro-scale behavior of the solution of (1.3), see *e.g.* [17]. The elements of the discretized matrix are typically constructed via local computations on patches of micro-structure.

1.1.4 Coarse-graining of the solution operator

The premise of our work is that it is possible, and often advantageous, to approximate the *solution operator* of (1.1), rather than the differential operator itself. We will demonstrate that with this approach, many of the difficulties encountered in common coarse-graining strategies can be side-stepped entirely. To be precise, we note that mathematically, the solution to (1.1) takes the form

$$u(x) = [K\, f](x) = \int_\Gamma G(x,y)\, f(y)\, ds(y), \qquad x \in \Gamma, \tag{1.4}$$

where G is a kernel function that depends both on the function a, and on the domain Ω. It is known analytically only in the most trivial cases (such as a being constant, and Ω being a square or a circle). However, it turns out that the solution operator can be constructed numerically relatively cheaply, and that it admits very data-sparse representations.

Roughly speaking, our proposal is that instead of seeking an approximation of the form (1.3) of (1.1), it is often advantageous to seek an approximation of the form

$$u_{\text{hom}}(x) = [K_{\text{hom}}\, f](x) = \int_\Gamma G_{\text{hom}}(x,y)\, f(y)\, ds(y), \qquad x \in \Gamma.$$

of (1.4). The purpose of the manuscript is to demonstrate the basic viability and desirability of this approach. Specifically, we seek to:

1. Demonstrate via numerical examples that the solution operators can to high precision be approximated by "data-sparse" representations.
2. Illustrate a framework in which highly accurate reduced models can be constructed even for situations involving boundary effects, and concentrated loads.
3. Demonstrate that the reduced models can in many instances be computed inexpensively from statistical experiments on RVEs.
4. Demonstrate that in situations where the full micro-structure needs to be resolved, there exist highly efficient techniques for doing so, and that the resulting reduced models form natural building blocks in computational models.

Remark 1. In this paper, we focus on problems with no boundary load, such as (1.1). However, the ideas set out can equally well be applied to problems such as

$$\begin{cases} -\nabla \cdot \big(a(x) \cdot \nabla u(x)\big) = h(x), & x \in \Omega, \\ u_n(x) = f(x), & x \in \Gamma. \end{cases} \tag{1.5}$$

The mathematical solution operator then contains two terms, one corresponding to each of the two data functions f and g,

$$u(x) = \int_\Gamma G(x,y)\, f(y)\, ds(y) + \int_\Omega K(x,y)\, h(y)\, dA(y), \qquad x \in \Omega. \tag{1.6}$$

The second term in (1.6) is compressible in a manner very similar to that of the first.

Remark 2. A reason why approximation of the solution operator may prove advantageous compared to approximating the differential operator is hinted at by the spectral properties of the problem. For a bounded domain, an elliptic operator A such as the one defined by equation (1.1) or (1.2) typically has a discrete spectrum $(\lambda_n)_{n=1}^\infty$, where $\lambda_n \to \infty$, and where eigenfunctions get more oscillatory the larger λ_n is. In up-scaling A, we seek to construct an operator A_{hom} whose low eigenvalues and eigenfunctions approximate those of A. Measuring success is tricky, however, since the operator $A - A_{\text{hom}}$ is in a certain sense dominated by the high eigenvalues. One way of handling this is to consider multi-scale representations of the operators, see, *e.g.*, [1, 9, 16, 18, 19]. Another way is to try to approximate the *inverse* of the operator. We observe that A^{-1} is typically compact, and its dominant eigenmodes are precisely those that we seek to capture. Roughly speaking, we advocate the numerical construction of a finite dimensional operator T such that $||A^{-1} - T||$ is small.

Remark 3. Our goal with this paper is not to set up a mathematical analysis of the properties of kernels such as the function G in (1.4). However, to give a sense of the type of questions that arise, let us consider a situation where the function a in (1.1) represents a micro-structure with a characteristic length-scale λ. We then let d denote a cut-off parameter that separates the *near-field* from the *far-field*, say $d = 5\,\lambda$, and set

$$G_{\text{near}}(x,y) = \begin{cases} G(x,y), & |x-y| \le d, \\ 0, & |x-y| > d, \end{cases} \qquad G_{\text{far}}(x,y) = \begin{cases} 0, & |x-y| \le d, \\ G(x,y), & |x-y| > d, \end{cases}$$

and

$$u_{\text{near}}(x) = \int_\Gamma G_{\text{near}}(x,y)\, f(y)\, ds(y), \qquad u_{\text{far}}(x) = \int_\Gamma G_{\text{far}}(x,y)\, f(y)\, ds(y).$$

The function $y \mapsto G_{\text{near}}(x,y)$ depends strongly on the local micro-structure near x, and cannot easily be compressed. This part of the operator must be resolved sufficiently finely to fully represent the micro-structure. However, this is a local interaction, and u_{near} can be evaluated cheaply once G_{near} has been determined. In contrast, G_{far} is compressible. If Γ_1 and Γ_2 are two non-touching pieces of the boundary, then the integral operator

$$[T_{\Gamma_1 \leftarrow \Gamma_2}\sigma](x) = \int_{\Gamma_2} G_{\text{far}}(x,y)\,\sigma(y)\,ds(y), \qquad x \in \Gamma_1,$$

is not only compact, but its singular values typically decay exponentially fast, with the rate of decay depending on the sizes of Γ_1 and Γ_2, and on the distance between them. More careful analysis of these issues in an appropriate multi-scale framework can be found in [22].

1.2 Data-sparse matrices

A ubiquitous task in computational science is to rapidly perform linear algebraic operations involving very large matrices. Such operations typically exploit special *structure* in the matrix since the costs for methods capable of handling general matrices tend to scale prohibitively fast with matrix size: For a general $N \times N$ matrix, it costs $O(N^2)$ operations to perform a matrix-vector multiplication, $O(N^3)$ operations to perform Gaussian elimination or to invert the matrix, *etc.* A well-known form of structure in a matrix is sparsity. When at most a few entries in each row of the matrix are non-zero (as is the case, *e.g.*, for matrices arising upon the discretization of differential equations, or representing the link structure of the World Wide Web) matrix-vector multiplications can be performed in $O(N)$ operations instead of $O(N^2)$. The description *data-sparse* applies to a matrix that may be dense, but that shares the key characteristic of a sparse matrix that some linear algebraic operations, typically the matrix-vector multiplication, can to high precision be executed in fewer than $O(N^2)$ operations (often in close to linear time).

There are many different types of data-sparse representations of a matrix. In this paper, we will utilize techniques for so called *Hierarchically Semi-Separable* (HSS) matrices [11, 13, 32], which arise upon the discretization of many of the integral operators of mathematical physics, in signal processing, in algorithms for inverting certain finite element matrices, and in many other applications, see *e.g.* [12, 29, 32]. An HSS matrix is a dense matrix whose off-diagonal blocks are rank-deficient in a certain sense. Without going into details, we for now simply note that an HSS matrix A can be expressed via a recursive formula in L levels,

$$\mathsf{A}^{(\ell)} = \mathsf{U}^{(\ell)}\,\mathsf{A}^{(\ell-1)}\,\mathsf{V}^{(\ell)} + \mathsf{B}^{(\ell)}, \qquad \ell = 2,\,3,\,\ldots,L, \tag{1.7}$$

where $\mathsf{A} = \mathsf{A}^{(L)}$, and the sequence $\mathsf{A}^{(L)}$, $\mathsf{A}^{(L-1)}$, $\ldots$, $\mathsf{A}^{(1)}$ consists of matrices that are successively smaller (typically, $\mathsf{A}^{(\ell-1)}$ is roughly half the size of $\mathsf{A}^{(\ell)}$). In (1.7), the matrices $\mathsf{U}^{(\ell)}$, $\mathsf{V}^{(\ell)}$ and $\mathsf{B}^{(\ell)}$ are all block-diagonal, so the formula directly leads to a fast technique for evaluating a matrix-vector product. The HSS property is similar to many other data-sparse representations in that it exploits rank-deficiencies in off-diagonal blocks to allow matrix-vector products to be evaluated rapidly; the Fast Multipole Method [23, 24], Barnes-Hut [3], and panel clustering [25] are all similar in this regard. The HSS property is different from these other formats in that it also allows the rapid computation of a matrix inverse, of an LU factorization, *etc*, [10, 11, 15, 28, 33]. The ability to perform algebraic operations other than the matrix-vector multiplication is also characteristic of the $\mathcal{H}$-matrix format of Hackbusch [27].

Remark 4. There currently is little consistency in terminology when it comes to "data-sparse" matrices. The property that we refer to as the "HSS" property has appeared under different names in, *e.g.*, [28, 30, 31, 33]. It is closely related to the "$\mathcal{H}^2$-matrix" format [5, 6, 7, 26] which is more restrictive than the $\mathcal{H}$-matrix format, and often admits $O(N)$ algorithms.

Remark 5. This remark describes in which sense the off-diagonal blocks of a matrix that is compressible in the HSS-sense have low rank; it can safely be by-passed as the material here is referenced only briefly in Section 1.3.3. Let A denote an $N \times N$ HSS matrix A. Let I denote an index vector

$$I = [n+1,\, n+2,\, \ldots,\, n+m],$$

where n and m are positive integers such that $n+m \leq N$. Then we define the *HSS row block* R_I as the $m \times N$ matrix

$$\mathsf{R}_I = \left[\begin{array}{ccc|c|ccc} a_{n+1,1} & a_{n+1,2} & \cdots & a_{n+1,n} & 0\ 0\ \cdots\ 0 & a_{n+1,n+m+1} & a_{n+1,n+m+2} & \cdots & a_{n+1,N} \\ a_{n+2,1} & a_{n+2,2} & \cdots & a_{n+2,n} & 0\ 0\ \cdots\ 0 & a_{n+2,n+m+1} & a_{n+2,n+m+2} & \cdots & a_{n+2,N} \\ \vdots & \vdots & & \vdots & \vdots\ \vdots\ \ \vdots & \vdots & \vdots & & \vdots \\ a_{n+m,1} & a_{n+m,2} & \cdots & a_{n+m,n} & 0\ 0\ \cdots\ 0 & a_{n+m,n+m+1} & a_{n+m,n+m+2} & \cdots & a_{n+m,N} \end{array}\right]$$

In other words, R_I is an $m \times N$ sub-matrix of A corresponding to the rows marked by the index vector I, but with the diagonal block corresponding to I replaced by a zero matrix. The *HSS column block* C_I is analogously defined as the $N \times m$ matrix consisting of m columns of A with the diagonal block excised. The principal criterion for a matrix A to be compressible in the HSS sense is that its HSS blocks should have numerically low rank.

1.3 Case study: A discrete Laplace equation on a square

In this section, we illustrate how the coarse-graining techniques outlined in Section 1.1.4 can be applied to a discrete equation closely related to (1.1). This discrete equation can be viewed either as the result of discretizing (1.1) via a finite difference method, or as an equation that in its own right models, for instance, electro-statics on a discrete grid.

1.3.1 Problem formulation

Given a positive integer N_{side}, we let Ω denote the $N_{\text{side}} \times N_{\text{side}}$ square subset of $\mathbb{Z}^2$ given by

$$\Omega = \{m = (m_1,\, m_2) \in \mathbb{Z}^2 :\ 1 \leq m_1 \leq N_{\text{side}} \text{ and } 1 \leq m_2 \leq N_{\text{side}}\}. \tag{1.8}$$

Figure 1.2(a) illustrates the definition. For a node $m \in \Omega$, we let $\mathbb{B}_m$ denote a list of of all nodes in Ω that directly connect to m. For instance, an interior node such as the node m shown in Figure 1.2(b) would have the neighbor list

$$\mathbb{B}_m = \{m_{\text{s}},\, m_{\text{e}},\, m_{\text{n}},\, m_{\text{w}}\},$$

while a node on a "western" boundary like n in Figure 1.2(c) would have the neighbor list

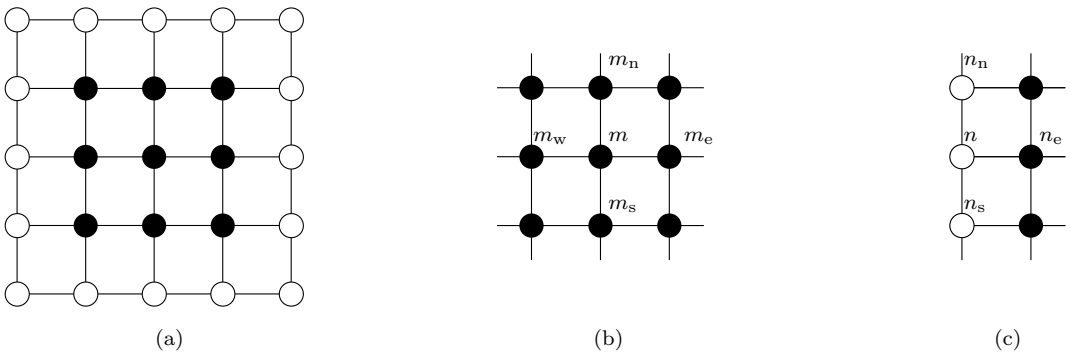


Fig. 1.2 Geometry of the lattice problem in Section 1.3.1. (a) The full lattice for $N_{\text{side}} = 5$. The boundary nodes in Ω_{b} are white and the interior nodes in Ω_{i} are black. (b) The four neighbors of an interior node m. (c) The three neighbors of a boundary node n.

$$\mathbb{B}_n = \{n_{\mathrm{s}},\, n_{\mathrm{e}},\, n_{\mathrm{n}}\}.$$

For each pair $\{m,\, n\}$ of connected nodes, we let $\alpha_{m,n}$ denote a parameter indicating the *conductivity* of the link. For a function $u = u(m)$ where $m \in \Omega$, the *discrete Laplace operator* is then defined via

$$[\mathsf{A}\, u](m) = \sum_{n \in \mathbb{B}_m} \alpha_{m,n} \big[u(m) - u(n)\big]. \tag{1.9}$$

Example: For the case where $\alpha_{m,n} = 1$ for all connected nodes, we retrieve the standard five-point stencil associated with discretization of the Laplace operator. For instance, with column-wise ordering of the nodes in the lattice shown in Figure 1.2(a), we obtain the 25×25 matrix

$$\mathsf{A} = \begin{bmatrix} \mathsf{C} & -\mathsf{I} & 0 & 0 & 0 \\ -\mathsf{I} & \mathsf{D} & -\mathsf{I} & 0 & 0 \\ 0 & -\mathsf{I} & \mathsf{D} & -\mathsf{I} & 0 \\ 0 & 0 & -\mathsf{I} & \mathsf{D} & -\mathsf{I} \\ 0 & 0 & 0 & -\mathsf{I} & \mathsf{C} \end{bmatrix}, \quad \text{where} \quad \mathsf{C} = \begin{bmatrix} 2 & -1 & 0 & 0 & 0 \\ -1 & 3 & -1 & 0 & 0 \\ 0 & -1 & 3 & -1 & 0 \\ 0 & 0 & -1 & 3 & -1 \\ 0 & 0 & 0 & -1 & 2 \end{bmatrix}, \quad \text{where} \quad \mathsf{D} = \begin{bmatrix} 3 & -1 & 0 & 0 & 0 \\ -1 & 4 & -1 & 0 & 0 \\ 0 & -1 & 4 & -1 & 0 \\ 0 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & -1 & 3 \end{bmatrix}, \tag{1.10}$$

and where I is the 5×5 identity matrix.

We let Ω_{b} denote the *boundary nodes* and we let Ω_{i} denote the *interior nodes* (*cf.* Figure 1.2(a)). Partitioning the matrix A accordingly, the discrete analog of (1.1) becomes

$$\begin{bmatrix} \mathsf{A}_{\mathrm{b,b}} & \mathsf{A}_{\mathrm{b,i}} \\ \mathsf{A}_{\mathrm{i,b}} & \mathsf{A}_{\mathrm{i,i}} \end{bmatrix} \begin{bmatrix} u_{\mathrm{b}} \\ u_{\mathrm{i}} \end{bmatrix} = \begin{bmatrix} f_{\mathrm{b}} \\ 0 \end{bmatrix}. \tag{1.11}$$

Solving for the boundary values of the potential, u_{b}, we find that[1]

$$u_{\mathrm{b}} = \big(\mathsf{A}_{\mathrm{b,b}} - \mathsf{A}_{\mathrm{b,i}}\, \mathsf{A}_{\mathrm{i,i}}^{-1}\, \mathsf{A}_{\mathrm{i,b}}\big)^{-1} f_{\mathrm{b}}.$$

In consequence, the discrete analog of the solution operator (in this case a discrete analog of the Neumann-to-Dirichlet operator) is

$$\mathsf{T} = \big(\mathsf{A}_{\mathrm{b,b}} - \mathsf{A}_{\mathrm{b,i}}\, \mathsf{A}_{\mathrm{i,i}}^{-1}\, \mathsf{A}_{\mathrm{i,b}}\big)^{-1}. \tag{1.12}$$

The operator T defined by (1.12) is dense, but turns out to be *data-sparse* in the sense described in Section 1.2. We will in this section substantiate this claim via numerical examples, and also outline strategies for rapidly constructing such an operator in different environments.

1.3.2 Model problems

The compressibility of the solution operator T defined by (1.12) was investigated in the following five model environments:

Case A: Constant conductivities. In this model, all conductivities are identically one,

$$\alpha_{m,n} = 1 \qquad \text{for each connected pair } \{m,\, n\}. \tag{1.13}$$

For $N_{\mathrm{side}} = 5$, the resulting matrix A is the one given as an example in (1.10). Since in this case the matrix A can be viewed as a discretization of the Laplace operator $-\Delta$ on a square, the solution operator T can be viewed as a discrete analog of the standard Neumann-to-Dirichlet operator associated with Laplace's equation.

[1] Strictly speaking, the matrix $\mathsf{A}_{\mathrm{b,b}} - \mathsf{A}_{\mathrm{b,i}}\, \mathsf{A}_{\mathrm{i,i}}^{-1}\, \mathsf{A}_{\mathrm{i,b}}$ has a one-dimensional null-space formed by the constant functions and is not invertible. This is easily dealt with by a regularization that restricts attention to functions summing to zero. In what follows, such regularization will be employed where appropriate without further mention.

Case B: Smooth periodic conductivities. This case is a discrete analog of the equation

$$-\nabla \cdot \big(b(x)\nabla u(x)\big) = f(x), \qquad x \in [0,\, 1]^2, \tag{1.14}$$

where b is a periodic function defined by

$$b(x) = 1 - 0.9\left(\cos(\pi\, N_{\text{cells}}\, x_1)\right)^2 \left(\cos(\pi\, N_{\text{cells}}\, x_2)\right)^2, \qquad x = (x_1,\, x_2) \in [0,\, 1]^2. \tag{1.15}$$

In other words, (1.14) models a medium whose conductivity repeats periodically across $N_{\text{cells}} \times N_{\text{cells}}$ cells in the square $[0,\, 1]^2$. Figure 1.4(a) illustrates the function b for $N_{\text{cells}} = 4$. A discrete analog of (1.14) is now obtained by setting

$$\alpha_{m,n} = b\left(\frac{m+n-2}{2\,(N_{\text{side}}-1)}\right) \qquad \text{for each connected pair } \{m,\, n\}.$$

In our experiments, we chose N_{cell} so that 25 nodes were used to resolve each period, $N_{\text{cell}} = (N_{\text{side}} - 1)/25$ (for clarity, Figure 1.4 shows a solution with only 15 nodes per period). In this case, the solutions are typically oscillatory on the boundary, *cf.* Figure 1.4(b). This is a basic two-scale problem that should be amenable to traditional homogenization techniques provided there is a sufficient separation of length-scales.

Case C: Random conductivities. The conductivities $\alpha_{m,n}$ are for each connected pair of nodes $\{m,\, n\}$ drawn independently from a uniform probability distribution on $[1,\, 2]$. In this case, there is no local regularity, but we would expect traditional homogenization to give accurate results whenever the length-scales are sufficiently separated.

Case D: Sparsely distributed missing bars. In this model, all bars are assigned conductivity 1 (as in Case A), but then a small percentage p of bars are completely removed (in the examples reported, $p = 4\%$). In other words,

$$\alpha_{m,n} = \begin{cases} 1, & \text{with probability } 1-p \quad \text{if } \{m,\, n\} \text{ is a connected pair,} \\ 0, & \text{with probability } p \qquad \text{if } \{m,\, n\} \text{ is a connected pair,} \\ 0, & \qquad\qquad\qquad\quad \text{if } \{m,\, n\} \text{ is not a connected pair.} \end{cases}$$

As in Case C, there is no local regularity, but we would expect traditional homogenization to give accurate results whenever the length-scales are sufficiently separated.

Case E: A lattice with two long cracks. This model is similar to Case D in that a small number of links have been cut, and all the remaining ones have unit conductivity. However, we organized the cut links into two long cracks running through the lattice. Figure 1.5(a) illustrates for a case where $N_{\text{side}} = 50$. In larger lattices, the cracks have the same proportions, but the gap between the two cracks is kept constant at four links. In this case, solutions may exhibit major discontinuities. Figure 1.5(b) illustrate the electric field resulting from placing oppositely signed unit sources at the locations marked *source* and *sink* in Figure 1.5(a). We would expect analytic derivation of a simplified model to be very hard work in a situation such as this.

1.3.3 Compressibility of the solution operator

While the operator T defined by (1.12) is dense, it is in many situations of interest *data-sparse* in the sense described in Section 1.2. To illustrate this point, we computed the matrix T by brute force for several different lattices, compressed it into the HSS format to ten digits of accuracy (we enforced that local truncation errors be less than 10^{-10}), and looked at how much memory was required to store the result. Tables 1.1 and 1.2 show our findings for each of the five different models described in Section 1.3.2, and for differently sized lattices. To provide more detail, Table 1.3 reports the

average ranks of the so called "HSS blocks" (as defined in Remark 5) of a $6\,396 \times 6\,396$ matrix T associated with a $1\,600 \times 1\,600$ square domain for each of the five examples.

An interesting aspect of the reported data is that the matrix T associated with the classical five-point stencil (represented by Case A) is highly compressible. To store it to ten digits of accuracy, less than 100 floating point numbers are required for each degree of freedom (see Table 1.2). This fact has been exploited in a series of recent papers, including [12, 22, 29]. What is perhaps more remarkable is that the compressibility property is extremely robust to small changes in the micro-structure. As the tables 1.1, 1.2, and 1.3 show, there is almost no discernible difference in compressibility between the five models considered.

Once the compressed solution operator has been computed, it can be applied to a vector more or less instantaneously. For our simple implementation, we found the following for the time t_{solve} (in seconds) required for a single solve:

N_{side}	200	400	800	1600	3200
t_{solve} (sec)	4.4e-3	8.7e-3	1.8e-2	3.4e-2	7.1e-2

These numbers refer to a reduced model that is precise to within ten digits, and we would like to emphasize that the largest example reported, which requires 0.07 seconds for one solve, involves a problem whose micro-structure was originally resolved using $3\,200 \times 3\,200 \approx 10^7$ nodes.

The results reported in tables 1.1, 1.2, and 1.3 indicate that reduced models that are precise to within ten digits of accuracy in principle exist, even in the presence of the following complications:

- Solutions that are oscillatory on the boundary, even when the period of the oscillation is not very much smaller than the size of the domain (as in Case B).
- Solutions that are highly irregular on the boundary (as in Cases C, D, and E).
- Boundary loads that exhibit no smoothness. (We observe that the solution operator is constructed under no assumption on smoothness of the boundary data.)
- Solutions that involving significant discontinuities (as shown in Figure 1.5(b)).

In Sections 1.3.4, 1.3.5, and 1.3.6, we will describe practical techniques for inexpensively computing such reduced models.

1.3.4 Techniques for computing the solution operator that fully resolve the micro-structure

Given a realization of a lattice model, the operator T defined by (1.12) can of course be computed with brute force. While Gaussian elimination has an $O(N_{\text{side}}^6)$ asymptotic cost that quickly becomes prohibitive, substantially more efficient techniques exist. Appendix A describes a variation of the classical *nested dissection* method which in the present environment requires $O(N_{\text{side}}^3)$ floating point operations (flops) and $O(N_{\text{side}}^2)$ memory. This technique is exact up to rounding errors, and is very easy to implement. It was used to calculate the numbers reported in Section 1.3.3 and is sufficiently fast that the solution operator associated with an 800×800 lattice can be determined in 40 seconds via a Matlab implementation running on a standard desktop PC.

More recently, techniques have been developed that compute an operator such as T in $O(N_{\text{side}}^2)$ time (or possibly $O(N_{\text{side}}^2 (\log N_{\text{side}})^\kappa)$ for a small integer κ), which is optimal since there are $O(N_{\text{side}}^2)$ links in the lattice [12, 22, 29]. These techniques are highly efficient, and enable the brute force calculation of a reduced model in many important environments in both two and three dimensions.

1.3.5 Techniques accelerated by collecting statistics from a representative volume element

In situations where there is a good separation of length-scales, variations of classical homogenization techniques can be used to dramatically accelerate the computation of a compressed boundary operator. To illustrate, let us investigate Case C in Section 1.3.2 (the case of random conductivities, drawn uniformly from the interval [1, 2]). The most basic "homogenized equation" is in this case a lattice with all links have the same conductivity. Through experiments on an RVE, we determined that this conductivity should be

$$c_3 = 1.4718\cdots$$

We let T_{hom} denote the solution operator (*i.e.* the lattice Neumann-to-Dirichlet operator) for the homogenized lattice. We measured the discrepancy between the homogenized operator T_{hom}, and the operator associated with the original lattice T, using the measures:

$$E_{\text{N2D}} = \frac{||\mathsf{T}_{\text{hom}} - \mathsf{T}||}{||\mathsf{T}||}, \qquad \text{and} \qquad E_{\text{D2N}} = \frac{||\mathsf{T}_{\text{hom}}^{-1} - \mathsf{T}^{-1}||}{||\mathsf{T}^{-1}||}. \tag{1.16}$$

Table 1.4 gives the results for a particular realization of a 50×50 lattice. In addition to the discrepancies measured in operator norm, the table also provides the errors

$$E_{\text{smooth}} = \frac{||\big(\mathsf{T}_{\text{hom}} - \mathsf{T}\big)\, f_{\text{smooth}}||}{||\mathsf{T}\, f_{\text{smooth}}||}, \qquad \text{and} \qquad E_{\text{rough}} = \frac{||\big(\mathsf{T}_{\text{hom}} - \mathsf{T}\big)\, f_{\text{rough}}||}{||\mathsf{T}\, f_{\text{rough}}||}, \tag{1.17}$$

associated with two particular Neumann data vectors f_{smooth} and f_{rough}. The solutions associated with these data vectors are shown in Figure 1.6. These examples show that as one would expect, the homogenized equation provides quite high accuracy for a smooth solution, and very poor accuracy for a rough one.

We next repeated all experiments for Case D (as defined in Section 1.3.2). In this case, numerical experiments indicated that the homogenized conductivity is

$$c_4 = 1 - \frac{1}{2}p + O(p^2).$$

Table 1.5 shows the errors associated with a realization of "Case D" on a 50×50 grid, with $p = 0.04$, and $c_4 = 0.98$.

Remark 6 (Computational cost). The solution operator T_{hom} associated with a constant coefficient lattice can be computed in time proportional to $O(N_{\text{side}})$ (in other words, in time proportional to the number of nodes on the boundary). This means that very large lattices can be handled rapidly. It was demonstrated in [21] that the solution operator associated with a lattice with 10^{10} nodes can be computed in less than two minutes on a standard desktop PC. (Observe that only the $4 \cdot 10^5$ nodes on the boundary actually need to enter the calculation.)

1.3.6 Fusing a homogenized model to a locally fully resolved region

In the environments under consideration here, domains are loaded only on the border. This of course raises the possibility of improving the accuracy in the homogenized model by preserving the actual micro-structure in a thin strip along the boundary, and use the homogenized equations only in the interior. In the frame-work proposed here, where the simplified model consists of a solution operator rather than a differential operator (or in the present case, difference operator), it is extra ordinarily simple to do so.

To illustrate, suppose that we are given a realization of an $N_{\text{side}} \times N_{\text{side}}$ lattice with heterogeneous conductivities. We fix a parameter b that indicates how broad of a band of cells we preserve, and then

replace all bars that are more than b cells away from the boundary by bars with the homogenized conductivity, as illustrated in Figure 1.3(a). Then use the techniques of Section 1.3.5 to compute the Neumann-to-Dirichlet operator for the constant coefficient lattice of size $(N_{\text{side}} - 2b) \times (N_{\text{side}} - 2b)$ in the center. As observed in Remark 6, the cost is only $O(N_{\text{side}})$, and the new reduced model involves only $O(N_{\text{side}})$ degrees of freedom. As Tables 1.4 and 1.5 demonstrate, for our model problems ("Case C" and "Case D") keeping only five layers of the original lattice leads to a reduced model that is accurate to three or four digits.

Remark 7 (Accuracy of Neumann vs. Dirichlet problems). Tables 1.4 and 1.5 show that while "full" homogenization (*i.e.* all conductivities are replaced by a homogenized value) is less accurate for a Dirichlet problem than it is for a Neumann problem, the accuracy of Dirichlet problems improve dramatically upon the introduction of even a very thin boundary layer. This is as one would expect since the Dirichlet-to-Neumann operator is dominated by short range interactions.

1.4 Case study: Two-phase media

In this section, we briefly investigate the compressibility of the Neumann-to-Dirichlet operator for a two-phase material modeled by equation (1.2). The two geometries we consider are shown in Figure 1.7, with the conductivity of the inclusions set to zero. In this case, the operator under consideration is a boundary integral operator T supported on the square outer boundary. Using techniques described in Remark 8, we constructed an 1144×1144 matrix T that approximated T. With this number of nodes, any Neumann data generated by point sources up to a distance of 0.5% of the side length of the square can be resolved to eight digits of accuracy. We compressed the matrix T into the HSS format described in Section 1.2 to a relative precision of 10^{-10}. The resulting data required 1.19KB of memory to store for the geometry shown in Figure 1.7(a), and 1.22KB of memory for the geometry shown in Figure 1.7(b). This corresponds to about 135 words of storage per row in the matrix. The HSS-ranks (as defined in Remark 5) are reported in Table 1.6. We make three observations:

- A compressed version of the boundary operator can in this case be stored using about the same amount of memory (100 words per degree of freedom) as the operators associated with the discrete problems described in Section 1.3.
- The two geometries shown in Figure 1.7 require about the same amount of memory. This is note-worthy since the one labeled (b) corresponds to an almost singular geometry in which the

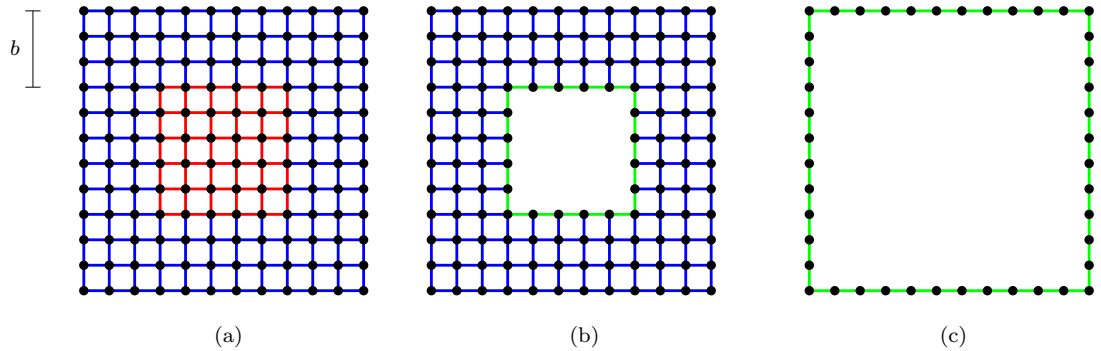


Fig. 1.3 Construction of a highly accurate reduced model by fusing a homogenized region with a region in which the micro-structure is fully resolved. (a) The blue links are within distance b of the boundary, and maintain their original conductivity. The red links are all assigned the "homogenized" conductivity. (b) All red links are eliminated from the model. This requires the construction of the solution operator for a constant coefficient lattice at cost $O(N_{\text{side}})$ (see Remark 6). (c) The few remaining links are eliminated to construct a highly approximate approximation to the solution operator.

domain is very close to being split in two halves. Standard assumptions used when homogenizing an elliptic differential operator are violated in this case.
- In Table 1.6, the ranks of HSS-blocks of size 143 are *larger* than those of HSS-blocks of size 286. We speculate that this unusual situation can be traced to the fact that the larger blocks are larger than the inclusions, and largely do not "see" the heterogeneities.

Remark 8 (Details of computation). To derive our approximation to the Neumann-to-Dirichlet operator, we recast equation (1.2) as a BIE defined on the joint boundary $\Gamma \cup \Gamma_{\text{int}}$. We used a single layer representation on the outer boundary Γ, and a double layer representation on the interior boundary Γ_{int}. In other words, we sought a solution of the form

$$u(x) = \int_{\Gamma} \log|x-y|\,\sigma(y)\,ds(y) + \int_{\Gamma_{\text{int}}} \frac{n(y)\cdot(x-y)}{|x-y|^2}\,\tau(y)\,ds(y), \tag{1.18}$$

where $n(y)$ is a unit normal vector. The resulting BIE was discretized using a Nyström method combined with trapezoidal quadrature on the interior holes, and a Gaussian quadrature on the exterior boundary supported on 44 panels with 26 nodes each. The quadrature rule was locally modified as described in [8] to maintain eight digit accuracy in the presence of corners. This resulted in a large linear system from which all degrees of freedom associated with internal nodes (those associated with the density τ in (1.18)) where eliminated. The final approximation to the Neumann-to-Dirichlet operator was then formed as the inverse of the resulting Schur complement.

1.5 Generalizations

This report focused on problems modeled by simple Laplace-type problems in two dimensions involving no body loads. However, the techniques can be extended to much more general environments:

Other boundary conditions: While we focused on problems with Neumann boundary conditions, the extension to Dirichlet or mixed boundary conditions is trivial.

Other elliptic equations: The methods described extend readily to other elliptic equations whose kernels are non-oscillary such as Stokes, elasticity, Yukawa, *etc.* The extension to wave problems modeled by Helmholtz equation, or the time-harmonic version of Maxwell, is more complicated for two reasons: (1) The presence of resonances (both true ones corresponding to the actual physics, and artificial ones present in the mathematical model only) must be dealt with. This can be done, but requires careful attention. (2) As the wave-number increases, the compressibility of the solution operator deteriorates, and eventually renders the proposed approach wholly unaffordable.

Body loads: The extension to problems involving body loads is in principle straight-forward (see Remark 1). However, the compressed solution operator becomes more expensive to store.

Problems in three dimensions: In principle, the methodology proposed extends straightforwardly to problems in three dimensions. However, the construction of the solution operator does become more expensive, and the method might be best suited for environments where a precomputation is possible, or where the construction of the solution operator can be accelerated via the use of homogenized models in parts of the domain (as illustrated in Section 1.3.6). Moreover, for problems in three dimensions involving body loads, memory requirements may become prohibitive.

1.6 Conclusions

The modest goal of this report is to draw attention to recent developments in numerical analysis that could be very useful in modeling heterogeneous media. Specifically, it has become possible to inexpensively compute an approximation to the solution operator associated with many elliptic

PDEs, and to perform various operations involving such solution operators: addition, multiplication, inversion, merging operators for different sub-domains, *etc.* We argue that such solution operators form excellent "reduced models" for many problems that have proven difficult to handle using traditional homogenization techniques.

Constructing reduced models by approximating the solution operator is particularly advantageous in the following environments:

Domains that are loaded on the boundary only: For problems that involve no body load, the solution operator is defined on the boundary only. This reduction in dimensionality means that once it is computed, it can be stored very efficiently, and applied to vectors sufficiently fast that real time simulations become possible. For some problems in this category, the actual construction of the solution operator require a large-scale (but very efficient) computation involving the entire micro-structure, but as shown in Section 1.3.6, the solution operator can sometime be dramatically accelerated by using a homogenized model in the interior of the domain.

Situations where a pre-computation is possible: When the entire micro-structure needs to be resolved (as happens when the problem involves a body load, or a micro-structure not suitable for homogenization methods), the initial construction of the solution operator can become somewhat expensive, in particular for problems in three dimensions. However, once it has been constructed, it can usually be applied to a vector very rapidly. This raises the possibility of pre-computing a library of compressed models which can then be used as building blocks in computational simulations.

Problems in two dimensions (whether involving volume loads or not): Given current trends in algorithmic and hardware development, we predict that for a great many problems in two dimensions, it will soon become entirely affordable to resolve the entire micro-structure, and computationally derive a reduced model of the solution operator. The automatic nature of such a procedure would save much human effort, and would be very robust in the sense that the computed model would be guaranteed to be accurate to whichever tolerance was requested.

Appendix A: Efficient computation of the Neumann-to-Dirichlet operator

In this appendix, we describe an efficient technique for computing the Neumann-to-Dirichlet operator T defined by (1.12). It is a variation of the classical *nested dissection* techniques [20]. Throughout the appendix, Ω is a rectangular lattice, as defined by (1.8), and A is an associated discrete Laplace operator, as defined by (1.9).

To be precise, the technique we will describe does not compute the Neumann-to-Dirichlet operator T, but rather the *Schur complement* S, defined via

$$\mathsf{S} = \mathsf{A}_{\mathrm{b,b}} - \mathsf{A}_{\mathrm{b,i}}\,\mathsf{A}_{\mathrm{i,i}}^{-1}\,\mathsf{A}_{\mathrm{i,b}}. \tag{1.19}$$

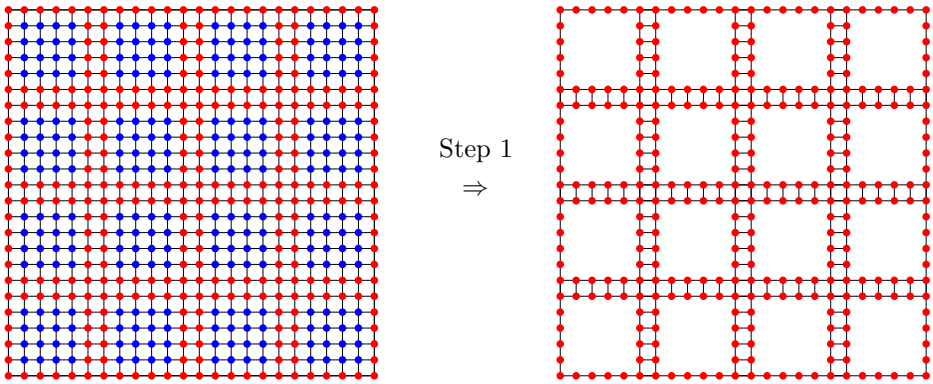
Comparing (1.12) and (1.19), we see $\mathsf{T} = \mathsf{S}^{-1}$.

1.A.1 Outline

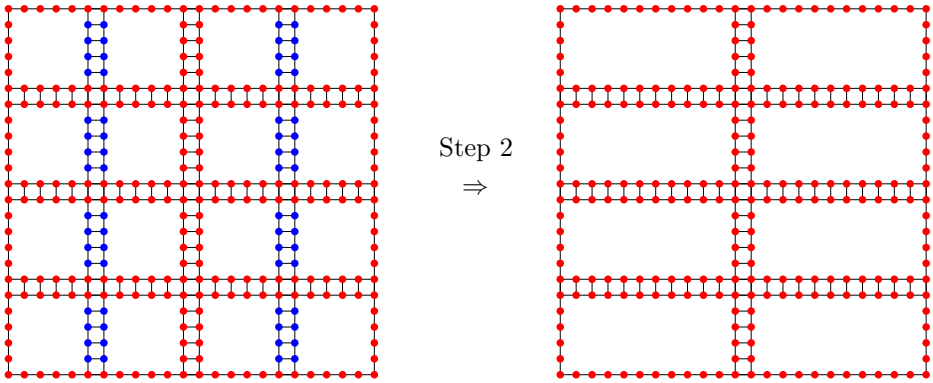
The technique is a divide-and-conquer scheme in which the computational domain Ω is first split into $2^L \times 2^L$ roughly equisized small boxes. The parameter L is chosen so that each of the small boxes is sufficiently small that its Schur complement can be computed by evaluating (1.19) via brute force. (In practice, we found that letting the smallest boxes be of size roughly 50×50, or $L \approx \log_2(N_{\mathrm{side}}/50)$, works well.) Then it turns out to be possible to merge the Schur complements of two small adjacent boxes to form the Schur complement of the larger box; the process is described in Section 1.A.2. The scheme proceeds by continuing the merge process to form the Schur complements of larger and larger boxes until eventually the entire box Ω has been processed. To illustrate, we describe the

process graphically for a 24×24 domain that is originally split into 4×4 boxes, each containing 6×6 nodes.

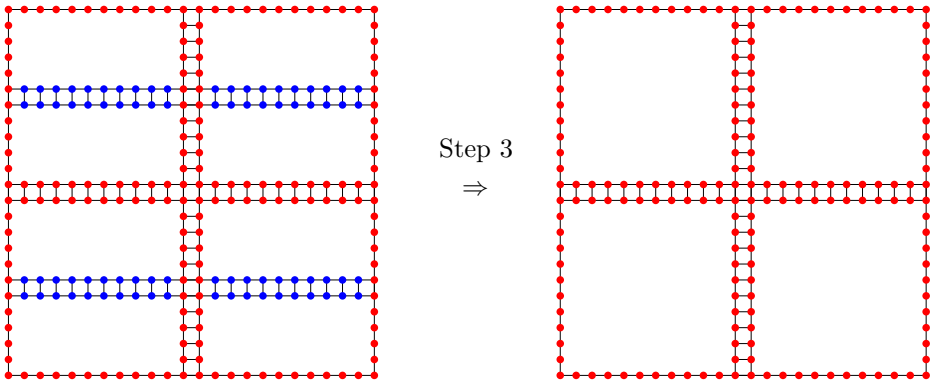
Step 1: Partition the box Ω into 16 small boxes. For each box, identify the internal nodes (marked in blue) and eliminate them using formula (1.19).



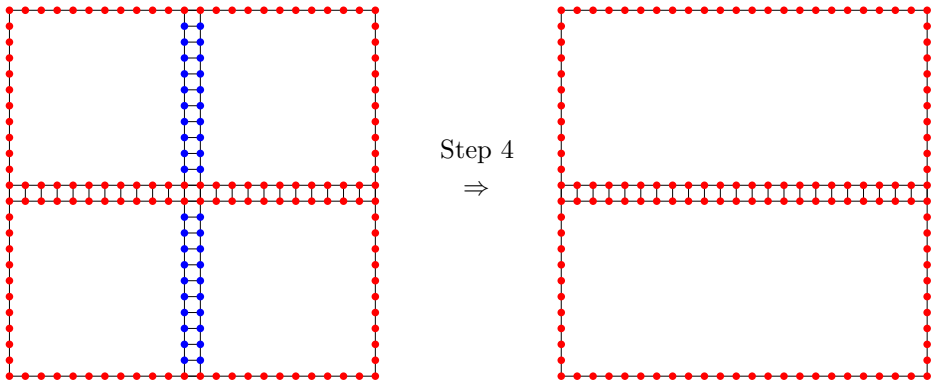
Step 2: Join the small boxes by pairs to form the Schur complements of boxes holding twice the number of nodes via the process to be described in Section 1.A.2. The effect is to eliminate the interior nodes (marked in blue) of the newly formed larger boxes.



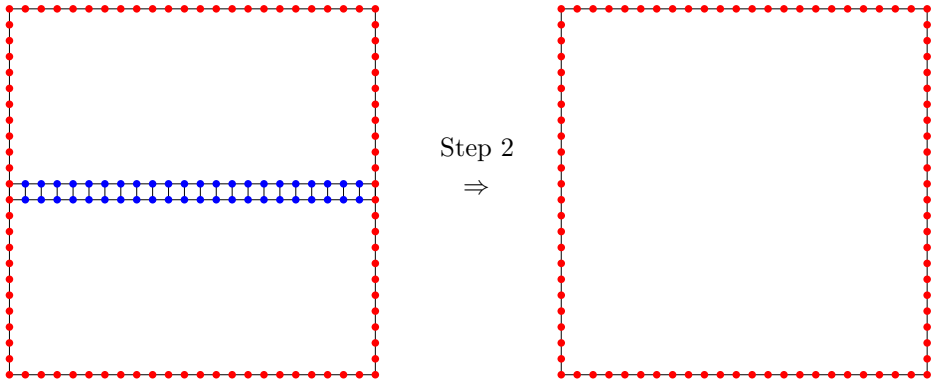
Step 3: Merge the boxes created in Step 2 in pairs, again via the process described in Section 1.A.2.



Step 4: Repeat the merge process once more.

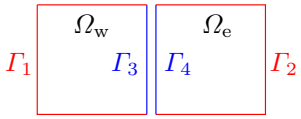


Step 5: Repeat the merge process one final time to obtain the Schur complement associated with the top level box Ω.



1.A.2 Merging of two Schur complements

Suppose that Ω is a box consisting of the two smaller boxes Ω_{w} and Ω_{e} (as in west and east):



Suppose further that we know the corresponding Schur complements S_{w} and S_{e} and seek the Schur complement S of Ω. In effect, we need to remove the "interior" points along the middle lines (marked in blue in the figure).

First partition the nodes in Γ_{w} into the subsets Γ_1 and Γ_3, and partition Γ_{e} into Γ_2 and Γ_4 as shown in the figure. The Schur complements S_{w} and S_{e} are partitioned accordingly,

$$\mathsf{S}_{\mathrm{w}} = \begin{bmatrix} \mathsf{S}_{11} & \mathsf{S}_{13} \\ \mathsf{S}_{31} & \mathsf{S}_{33} \end{bmatrix}, \qquad \text{and} \qquad \mathsf{S}_{\mathrm{e}} = \begin{bmatrix} \mathsf{S}_{22} & \mathsf{S}_{24} \\ \mathsf{S}_{42} & \mathsf{S}_{44} \end{bmatrix}.$$

Since the interior edges are unloaded, the joint equilibrium equation for the two boxes now reads

$$\left[\begin{array}{cc|cc} \mathsf{S}_{11} & \mathsf{A}_{12} & \mathsf{S}_{13} & 0 \\ \mathsf{A}_{21} & \mathsf{S}_{22} & 0 & \mathsf{S}_{24} \\ \hline \mathsf{S}_{31} & 0 & \mathsf{S}_{33} & \mathsf{A}_{34} \\ 0 & \mathsf{S}_{24} & \mathsf{A}_{43} & \mathsf{S}_{44} \end{array}\right] \left[\begin{array}{c} u_1 \\ u_2 \\ u_3 \\ u_4 \end{array}\right] = \left[\begin{array}{c} f_1 \\ f_2 \\ 0 \\ 0 \end{array}\right], \tag{1.20}$$

where A_{ij} are the relevant submatrices of the original discrete Laplacian A. To be precise, with A denoting the global discrete Laplace operator, and with J_i denoting an index vector marking the nodes in Γ_i, we have $\mathsf{A}_{ij} = \mathsf{A}(J_i, J_j)$. We observe that all matrices A_{ij} are very sparse (indeed, A_{12} and A_{21} have only two non-zero elements each). From (1.20), it is clear that the Schur complement of the large box is

$$\mathsf{S} = \begin{bmatrix} \mathsf{S}_{11} & \mathsf{A}_{12} \\ \mathsf{A}_{21} & \mathsf{S}_{22} \end{bmatrix} - \begin{bmatrix} \mathsf{S}_{13} & 0 \\ 0 & \mathsf{S}_{24} \end{bmatrix} \begin{bmatrix} \mathsf{S}_{33} & \mathsf{A}_{34} \\ \mathsf{A}_{43} & \mathsf{S}_{44} \end{bmatrix}^{-1} \begin{bmatrix} \mathsf{S}_{31} & 0 \\ 0 & \mathsf{S}_{42} \end{bmatrix}. \tag{1.21}$$

1.A.3 Accelerations

The scheme as described requires $O(N_{\text{side}}^3)$ floating point operations, and $O(N_{\text{side}}^2)$ storage, just like the original nested dissection scheme. This cost is incurred by the repeated evaluation of the formula (1.21) which involve matrices S_{ij} that are dense. However, as discussed at length in Section 1.3.3, these matrices have internal structure that allows operations such as matrix inversion, and matrix-matrix multiplication, to be evaluated in linear time. Incorporating such accelerated procedures reduces the overall cost (both floating point operations and memory) of the scheme to $O(N_{\text{side}}(\log N_{\text{side}})^\kappa)$. For recent work in this direction, see, *e.g.* [12, 22, 29].

Remark 9. The process described in Section 1.A.1 requires all Schur complements associated with one level to be kept in memory at one time. It is straight-forward to change the order in which the boxes are processed so that at most four Schur complements on each level must be kept in memory. When dense linear algebra is used, either approach requires $O(N_{\text{side}}^2)$ memory, but when data-sparse matrix formats are used, such an ordering reduces the memory requirement from $O(N_{\text{side}}^2)$ to $O(N_{\text{side}}(\log N_{\text{side}})^\kappa)$.

Remark 10. Even without accelerations, the scheme described in Section 1.A.1 can handle moderate size problems quite efficiently. For a rudimentary implementation in Matlab executed on a standard desktop (with an Intel i7 CPU running at 2.67 GHz), the time t required to compute T was:

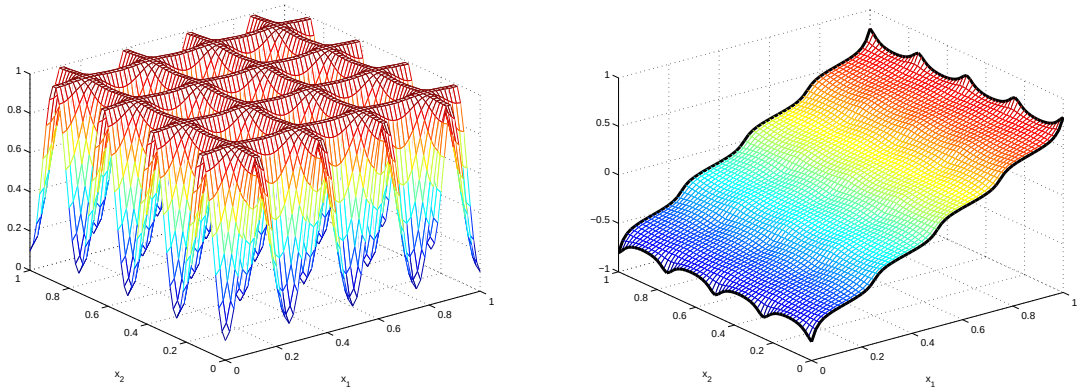
N_{side}	100	200	400	800	1600	3200
t (sec)	2.6e-1	1.2e0	6.4e0	4.5e1	5.0e2	6.7e3

Note that less than a minute is required to process a lattice involving $800^2 = 640\,000$ nodes.

References

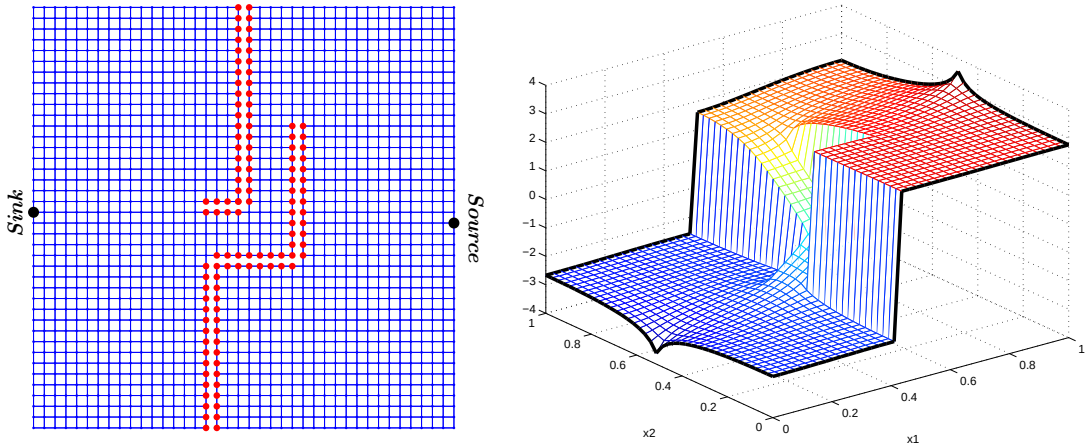
1. Ulf Andersson, Björn Engquist, Gunnar Ledfelt, and Olof Runborg, *A contribution to wavelet-based subgrid modeling*, Appl. Comput. Harmon. Anal. **7** (1999), no. 2, 151–164. MR MR1711012 (2000f:65130)
2. N. Bakhvalov and G. Panasenko, *Homogenisation: averaging processes in periodic media*, Kluwer Academic Publishers Group, Dordrecht, 1989, Mathematical problems in the mechanics of composite materials, Translated from the Russian by D. Leĭtes. MR 92d:73002
3. J. Barnes and P. Hut, *A hierarchical $o(n \log n)$ force-calculation algorithm*, Nature **324** (1986), no. 4.
4. G. Beylkin, R. Coifman, and V. Rokhlin, *Wavelets in numerical analysis*, Wavelets and their applications, Jones and Bartlett, Boston, MA, 1992, pp. 181–210. MR 93j:65215
5. S. Börm, *$\mathcal{H}^2$-matrix arithmetics in linear complexity*, Computing **77** (2006), no. 1, 1–28. MR MR2207953 (2006k:65111)
6. ———, *Approximation of solution operators of elliptic partial differential equations by $\mathcal{H}$-and $\mathcal{H}^2$–matrices*, Tech. Report 85/2007, Max Planck Institute, 2007.

7. ———, *Construction of data-sparse $\mathcal{H}^2$-matrices by hierarchical compression*, Tech. Report 92/2007, Max Planck Institute, 2007.
8. J. Bremer and V. Rokhlin, *Efficient discretization of Laplace boundary integral equations on polygonal domains*, J. Comput. Phys. **229** (2010), no. 7, 2507–2525. MR MR2586199
9. M. E. Brewster and G. Beylkin, *A multiresolution strategy for numerical homogenization*, Appl. Comput. Harmon. Anal. **2** (1995), no. 4, 327–349. MR 96h:65156
10. S. Chandrasekaran and M. Gu, *Fast and stable algorithms for banded plus semiseparable systems of linear equations*, SIAM J. Matrix Anal. Appl. **25** (2003), no. 2, 373–384 (electronic). MR MR2047424 (2005f:65039)
11. S. Chandrasekaran, M. Gu, X.S. Li, and J. Xia, *Some fast algorithms for hierarchically semiseparable matrices*, Tech. Report 08-24, UCLA/CAM, 2008.
12. ———, *Superfast multifrontal method for structured linear systems of equations*, SIAM J. Matrix Anal. Appl. **31** (2009), 1382 – 1411.
13. S. Chandrasekaran, M. Gu, and W. Lyons, *A fast adaptive solver for hierarchically semiseparable representations*, Calcolo **42** (2005), no. 3-4, 171–185. MR MR2191196 (2006i:65038)
14. Doina Cioranescu and Jeannine Saint Jean Paulin, *Homogenization of reticulated structures*, Applied Mathematical Sciences, vol. 136, Springer-Verlag, New York, 1999. MR MR1676922 (2000d:74064)
15. Patrick Dewilde and Shivkumar Chandrasekaran, *A hierarchical semi-separable Moore-Penrose equation solver*, Wavelets, multiscale systems and hypercomplex analysis, Oper. Theory Adv. Appl., vol. 167, Birkhäuser, Basel, 2006, pp. 69–85. MR MR2240291 (2007c:65039)
16. Mihai Dorobantu and Björn Engquist, *Wavelet-based numerical homogenization*, SIAM J. Numer. Anal. **35** (1998), no. 2, 540–559 (electronic). MR 99a:65183
17. Yalchin Efendiev and Thomas Y. Hou, *Multiscale finite element methods*, Surveys and Tutorials in the Applied Mathematical Sciences, vol. 4, Springer, New York, 2009, Theory and applications. MR MR2477579
18. Björn Engquist and Olof Runborg, *Wavelet-based numerical homogenization with applications*, Multiscale and multiresolution methods, Lect. Notes Comput. Sci. Eng., vol. 20, Springer, Berlin, 2002, pp. 97–148. MR MR1928565 (2003k:65180)
19. ———, *Wavelet-based numerical homogenization*, Highly oscillatory problems, London Math. Soc. Lecture Note Ser., vol. 366, Cambridge Univ. Press, Cambridge, 2009, pp. 98–126. MR MR2562507
20. A. George, *Nested dissection of a regular finite element mesh*, SIAM J. on Numerical Analysis **10** (1973), 345–363.
21. A. Gillman and P.G. Martinsson, *Fast and accurate numerical methods for solving elliptic difference equations defined on lattices"*., 2010, Submitted for publication.
22. Lars Grasedyck, Ronald Kriemann, and Sabine Le Borne, *Domain decomposition based $\mathcal{H}$-LU preconditioning*, Numer. Math. **112** (2009), no. 4, 565–600. MR MR2507619 (2010e:65200)
23. L. Greengard and V. Rokhlin, *A fast algorithm for particle simulations*, J. Comput. Phys. **73** (1987), no. 2, 325–348. MR MR918448 (88k:82007)
24. Leslie Greengard and Vladimir Rokhlin, *A new version of the fast multipole method for the Laplace equation in three dimensions*, Acta numerica, 1997, Acta Numer., vol. 6, Cambridge Univ. Press, Cambridge, 1997, pp. 229–269.
25. W. Hackbusch, *The panel clustering technique for the boundary element method (invited contribution)*, Boundary elements IX, Vol. 1 (Stuttgart, 1987), Comput. Mech., Southampton, 1987, pp. 463–474. MR MR965331 (89i:76011)
26. W. Hackbusch, B. Khoromskij, and S. Sauter, *On $\mathcal{H}^2$-matrices*, Lectures on Applied Mathematics, Springer Berlin, 2002, pp. 9–29.
27. Wolfgang Hackbusch, *A sparse matrix arithmetic based on H-matrices; Part I: Introduction to H-matrices*, Computing **62** (1999), 89–108.
28. P. G. Martinsson and V. Rokhlin, *A fast direct solver for boundary integral equations in two dimensions*, J. Comput. Phys. **205** (2005), no. 1, 1–23.
29. Per-Gunnar Martinsson, *A fast direct solver for a class of elliptic partial differential equations*, J. Sci. Comput. **38** (2009), no. 3, 316–330. MR MR2475654 (2010c:65041)
30. P.G. Martinsson and V. Rokhlin, *An accelerated kernel independent fast multipole method in one dimension*, SIAM Journal of Scientific Computing **29** (2007), no. 3, 1160–11178.
31. E. Michielssen, A. Boag, and W. C. Chew, *Scattering from elongated objects: direct solution in $O(N \log^2 N)$ operations*, IEE Proc. Microw. Antennas Propag. **143** (1996), no. 4, 277 – 283.
32. Zhifeng Sheng, Patrick Dewilde, and Shivkumar Chandrasekaran, *Algorithms to solve hierarchically semi-separable systems*, System theory, the Schur algorithm and multidimensional analysis, Oper. Theory Adv. Appl., vol. 176, Birkhäuser, Basel, 2007, pp. 255–294. MR MR2342902
33. Page Starr and Vladimir Rokhlin, *On the numerical solution of two-point boundary value problems. II*, Comm. Pure Appl. Math. **47** (1994), no. 8, 1117–1159. MR MR1288634 (95j:65090)



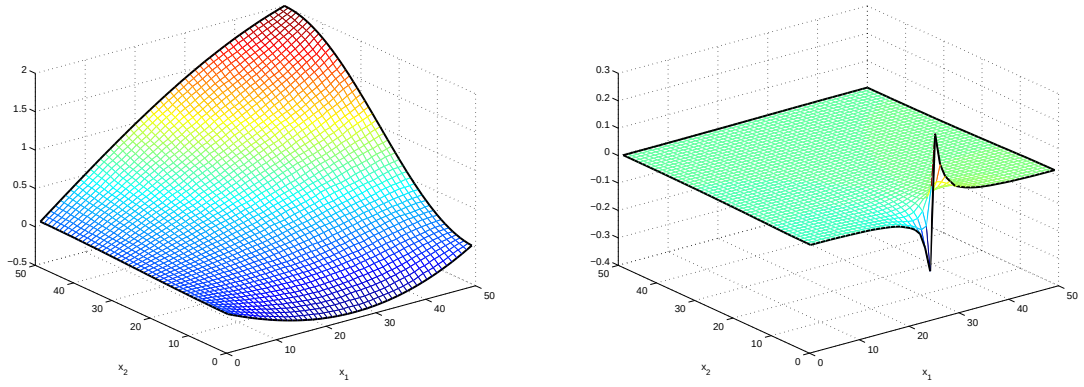
(a) (b)

Fig. 1.4 The periodic problem described as *Case B* in Section 1.3.2 with $N_{\text{cells}} = 4$ and $N_{\text{side}} = 61$. (a) The function $b = b(x)$ defined by (1.15). (b) A solution to the Neumann problem (1.11) with a constant inflow at $x_1 = 1$ and a constant outflow at $x_1 = 0$.



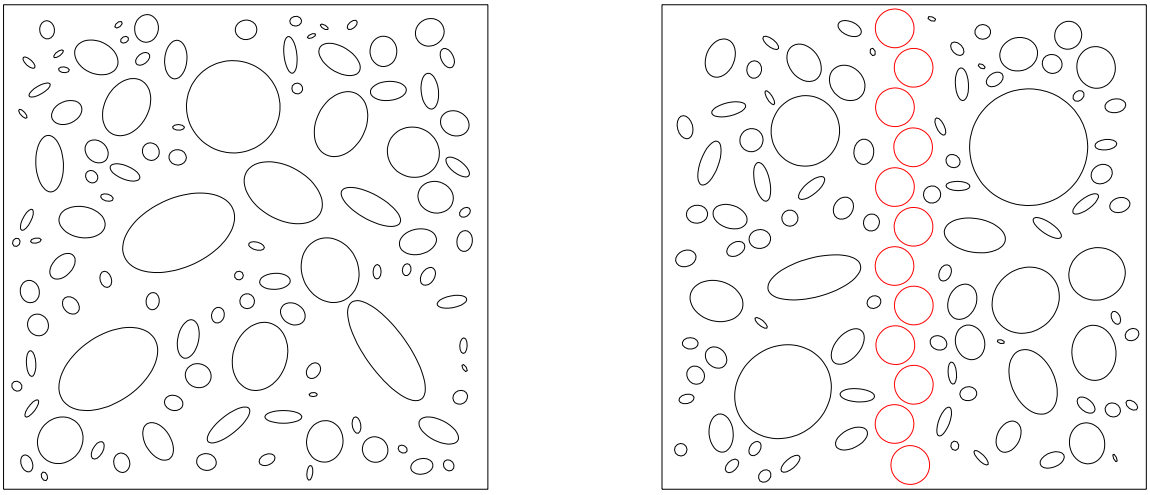
(a) (b)

Fig. 1.5 (a) The lattice with cracks described as *Case E* in Section 1.3.2 for $N_{\text{side}} = 40$. (b) A solution to the Neumann problem (1.11) with a unit inflow at the location marked *source* in (a), and a unit outflow at the location marked *sink*.



(a) (b)

Fig. 1.6 Solutions for non-homogenized equation. (a) Solution resulting the smooth boundary data f_{smooth}. (b) Solution resulting from the rough boundary data f_{rough}.



(a) (b)

Fig. 1.7 Geometry for computations in Section 1.4. (a) A perforated material. (b) A perforated material with a chain of holes that almost line up.

Memory requirements in KB

	$N_{\text{side}} = 100$	$N_{\text{side}} = 200$	$N_{\text{side}} = 400$	$N_{\text{side}} = 800$	$N_{\text{side}} = 1600$
General matrix	1.23e3	4.95e3	1.99e4	7.98e4	3.20e5
Case A (constant cond.)	3.02e2	6.13e2	1.22e3	2.42e3	4.78e3
Case B (periodic cond.)	2.97e2	6.06e2	1.21e3	2.38e3	4.69e3
Case C (random cond.)	3.03e2	6.20e2	1.23e3	2.43e3	4.80e3
Case D (random cuts)	2.96e2	6.06e2	1.20e3	2.38e3	4.70e3
Case E (cracks)	2.96e2	6.10e2	1.22e3	2.42e3	4.77e3

Table 1.1 The table shows the amount of memory (in KB) required for storing the matrix T defined by (1.12) for different problem sizes N_{size}. The first line gives the memory required for storing a general dense matrix of size $4(N_{\text{side}}-1)\times 4(N_{\text{side}}-1)$. The following lines give the amount of memory required to store T in the "HSS" data-sparse format described in Section 1.2 for each each of the five cases described in Section 1.3.2, to within precision 10^{-10}.

Memory requirements in words per degree of freedom

	$N_{\text{side}} = 100$	$N_{\text{side}} = 200$	$N_{\text{side}} = 400$	$N_{\text{side}} = 800$	$N_{\text{side}} = 1600$
General matrix	396	796	1596	3196	6396
Case A (constant conductivities)	97.7	98.6	98.1	96.8	95.7
Case B (periodic conductivities)	95.9	97.4	96.7	95.4	93.9
Case C (random conductivities)	97.8	99.7	98.8	97.5	96.0
Case D (random cuts)	95.5	97.5	96.6	95.4	94.1
Case E (cracks)	95.7	98.1	97.7	96.8	95.5

Table 1.2 The table shows the same data given in Table 1.1, but now scaled to demonstrate that the memory requirement scales linearly with problem size. To be precise, the entries given are the number of "words" (the memory required to store a floating point number to double precision accuracy) required per node on the boundary.

HSS ranks of the Schur complements for a matrix of size 6396×6396

	$N_{\text{block}} = 50$	$N_{\text{block}} = 100$	$N_{\text{block}} = 200$	$N_{\text{block}} = 400$	$N_{\text{block}} = 800$	$N_{\text{block}} = 1600$
General matrix	50	100	200	400	800	1600
Case A (constant conductivities)	19.3	22.7	26.0	31.0	39.0	53.0
Case B (periodic conductivities)	18.8	21.6	24.8	29.3	37.0	50.0
Case C (random conductivities)	19.3	22.8	26.8	31.6	39.8	54.0
Case D (random cuts)	18.7	21.9	25.5	30.8	38.8	52.5
Case E (cracks)	19.2	22.7	25.9	30.9	38.8	52.5

Table 1.3 The table shows the HSS-ranks (as described in Remark 5) of blocks in the solution operator for the different models. The reported rank was the average numerical rank (at precision 10^{-10}) over all HSS blocks of size N_{block} that arise in the compressed representation.

Errors in homogenized operator for "Case C"

	Homogenization	Homogenization with buffer of width b					
	with no buffer	$b = 1$	$b = 2$	$b = 3$	$b = 4$	$b = 5$	$b = 10$
E_{D2N}	1.9e-01	5.4e-03	1.2e-03	3.9e-04	3.3e-04	1.3e-04	6.6e-05
E_{N2D}	1.1e-02	7.5e-03	5.6e-03	5.7e-03	4.3e-03	4.9e-03	2.4e-03
E_{smooth}	7.3e-03	4.1e-03	4.1e-03	4.1e-03	2.8e-03	2.6e-03	1.4e-03
E_{rough}	1.5e-01	2.1e-02	1.1e-02	2.2e-03	8.8e-04	3.5e-03	9.2e-04

Table 1.4 Discrepancy between the solution operator of an given lattice, and the homogenized solution operator. These numbers refer to the model described as "Case C" in Section 1.3.2 (random conductivities). The errors E_{D2N}, E_{N2D}, E_{smooth}, and E_{rough} are defined in equations (1.16) and (1.17).

Errors in homogenized operator for "Case D"

	Homogenization	Homogenization with buffer of width b					
	with no buffer	$b = 1$	$b = 2$	$b = 3$	$b = 4$	$b = 5$	$b = 10$
E_{D2N}	4.4e-01	1.5e-02	4.5e-03	1.7e-03	1.2e-03	7.6e-04	3.3e-04
E_{N2D}	8.7e-02	6.1e-02	5.6e-02	5.2e-02	4.5e-02	4.4e-02	2.8e-02
E_{smooth}	7.4e-02	5.9e-02	5.4e-02	4.8e-02	4.2e-02	4.1e-02	2.7e-02
E_{rough}	1.0e-01	7.0e-02	6.8e-02	6.2e-02	5.1e-02	5.0e-02	3.4e-02

Table 1.5 Discrepancy between the solution operator of an given lattice, and the homogenized solution operator. These numbers refer to the model described as "Case D" in Section 1.3.2 (randomly cut bars). The errors E_{D2N}, E_{N2D}, E_{smooth}, and E_{rough} are defined in equations (1.16) and (1.17).

Average ranks of HSS blocks for composite material example in Section 1.4

	$N_{\text{block}} = 36$	$N_{\text{block}} = 71$	$N_{\text{block}} = 143$	$N_{\text{block}} = 286$
Geometry shown in Figure 1.7(a)	18.2	27.0	39.5	25.8
Geometry shown in Figure 1.7(b)	18.3	27.3	41.1	28.0

Table 1.6 The average HSS-ranks (as defined in Remark 5) for the blocks in a data-sparse representation of the Neumann-to-Dirichlet operator for the geometries shown in Figure 1.7.