Locally-synchronous, iterative solver for Fourier-based homogenization

R. Glüge¹ · H. Altenbach² · S. Eisentträger³

Received: 22 April 2020 / Accepted: 11 January 2021 / Published online: 16 February 2021
 © The Author(s) 2021

Abstract

We use the algebraic orthogonality of rotation-free and divergence-free fields in the Fourier space to derive the solution of a class of linear homogenization problems as the solution of a large linear system. The effective constitutive tensor constitutes only a small part of the solution vector. Therefore, we propose to use a synchronous and local iterative method that is capable to efficiently compute only a single component of the solution vector. If the convergence of the iterative solver is ensured, i.e., the system matrix is positive definite and diagonally dominant, it outperforms standard direct and iterative solvers that compute the complete solution. It has been found that for larger phase contrasts in the homogenization problem, the convergence is lost, and one needs to resort to other linear system solvers. Therefore, we discuss the linear system's properties and the advantages as well as drawbacks of the presented homogenization approach.

Keywords Computational micromechanics · Homogenization · Fourier method · Locally-synchronous solver · Iterative solver

Contents

Highlights	599	4.2 Stationary linear iterative methods	606
1 Introduction	600	4.2.1 Richardson method	607
2 Notation	601	4.2.2 Jacobi method	607
3 Fourier method	601	4.2.3 Gauss–Seidel method	608
3.1 Fourier series representation	601	4.2.4 Successive over-relaxation	608
3.2 Decomposition of periodic fields into three orthogonal parts	602	4.3 Locally synchronous approximation of a single component of the solution vector	609
3.3 The polarization problem	604	4.3.1 Neumann series representation	609
3.4 Solving for the effective conductivity	604	4.3.2 Extraction of a single component	609
4 Synchronous local algorithm for the iterative solution of linear systems of equations	605	4.3.3 Algorithm	609
4.1 Motivation	606	5 Results	610
		5.1 Heat conduction problem	610
		5.2 Verification problem: Laminate structure	611
		5.3 Complex microstructure	611
		5.4 Properties of the linear system	614
		5.5 Comparison to other linear systems	614
		5.6 Limitations	615
		6 Summary	615
		Appendix A. Mathematica scripts	616
		References	617

✉ R. Glüge
 gluege@uni-bremen.de
 H. Altenbach
 holm.altenbach@ovgu.de
 S. Eisentträger
 s.eisentraeger@unsw.edu.au

¹ Faculty of Production Engineering, University of Bremen, Bibliothekstraße 1, 28359 Bremen, Germany
² Institute of Mechanics, Faculty of Mechanical Engineering, Otto von Guericke University Magdeburg, Universitätsplatz 2, 39106 Magdeburg, Germany
³ School of Civil and Environmental Engineering, The University of New South Wales Sydney, Sydney, NSW 2052, Australia

Highlights

- Development of a Fourier method for linear homogenization problems.
- Implementation of a local and synchronous solver for the solution of systems of linear equations.

- Iterative computation of a single component of the solution vector.
- Performance analysis of the novel approach compared to classical direct and iterative solvers.

1 Introduction

Homogenization subsumes methods and approaches that aim to provide the effective material properties of materials with microstructures. In the context of linear constitutive laws, the effective constitutive tensor $\mathbf{L}^*$ relates the volume average of a gradient field $\mathbf{g}$ (strains, temperature gradient, charge density gradient, concentration gradient, etc.) to its thermodynamic conjugate divergence free field $\mathbf{q}$ (stresses, heat flux, electric current, concentration flow, etc.),

$$\bar{\mathbf{q}} = \mathbf{L}^* \bar{\mathbf{g}}. \quad (1)$$

Locally, the constitutive law is location dependent. Often, $\mathbf{L}^*$ is approximated by $\bar{\mathbf{L}}$ or a more elaborate mean field approach, some of which are summarized in Ref. [1]. More accurate results are obtained by solving the boundary value problem of a large virtual material sample—a representative volume element (RVE)—and extracting the average fields from the numerical solution. In the linear case, $\mathbf{L}^*$ is then reconstructed by at most n such solutions, where n is the dimension of the vector space of $\mathbf{g}$ and $\mathbf{q}$.

In the context of examining RVE realizations of microstructures, Fourier methods are an indispensable tool for various reasons. They are typically applied in different ways which are summarized in the following:

- Microstructural data is often provided as pixel (2D) or voxel (3D) data, which can be efficiently transformed to its discrete Fourier representation.
- The Fourier function basis satisfies the boundary conditions of the most common RVE problem of a periodically repeatable unit cell a priori.
- The Fast Fourier Transform (FFT) allows to switch quickly between real and Fourier space, which allows for a fast evaluation of differential operators.
- In their Fourier representation, the orthogonality of divergence-free and rotation-free fields (i.e., the Helmholtz decomposition of a vector field) can be exploited algebraically to derive a closed-form expression for the effective constitutive law.

That is to say, two distinct features—the availability of the fast transformation and the algebraic orthogonality of

divergence- and rotation-free fields in Fourier space—are exploited differently, but both methods are summarized under the Fourier label.

The first kind of Fourier (spectral) solver is a fixed point iteration scheme introduced by [2,3]. It solves a concrete RVE problem by rewriting the homogeneous partial differential equation (PDE) with non-constant coefficients that is to be solved. On the RVE domain the local material tensor $\mathbf{L}(\mathbf{x}) = \mathbf{L}_0 + \tilde{\mathbf{L}}(\mathbf{x})$ is split and the $\tilde{\mathbf{L}}(\mathbf{x})$ -part is moved to the right-hand side, interpreting the new equation as an inhomogeneous PDE with constant coefficients. Solving this auxiliary problem successively corresponds to a fixed point iteration for the original problem. The easy transformation between real and Fourier spaces enables a fast evaluation of the right-hand side, i.e., an efficient iteration, by taking the derivatives in the Fourier space and applying the constitutive law involving $\tilde{\mathbf{L}}(\mathbf{x})$ in real space. Per iteration, one forward and backward transform are required. The effective properties are obtained as the linear relation between the homogeneous parts of the solution fields. For a complete identification of the effective properties, several RVE problems need to be solved (e.g., three average temperature gradients that are prescribed in orthogonal directions) or material symmetries need to be presumed. The main advantage of this method is that only function evaluations are needed, i.e., we do not explicitly need to solve a linear system. Therefore, this method is less memory-consuming compared to standard homogenization approaches, such that discretizations of 256^3 voxels are feasible [4]. Note that only the discretized fields of the microstructural data and the $\mathbf{q}$, $\mathbf{g}$, $\mathbf{L}$ -fields need to be kept in the computer's memory. Such schemes are usually referred to as spectral or FFT solvers. Variants of this approach can be found in the pertinent literature. Some effort went into reducing the already low memory requirements [5–8] and improving the iteration's convergence rate [9–12]. A more complete summary can be found in a recent paper by Wicht et al. [4]. In the remainder of this article, we refer to this method as “Moulinec-Suquet (MS) fixed point iteration”.

The second kind of spectral solver exploits the Helmholtz decomposition, i.e., the orthogonality of gradient (rotation-free) and divergence-free fields, which is algebraic in Fourier space but differential in real space. The algebraic orthogonality in Fourier space allows to introduce projectors based on which a closed-form expression for the effective properties is derived. These methods rely on the linearity of the constitutive law, which is merged with the linear projection operations. To distinguish this method from a fixed-point iteration, we refer to it as the “projection method”. An excellent account can be found in the textbook by G. Milton [13] in

Sect. 12. This method is entirely formulated in the Fourier space. The fast Fourier transform is only needed once to set up the linear system in Fourier space. The latter is quite large, but needs to be solved only for a few variables. Approximate variants of this approach have been presented in Refs. [14–18]. At least to the authors' knowledge, a concise formulation of this approach has been first presented by Chen in Chap. 5 of Ref. [19]. She managed to solve such systems for discretizations d of the unit cell up to 11^3 voxels (data points). That is rather limited compared to today's standards, which is due to the memory intensive linear systems (see Sect. 5). It has complex coefficients, is unsymmetric, not sparse, and without a band structure. For the sake of clarity, the two approaches are compared in Table 1.

In the present contribution, the focus is on the second class of methods, which is by far the less popular of the two Fourier-based approaches. This is due to the much larger numerical effort, compared to the fixed-point method, because an unsymmetric non-sparse linear system with complex coefficients is solved for the tensorial auxiliary field (constitutive tensor, stiffness or conductivity), which has moreover more components than the usual gradient field (strains, temperature gradient) that is solved for in the fixed-point iteration approach. However, we found that on the numerical side, some unused potential can be leveraged. Because only the homogeneous part of the auxiliary tensorial field is needed, and this is represented by the zeroth Fourier coefficient, only few solution components need to be computed. Thus, instead of applying a general purpose linear solver, a Jacobi iteration for the relevant solution components is used. Although the resulting scheme is similar to the fixed-point iteration method in the sense that the solution is iterated, it is still a different approach.

The article is divided into six sections including the introduction. Before we discussed the proposed methodology, the basic notation is explained in Sect. 2. The Fourier method is recalled in Sect. 3, while we describe the local iterative solver in Sect. 4. With this the theoretical foundation for our method is laid and we examine a specific example in Sect. 5. Here, also a comparison of the novel method and standard solvers in terms of their computational efficiency is provided. In Sect. 6, a brief summary of the most important findings is provided.

2 Notation

In this article, a direct notation is preferred. Vectors are denoted as bold minuscules, like $\mathbf{p}$ for the polarization, $\mathbf{q}$ for the heat flux or $\mathbf{g}$ for the temperature gradient. All tensorial quantities are represented w.r.t. the orthonormal basis $\mathbf{e}_i$ when their components are considered. Locations in real space are denoted by $\mathbf{x}$, for which we use Cartesian coor-

dinates, i.e., $\mathbf{x} = x_i \mathbf{e}_i$. Second-order tensors are written as bold majuscules, such as $\mathbf{L}$ for the heat conduction tensor. The dyadic product and scalar contractions are denoted by $(\mathbf{a} \otimes \mathbf{b} \otimes \mathbf{c}) : (\mathbf{d} \otimes \mathbf{e}) = (\mathbf{b} \cdot \mathbf{d})(\mathbf{c} \cdot \mathbf{e})\mathbf{a}$, with a dot being the usual scalar product between vectors. This notation is mostly needed for linear mappings between vectors, as in $\mathbf{q} = \mathbf{L} \cdot \mathbf{g}$. When the product is clear from the context, as in the last equation, the scalar dot is occasionally omitted. We further use the nabla operator ∇ and projection operators $\Gamma\{\square\}$. The latter involves the projectors $\mathbf{K} = (\mathbf{k} \cdot \mathbf{k})^{-1} \mathbf{k} \otimes \mathbf{k}$ and $\mathbf{I} - \mathbf{K}$, which project a vector onto its part parallel and perpendicular to $\mathbf{k}$, where $\mathbf{k}$ is the wave vector of the Fourier base function.

3 Fourier method

In the current section, we present the fundamental ideas of the applied Fourier method. They can be found as well in the textbook of Milton [13] in Section 12. To this end, the Fourier series, the decomposition of periodic fields, the polarization problem, and the solution for effective material properties are discussed based on a generic problem and later for elastostatics and heat conduction as specific example problems.

3.1 Fourier series representation

For the sake of completeness, a brief introduction to Fourier series representations of functions is provided in the following. Let

$$[f(x), g(x)] = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(x) \overline{g(x)} dx \quad (2)$$

denote the scalar product of two functions, where the overline $\overline{\square}$ denotes the complex conjugate. It is easy to see that the functions

$$f_k(x) = e^{ikx} \text{ with } k \in \mathbb{Z} \quad (3)$$

form an orthonormal basis, i.e.,

$$[f_k(x), f_l(x)] = \delta_{kl}, \quad (4)$$

where δ_{kl} denotes the Kronecker(-delta) symbol and $i = \sqrt{-1}$ is the imaginary unit. With respect to this basis, a sufficiently smooth function can be represented as an infinite series of $f_k(x)$

$$f(x) = \sum_{k=-\infty}^{\infty} \hat{f}_k e^{ikx}, \quad (5)$$

Table 1 Comparison of different Fourier methods

Method	Fixed-point iteration	Projection method
Unknown field solution variables	gradient field that is governed by the equilibrium equation, i.e., strain field, temperature gradient, etc., defined in the space $\mathcal{V}$	auxiliary tensorial field in space $\mathcal{V} \otimes \mathcal{V}$ in which the constitutive tensor is defined
Equation to be solved	equilibrium equation of a boundary value problem in a virtual material sample	linear system in Fourier space
Solution method	fixed-point iteration	general purpose linear solver
Fourier transform	one FFT and inverse FFT per iteration	once before setting up the linear system
Advantages	very fast, low memory usage, can be extended to nonlinear problems	expression for the effective properties

where the component $\hat{f}_k$ (also called amplitude or spectral coefficient) is given by the scalar product with $f_k(x)$

$$\hat{f}_k = [f(x), \bar{f}_k(x)]. \quad (6)$$

A generalization to functions defined in $\mathbb{R}^3$ is obtained by specifying the direction, usually referred to as the plane wave vector $\mathbf{k}$ with integer components

$$f_{\mathbf{k}}(\mathbf{x}) = e^{i\mathbf{k} \cdot \mathbf{x}} \text{ with } k_i \in \mathbb{Z}. \quad (7)$$

where k_i are the components of $\mathbf{k}$ w.r.t. the orthonormal basis $\mathbf{e}_i$. The confinement to integer k_i is due to the Fourier series expansion. In the next step, the plane wave vector $\mathbf{k}$ is decomposed into a unit direction vector $\mathbf{n}_k$ and its magnitude $\tilde{k}$

$$\mathbf{n}_k = \mathbf{k}/|\mathbf{k}|, \quad (8)$$

$$\tilde{k} = |\mathbf{k}|. \quad (9)$$

$\mathbf{n}_k$ represents the direction of the plane wave, while $\tilde{k}$ is the wave number, which is connected to the frequency f and the wavelength λ , respectively, by

$$f = \tilde{k}^{-1} \quad \text{or} \quad \lambda = 2\pi/\tilde{k}.$$

The scalar product defined by Eq. (2) is then

$$[f_{\mathbf{k}}(\mathbf{x}), f_{\mathbf{g}}(\mathbf{x})] = \frac{1}{(2\pi)^3} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} f_{\mathbf{k}}(\mathbf{x}) \bar{f}_{\mathbf{g}}(\mathbf{x}) \, dx_1 dx_2 dx_3. \quad (10)$$

Again we see that

$$[f_{\mathbf{k}}(\mathbf{x}), f_{\mathbf{g}}(\mathbf{x})] = \delta_{\mathbf{kg}}, \quad (11)$$

where $\delta_{\mathbf{kg}}$ may be considered as a generalized Kronecker symbol

$$\delta_{\mathbf{kg}} = \begin{cases} 1 & : k_1 = g_1, k_2 = g_2, k_3 = g_3 \\ 0 & : \text{otherwise} \end{cases}. \quad (12)$$

With this, the approximation of a function defined in $\mathbb{R}^3$ by a Fourier series becomes

$$\begin{aligned} f(\mathbf{x}) &= \sum_{k_1, k_2, k_3 = -\infty}^{\infty} e^{i\mathbf{k} \cdot \mathbf{x}} \hat{f}_{\mathbf{k}} \\ &= \sum_{k_1 = -\infty}^{\infty} e^{ik_1 x_1} \sum_{k_2 = -\infty}^{\infty} e^{ik_2 x_2} \sum_{k_3 = -\infty}^{\infty} e^{ik_3 x_3} \hat{f}_{\mathbf{k}}. \end{aligned} \quad (13)$$

The components of the function in the Fourier space are obtained via a scalar product

$$\hat{f}_{\mathbf{k}} = [f(\mathbf{x}), f_{\mathbf{k}}(\mathbf{x})]. \quad (14)$$

3.2 Decomposition of periodic fields into three orthogonal parts

In elastostatics and Fourier heat conduction we have the following relations that govern the behaviour of a system, with the local balances, the constitutive laws and the integrability condition for the driving force of the process (listed from left to right):

$$\nabla \cdot \boldsymbol{\sigma} = \mathbf{o}, \quad \boldsymbol{\sigma} = \mathbb{C} : \boldsymbol{\varepsilon}, \quad \nabla \times \boldsymbol{\varepsilon} \times \nabla = \mathbf{O}, \quad (15)$$

$$\nabla \cdot \mathbf{q} = 0, \quad \mathbf{q} = \mathbf{L} \cdot \mathbf{g}, \quad \nabla \times \mathbf{g} = \mathbf{o}. \quad (16)$$

The same mathematical structure is found for Fick's diffusion, Ohmic resistance, or any process in which the flux of a conserved quantity is linear in the gradient of a field. Being concerned with a homogenization task, we removed the source terms from the balance equations, which are a heat source (or sink) in heat conduction problems and a volume specific force $\rho \mathbf{b}$ and the inertia force density $\rho \ddot{\mathbf{x}}$ in elastostatics. The reason for this is that these are not material properties that can be homogenized, but externally applied, material independent effects.

For the sake of notational convenience, we drop the dependence on the location $\mathbf{x}$ of all fields. The differential constraints on $\boldsymbol{\varepsilon}$ and $\mathbf{g}$ are the so-called integrability con-

ditions which ensure the existence of a field in case of the gradient $\mathbf{g} = \nabla f$, where the interpretation of f depends on the physical problem and may be the temperature field, a concentration, or electric charge. In case of elasticity, the integrability conditions ensure the existence of a displacement field $\mathbf{u}$ for the strain field $\boldsymbol{\varepsilon} = (\mathbf{u} \otimes \nabla + \nabla \otimes \mathbf{u})/2$.

Next, we consider all periodic fields on a unit cell Ω . Depending on the homogenization task, these are $\{\mathbf{q}, \mathbf{L}, \mathbf{g}\}$ or $\{\sigma, \mathbb{C}, \boldsymbol{\varepsilon}\}$. We proceed, for convenience, with the heat conduction problem. One can directly generalize the derivation given in the following to elastostatics. The divergence is obtained by the scalar product of $\mathbf{f}_{\mathbf{k}}$ with $\mathbf{k}$

$$\nabla \cdot \mathbf{f}(\mathbf{x}) = \sum_{\forall \mathbf{k}} i e^{i\mathbf{x} \cdot \mathbf{k}} (\hat{\mathbf{f}}_{\mathbf{k}} \cdot \mathbf{k}). \quad (17)$$

Therefore, we can expand the Fourier series by the projector

$$\mathbf{K} = \frac{1}{\mathbf{k} \cdot \mathbf{k}} \mathbf{k} \otimes \mathbf{k} \quad (18)$$

without affecting the divergence-part of the field

$$\mathbf{f}_{\nabla}(\mathbf{x}) := \sum_{\forall \mathbf{k}} i e^{i\mathbf{x} \cdot \mathbf{k}} \mathbf{K} \hat{\mathbf{f}}_{\mathbf{k}}. \quad (19)$$

With this definition we have $\mathbf{f}(\mathbf{x}) \cdot \nabla = \mathbf{f}_{\nabla}(\mathbf{x}) \cdot \nabla$. An analogous calculation for the rotation gives

$$\mathbf{f}(\mathbf{x}) \times \nabla = \sum_{\forall \mathbf{k}} i e^{i\mathbf{x} \cdot \mathbf{k}} (\hat{\mathbf{f}}_{\mathbf{k}} \times \mathbf{k}). \quad (20)$$

This time, the part of $\hat{\mathbf{f}}_{\mathbf{k}}$ parallel to $\mathbf{k}$ can be removed due to the vector product of parallel vectors being zero. Thus, we can expand with $\mathbf{I} - \mathbf{K}$ without affecting the rotational part of the field

$$\mathbf{f}_{\times \nabla}(\mathbf{x}) := \sum_{\forall \mathbf{k}} i e^{i\mathbf{x} \cdot \mathbf{k}} (\mathbf{I} - \mathbf{K}) \hat{\mathbf{f}}_{\mathbf{k}}. \quad (21)$$

When normalizing $\mathbf{k}$, one can see that the part for $\mathbf{k} = \mathbf{o}$ is problematic. It represents the homogeneous mean value of the field, which is both divergence- and rotation-free. It needs to be addressed directly in the decomposition into three parts

$$\begin{aligned} \mathbf{f}(\mathbf{x}) = & \underbrace{\hat{\mathbf{f}}_{\mathbf{o}}}_{\text{homogeneous part } \bar{\mathbf{f}}} + \underbrace{\sum_{\forall \mathbf{k} \neq \mathbf{o}} e^{i\mathbf{x} \cdot \mathbf{k}} (\mathbf{I} - \mathbf{K}) \hat{\mathbf{f}}_{\mathbf{k}}}_{\text{divergence-free fluctuating part } \tilde{\mathbf{f}}_{\nabla}(\mathbf{x})} \\ & + \underbrace{\sum_{\forall \mathbf{k} \neq \mathbf{o}} e^{i\mathbf{x} \cdot \mathbf{k}} \mathbf{K} \hat{\mathbf{f}}_{\mathbf{k}}}_{\text{rotation-free fluctuating part } \tilde{\mathbf{f}}_{\nabla}(\mathbf{x})}. \end{aligned} \quad (22)$$

As can be seen, $\mathbf{K}$ and $\mathbf{I} - \mathbf{K}$ filter the corresponding parts from the function *without the imaginary unit*. The imaginary unit appears upon taking the derivative, but we want to decompose the function itself. In real space, the orthogonality is easy to see as well

$$(\mathbf{f}(\mathbf{x}) \times \nabla) \cdot \nabla = f_{i,jk} \varepsilon_{ijk} = 0 \quad (23)$$

with the permutation symbol ε_{ijk} and Schwarz' theorem. The derivatives with respect to the coordinates x_i are denoted by $\square_{,i}$. The Helmholtz-decomposition of $\mathbf{f}(\mathbf{x})$ into rotation- and divergence-free parts is not easily obtained in real space. But in the Fourier space, we just define three different Gamma-operators $\Gamma_{\square}$:

$$\Gamma_0\{\mathbf{f}(\mathbf{x})\} = \bar{\mathbf{f}} = \hat{\mathbf{f}}_{\mathbf{o}}, \quad (24)$$

$$\begin{aligned} \Gamma_{\nabla}\{\mathbf{f}(\mathbf{x})\} &= \mathbf{f}(\mathbf{x}) - \bar{\mathbf{f}} - \tilde{\mathbf{f}}_{\times \nabla}(\mathbf{x}) = \tilde{\mathbf{f}}_{\nabla}(\mathbf{x}) \\ &= \sum_{\forall \mathbf{k} \neq \mathbf{o}} e^{i\mathbf{x} \cdot \mathbf{k}} \mathbf{K} \hat{\mathbf{f}}_{\mathbf{k}}, \end{aligned} \quad (25)$$

$$\begin{aligned} \Gamma_{\times \nabla}\{\mathbf{f}(\mathbf{x})\} &= \mathbf{f}(\mathbf{x}) - \bar{\mathbf{f}} - \tilde{\mathbf{f}}_{\nabla}(\mathbf{x}) = \tilde{\mathbf{f}}_{\times \nabla}(\mathbf{x}) \\ &= \sum_{\forall \mathbf{k} \neq \mathbf{o}} e^{i\mathbf{x} \cdot \mathbf{k}} (\mathbf{I} - \mathbf{K}) \hat{\mathbf{f}}_{\mathbf{k}}. \end{aligned} \quad (26)$$

These projectors extract the mean value ($\Gamma_0\{\square\}$), the divergence-free ($\Gamma_{\times \nabla}\{\square\}$) and the rotation-free fluctuating ($\Gamma_{\nabla}\{\square\}$) parts of a field. Note that the application of the $\Gamma_{\square}$ -operators is an algebraic operation, but not a simple matrix-product, and the application is not associative, i.e., $\mathbf{A}(\Gamma_{\square}\mathbf{B}) \neq (\mathbf{A}\Gamma_{\square})\mathbf{B}$. Therefore, we denote the action of the operators by curly brackets as $\Gamma_{\square}\{\square\}$. It is clear that the three parts are orthogonal to each other w.r.t. the scalar product

$$[\mathbf{f}_i(\mathbf{x}), \mathbf{f}_j(\mathbf{x})] = \frac{1}{V} \int_{\Omega} \mathbf{f}_i(\mathbf{x}) \cdot \mathbf{f}_j(\mathbf{x}) d\mathbf{x} = 0 \quad \text{if } i \neq j. \quad (27)$$

Elasticity The equations of elastostatics obey the same structure as the equations of heat conduction, cf. Eq. (15), which results in a similar decomposition of symmetric tensor fields into the homogeneous, divergence- and rotation-free parts:

$$\begin{aligned} \mathbf{A}(\mathbf{x}) = & \underbrace{\hat{\mathbf{A}}_{\mathbf{o}}}_{\text{homogeneous part}} + \underbrace{\sum_{\forall \mathbf{k} \neq \mathbf{o}} e^{i\mathbf{x} \cdot \mathbf{k}} (\mathbb{I} - \mathbf{K} \otimes \mathbf{K}) : \hat{\mathbf{A}}_{\mathbf{k}}}_{\text{divergence-free fluctuating part } \tilde{\mathbf{A}}_{\nabla}(\mathbf{x})} \\ & + \underbrace{\sum_{\forall \mathbf{k} \neq \mathbf{o}} e^{i\mathbf{x} \cdot \mathbf{k}} \mathbf{K} \otimes \mathbf{K} : \hat{\mathbf{A}}_{\mathbf{k}}}_{\text{rotation-free fluctuating part } \tilde{\mathbf{A}}_{\nabla}(\mathbf{x})}. \end{aligned} \quad (28)$$

In the latter equation, one can easily identify the projection operators, where the projector that removes the diverging part

of $\mathbf{A}$ involves the fourth order identity on symmetric second order tensors $\mathbb{I}$. The strain tensor field $\boldsymbol{\varepsilon}(\mathbf{x})$ is rotation-free, hence its Fourier series can be written as

$$\boldsymbol{\varepsilon}(\mathbf{x}) = \hat{\boldsymbol{\varepsilon}}_0 + \sum_{\forall \mathbf{k} \neq 0} e^{i\mathbf{x} \cdot \mathbf{k}} \mathbf{K} \otimes \mathbf{K} : \hat{\boldsymbol{\varepsilon}}_{\mathbf{k}}, \quad (29)$$

while the stress field $\boldsymbol{\sigma}(\mathbf{x})$ is divergence-free and symmetric, hence its Fourier series can be written as

$$\boldsymbol{\sigma}(\mathbf{x}) = \hat{\boldsymbol{\sigma}}_0 + \sum_{\forall \mathbf{k} \neq 0} e^{i\mathbf{x} \cdot \mathbf{k}} (\mathbb{I} - \mathbf{K} \otimes \mathbf{K}) : \hat{\boldsymbol{\sigma}}_{\mathbf{k}}. \quad (30)$$

3.3 The polarization problem

We consider the same boundary value problem with an inhomogeneous conductivity $\mathbf{L}(\mathbf{x})$ and a homogeneous reference conductivity $\mathbf{L}_0$

$$\mathbf{q}(\mathbf{x}) \cdot \nabla = 0, \quad \mathbf{q}(\mathbf{x}) = \mathbf{L}(\mathbf{x})\mathbf{g}(\mathbf{x}), \quad (31)$$

$$\mathbf{q}_0 \cdot \nabla = 0, \quad \mathbf{q}_0 = \mathbf{L}_0\mathbf{g}_0. \quad (32)$$

We apply homogeneous boundary conditions $\bar{\mathbf{g}}$ on the entire surface. This makes the second problem trivial: all fields are homogeneous, moreover we have $\mathbf{g}_0 = \bar{\mathbf{g}} = \mathbf{g}(\mathbf{x}) - \tilde{\mathbf{g}}(\mathbf{x})$, where $\tilde{\mathbf{g}}(\mathbf{x})$ denotes the fluctuation part of the $\mathbf{g}$ -field. Our interest is to find the linear mapping between the homogeneous parts

$$\bar{\mathbf{q}} = \mathbf{L}^* \cdot \bar{\mathbf{g}}, \quad (33)$$

which constitutes the implicit definition of the effective conductivity $\mathbf{L}^*$. We take the difference between Eqs. (31) and (32)

$$\underbrace{(\mathbf{q}(\mathbf{x}) - \mathbf{q}_0)}_{\Delta \mathbf{q}(\mathbf{x})} \cdot \nabla = 0, \quad \underbrace{(\mathbf{q}(\mathbf{x}) - \mathbf{q}_0)}_{\Delta \mathbf{q}(\mathbf{x})} = \mathbf{L}(\mathbf{x})\mathbf{g}(\mathbf{x}) - \mathbf{L}_0(\mathbf{g}(\mathbf{x}) - \tilde{\mathbf{g}}(\mathbf{x})) \quad (34)$$

which is more compactly written as

$$\Delta \mathbf{q}(\mathbf{x}) \cdot \nabla = 0, \quad \Delta \mathbf{q}(\mathbf{x}) = (\mathbf{L}(\mathbf{x}) - \mathbf{L}_0)\mathbf{g}(\mathbf{x}) - \mathbf{L}_0\tilde{\mathbf{g}}(\mathbf{x}). \quad (35)$$

Note that here Δ only denotes a difference and should not be confused with the Laplace-operator. We move $\mathbf{L}_0\tilde{\mathbf{g}}(\mathbf{x})$ to the left-hand side

$$\mathbf{p}(\mathbf{x}) = \Delta \mathbf{q}(\mathbf{x}) + \mathbf{L}_0\tilde{\mathbf{g}}(\mathbf{x}), \quad (36)$$

where $\mathbf{p}(\mathbf{x})$ is defined as

$$\mathbf{p}(\mathbf{x}) = (\mathbf{L}(\mathbf{x}) - \mathbf{L}_0)\mathbf{g}(\mathbf{x}), \quad (37)$$

or in expanded form

$$\mathbf{p}(\mathbf{x}) = \mathbf{q}(\mathbf{x}) - \mathbf{L}_0\mathbf{g}(\mathbf{x}). \quad (38)$$

This is the so-called polarization problem. The name is most likely related to its first area of application which was electric polarization. Note that $\mathbf{p}(\mathbf{x})$ is neither divergence- nor rotation-free. This underlines its character as an auxiliary problem, which is not solved, but used to derive an explicit expression for $\mathbf{L}^*$. Taking the homogeneous part by applying Γ_0 gives

$$\bar{\mathbf{p}} = \bar{\mathbf{q}} - \mathbf{L}_0\bar{\mathbf{g}}, \quad (39)$$

which can be brought into the form

$$\bar{\mathbf{q}} = \bar{\mathbf{p}} + \mathbf{L}_0\bar{\mathbf{g}}. \quad (40)$$

With an explicit expression for $\bar{\mathbf{p}}$, which is linear in $\bar{\mathbf{g}}$, one can identify the effective conductivity.

3.4 Solving for the effective conductivity

All fields, except the reference conductivity $\mathbf{L}_0$, the mean values $\bar{\mathbf{q}}$, $\bar{\mathbf{p}}$ and $\bar{\mathbf{g}}$ and the effective conductivity $\mathbf{L}^*$ depend on the location $\mathbf{x}$ in real space or the wave vector $\mathbf{k}$ in Fourier space. For the sake of convenience, we drop these dependencies in the remainder of this section. Further, we use $\mathbf{L}_0 = l_0\mathbf{I}$, which simplifies calculations. Apart from that, the choice of $\mathbf{L}_0$ is not important for us, since we obtain a closed form expression for $\mathbf{L}^*$. For spectral solvers of the first type (see Sect. 1), the choice of $\mathbf{L}_0$ is crucial for the convergence. Next, we solve symbolically for $\bar{\mathbf{p}}$ with the aid of the $\Gamma_{\square}$ -operators. We firstly apply $\Gamma_{\nabla}\{\mathbf{L}_0^{-1}\{\square\}\}$ to Eq. (38):

$$\Gamma_{\nabla}\{\mathbf{L}_0^{-1}\mathbf{p}\} = \Gamma_{\nabla}\{\mathbf{L}_0^{-1}\mathbf{q}\} - \Gamma_{\nabla}\{\mathbf{g}\}. \quad (41)$$

Since $\mathbf{g} = \bar{\mathbf{g}} + \tilde{\mathbf{g}}$ is rotation-free, $\Gamma_{\nabla}\{\square\}$ removes merely the homogeneous part of $\mathbf{g}$. Since $\mathbf{q}$ is divergence-free and $\mathbf{L}_0$ is a multiple of the identity tensor, we have $\Gamma_{\nabla}\{\mathbf{L}_0^{-1}\mathbf{q}\} = \mathbf{0}$. Thus, we obtain

$$\Gamma_{\nabla}\{\mathbf{L}_0^{-1}\mathbf{p}\} = -\tilde{\mathbf{g}} = -\mathbf{g} + \bar{\mathbf{g}}. \quad (42)$$

Abbreviating $\Gamma\{\square\} = \Gamma_{\nabla}\{\mathbf{L}_0^{-1}\{\square\}\}$, this can be rearranged for $\mathbf{g} = \bar{\mathbf{g}} - \Gamma\{\mathbf{p}\}$ and inserted back into the Eq. (37) for $\mathbf{p}$,

$$\mathbf{p} = \Delta \mathbf{L}\bar{\mathbf{g}} - \Delta \mathbf{L}\Gamma\{\mathbf{p}\}. \quad (43)$$

For the sake of brevity, we define $\mathbf{L} - \mathbf{L}_0 = \Delta\mathbf{L}$. This system of equations can be solved for $\mathbf{p}$

$$(\mathbf{I} + \Delta\mathbf{L}\Gamma)\{\mathbf{p}\} = \Delta\mathbf{L}\bar{\mathbf{g}}, \quad (44)$$

$$\mathbf{p} = (\mathbf{I} + \Delta\mathbf{L}\Gamma)^{-1} \{\Delta\mathbf{L}\bar{\mathbf{g}}\}. \quad (45)$$

The inverse is not a matrix or tensor inverse, but the inverse action of the operator in the round brackets, which would be the integral with the operator's corresponding Green function. The inverse of the pure Γ -operators clearly does not exist, since they project some part onto zero. But with the identity $\mathbf{I}$ inside the round bracket and the flexibility to choose l_0 as we please we can always circumvent singular operators. We now extract the mean value by applying $\Gamma_0\{\square\}$

$$\bar{\mathbf{p}} = \Gamma_0\{(\mathbf{I} + \Delta\mathbf{L}\Gamma)^{-1} \{\Delta\mathbf{L}\bar{\mathbf{g}}\}\}. \quad (46)$$

In the next step, we insert the result into Eq. (40) and factor out $\bar{\mathbf{g}}$ as it does not depend on $\mathbf{x}$

$$\bar{\mathbf{q}} = (\Gamma_0\{(\mathbf{I} + \Delta\mathbf{L}\Gamma)^{-1} \{\Delta\mathbf{L}\}\} + \mathbf{L}_0)\bar{\mathbf{g}}. \quad (47)$$

Therefore, the effective conductivity is

$$\mathbf{L}^* = \mathbf{L}_0 + \Gamma_0\{(\mathbf{I} + \Delta\mathbf{L}\Gamma)^{-1} \{\Delta\mathbf{L}\}\} \quad (48)$$

$$= \mathbf{L}_0 + \underbrace{\langle (\mathbf{I} + \Delta\mathbf{L}\Gamma)^{-1} \{\Delta\mathbf{L}\} \rangle}_{\mathbf{L}_*}. \quad (49)$$

The action of the Γ_0 -operator is to just take the volume average, which is usually denoted with $\langle \square \rangle$. In the Fourier space, this is done by setting all Fourier coefficients to zero except the one that belongs to $\mathbf{k} = \mathbf{o}$. We can easily determine $\mathbf{L}^*$ once we find the auxiliary field $\mathbf{L}_*$, which has no direct physical interpretation. Instead of inverting the operator $\mathbf{I} + \Delta\mathbf{L}\Gamma$, we consider the solution of the linear system in the Fourier space:

$$(\mathbf{I} + \Delta\mathbf{L}\Gamma)\{\mathbf{L}_*\} = \Delta\mathbf{L}, \quad (50)$$

$$\mathbf{L}_* + \Delta\mathbf{L}\Gamma\{\mathbf{L}_*\} = \Delta\mathbf{L}, \quad (51)$$

$$\sum_{\forall \mathbf{a}} e^{i\mathbf{x} \cdot \mathbf{a}} \hat{\mathbf{L}}_{*\mathbf{a}} + \sum_{\forall \mathbf{b}} e^{i\mathbf{x} \cdot \mathbf{b}} \Delta\hat{\mathbf{L}}_{\mathbf{b}} \cdot \sum_{\forall \mathbf{k} \neq \mathbf{o}} e^{i\mathbf{x} \cdot \mathbf{k}} l_0^{-1} \mathbf{K} \cdot \hat{\mathbf{L}}_{*\mathbf{k}} = \sum_{\forall \mathbf{c}} e^{i\mathbf{x} \cdot \mathbf{c}} \Delta\hat{\mathbf{L}}_{\mathbf{c}}, \quad (52)$$

$$\sum_{\forall \mathbf{a}} e^{i\mathbf{x} \cdot \mathbf{a}} \hat{\mathbf{L}}_{*\mathbf{a}} + \sum_{\forall \mathbf{b}, \forall \mathbf{k} \neq \mathbf{o}} e^{i\mathbf{x} \cdot (\mathbf{b} + \mathbf{k})} \Delta\hat{\mathbf{L}}_{\mathbf{b}} \cdot l_0^{-1} \mathbf{K} \cdot \hat{\mathbf{L}}_{*\mathbf{k}} = \sum_{\forall \mathbf{c}} e^{i\mathbf{x} \cdot \mathbf{c}} \Delta\hat{\mathbf{L}}_{\mathbf{c}}. \quad (53)$$

The second summand in the last equation represents a convolution. Because we have discrete periodic data in real space, the Fourier data is as well discrete and periodic. Hence, the summation is only carried out over one unit cell in the Fourier space, as depicted in Fig. 1.

A comparison of coefficients w.r.t. the function basis $e^{i\mathbf{x} \cdot \mathbf{c}}$ in Eq. (53) requires $\mathbf{b} = \mathbf{c} - \mathbf{k}$ and $\mathbf{a} = \mathbf{c}$ and yields

$$\hat{\mathbf{L}}_{*\mathbf{c}} + \sum_{\forall \mathbf{k} \neq \mathbf{o}} \Delta\hat{\mathbf{L}}_{\mathbf{c}-\mathbf{k}} \cdot l_0^{-1} \mathbf{K} \cdot \hat{\mathbf{L}}_{*\mathbf{k}} = \Delta\hat{\mathbf{L}}_{\mathbf{c}} \quad \forall \mathbf{c}, \quad (54)$$

which constitutes a linear system for the unknown Fourier coefficients $\hat{\mathbf{L}}_{*\mathbf{c}}$. One can see that this formulation has distinct mathematical advantages that can be directly exploited: The principal diagonal contains a sum of $\hat{\mathbf{L}}_{*\mathbf{c}}$ and a product with coefficients that should decay at high frequencies. *Most important, we are only interested in the homogeneous part of the solution, i.e., the $\hat{\mathbf{L}}_{*\mathbf{o}}$ -component, see Eq. (49). This fact allows us to apply an efficient locally synchronous linear solver.*

The presented system of equations is rarely set up and solved. Mostly, a series expansion for $\mathbf{L}^*$ is used, see e.g., Chap. 14 in Ref. [13] or Sect. 11.2 in Ref. [20]. Then, the problem is addressed in the real space, which involves correlation functions of the microstructure and integrals over Green's functions. The latter integrals are divergent, hence a renormalization is necessary [21]. Even if this issue can be avoided, the n th order coefficients involve integrals over n -point correlation functions, which are quite tedious to obtain. Moreover, since one can control by the choice of $\mathbf{L}_0$ whether a series alternates or converges from above or below, series expansions can give bounds for the effective properties.

As mentioned in the introduction (Sect. 1), other approaches solve for the gradient field $\mathbf{g}(\mathbf{x})$, which appears in an integral equation that has the same form as the Lippmann–Schwinger equation for quantum scattering. It is obtained by inserting Eq. (38) into Eq. (44). In this case, iterative methods to determine $\mathbf{g}(\mathbf{x})$ such as the alternating Real–Fourier space iterator [2,3] can be exploited. In this approach, several specific boundary problems must be solved if no symmetries are present. A variety of improvements and derivations of this method has been published, see e.g., Refs. [4,22].

In the next section, we will discuss a new type of iterative solver for systems of linear equations which will be used to determine $\mathbf{L}_*$ and ultimately $\mathbf{L}^* = \mathbf{L}_0 + \mathbf{L}_*$. This particular approach is capable of efficiently computing individual components of the entire solution vector which is in line with the proposed Fourier method, where only the homogeneous part of the solution is of interest to us.

4 Synchronous local algorithm for the iterative solution of linear systems of equations

In Sect. 3, it has been shown that analytical homogenization techniques lead to large linear systems despite the fact that only a few components of the result vector are of interest.

Fig. 1 Schematic representation of Eq. (53) for a function on $\mathbb{R}^2$ with 5×5 Fourier coefficients

Therefore, direct solvers are ill-suited to obtain solutions for such homogenization problems. In the following, we make use of a local solution algorithm that is capable of approximating only a few components of the solution vector. To achieve this goal, the sparsity of the coefficient (system) matrix has to be exploited. In the algorithm that is discussed in the following only the i th component of the solution to a system of linear equations is iteratively computed based on a Neumann series representation.

4.1 Motivation

The starting point for our discussion of the local iterative solver is a generic system of linear equations

$$\mathbf{Ax} = \mathbf{b}, \quad (55)$$

where $\mathbf{A}$ is a non-singular, real, square matrix of dimension n and $\mathbf{b}$ is the right-hand side vector of the system. Equation (55) can be solved by a wide variety of methods that can be classified into two major categories: (i) direct and (ii) indirect or iterative solvers. Typical representatives of the first class of methods are Gaussian elimination, LU- or Cholesky factorization to name just a few [23]. Iterative methods include stationary linear approaches (which will be discussed in more detail in Sect. 4.2) such as the Jacobi, Gauss–Seidel, or Richardson methods and gradient methods (conjugate gradients, preconditioned conjugate gradients, steepest descent methods) [24–26]. All of these methods have in common that the *entire* solution vector $\mathbf{x}$ is computed either exactly or in an approximate fashion.

To compute all components of vector $\mathbf{x}$ seems to be rather inefficient and a waste of computational resources if only a few components are actually of interest. It is easy to see that an iterative method that only computes selected components is much more efficient than a direct solution of the entire system of linear equations. For example, an LU-factorization of

a fully populated matrix requires $2n^3/3$ floating-point operations (additions and multiplications), where n denotes the size of the system. The iteration procedure with fully populated matrices and residual requires $2n^2$ per iteration [27]. We can expect, however, that much less than n iterations will be needed. Since only a few components of the solution vector are of interest in our analytical homogenization process, detailed in Sect. 3, we need to identify promising methods that are capable of computing individual components of the solution vector only. Therefore, the question regarding how this problem can be tackled. One possibility, based on simple stationary linear iterative methods, has been proposed by Lee et al. in a series of reports [27–29] and is discussed in the remainder of this section.

4.2 Stationary linear iterative methods

In order to keep the algorithm of the locally synchronous solution procedure as simple as possible, only stationary linear schemes are included in the discussion. In this contribution, we make use of different iterative solution schemes such as the (i) Jacobi, (ii) Gauss–Seidel, (iii) successive over-relaxation, and (iv) Richardson methods. The discussion of these methods closely follows Refs. [24–26]. The point of departure to derive algorithms for these methods is Eq. (55) which can be re-written as:

$$\mathbf{x}^{(i+1)} = \mathbf{G}\mathbf{x}^{(i)} + \mathbf{z}, \quad (56)$$

where $\mathbf{G}$ denotes the iteration matrix, $\mathbf{z}$ the auxiliary solution vector, and the superscript $\square^{(i)}$ represents the i th step in the iteration. To ensure convergence of the results a necessary condition is

$$\rho(\mathbf{G}) < 1, \quad (57)$$

i.e., the spectral radius of the iteration matrix must be less than unity¹. This can be guaranteed if the system matrix $\mathbf{A}$ is positive-definite and/or diagonally dominant. Considering the matrix $\mathbf{M}$, being introduced in Sect. 3, inequality (57) holds and consequently, iterative solvers are applicable.

In general, each matrix $\mathbf{A}$ can be additively split into a sum of a strictly lower triangular matrix $\mathbf{L}$, a strictly upper triangular matrix $\mathbf{U}$, and a diagonal matrix $\mathbf{D}$

$$\mathbf{A} = \mathbf{D} + \mathbf{L} + \mathbf{U}. \quad (58)$$

The solution to the system of equations (55) can be obtained iteratively by means of Eq. (56) which can be re-written using the matrix $\tilde{\mathbf{M}}$ which serves as a pre-conditioner of the system matrix $\mathbf{A}$:

$$\mathbf{x}^{(i+1)} = \underbrace{(\mathbf{I} - \tilde{\mathbf{M}}^{-1}\mathbf{A})}_{\mathbf{G}} \mathbf{x}^{(i)} + \underbrace{\tilde{\mathbf{M}}^{-1}\mathbf{b}}_{\mathbf{z}}, \quad (59)$$

with $\mathbf{I}$ being the unity matrix. Obviously, different choices for $\mathbf{G}$ and $\mathbf{z}$ are possible and will be addressed in the following, see Sects. 4.2.1 to 4.2.4.

4.2.1 Richardson method

The Richardson iteration technique is perceivably the simplest possible method and the prototype of any linear iteration. In this case, we choose the matrix $\tilde{\mathbf{M}}$ to be a multiple of the unity matrix, i.e.,

$$\tilde{\mathbf{M}}_{\mathbf{R}} = \gamma \mathbf{I}. \quad (60)$$

Therefore, the resulting iterative scheme is

$$\mathbf{x}_{\mathbf{R}}^{(i+1)} = \mathbf{x}_{\mathbf{R}}^{(i)} + \frac{1}{\gamma} (\mathbf{b} - \mathbf{A}) \mathbf{x}_{\mathbf{R}}^{(i)}. \quad (61)$$

In this method, optimal convergence is achieved for a γ -value of

$$\gamma_{\mathbf{R},\text{opt}} = \frac{1}{2} (\lambda_{\max} - \lambda_{\min}), \quad (62)$$

where $\lambda_{\max}$ and $\lambda_{\min}$ denote the largest and smallest eigenvalues of the iteration matrix $\mathbf{G}_{\mathbf{R}} = \mathbf{I} - 1/\gamma \mathbf{A}$. This choice minimizes the spectral radius $\rho(\mathbf{G}_{\mathbf{R}})$ of the iteration matrix and thus, maximizes the rate of convergence of the iterative method. If the coefficient matrix $\mathbf{A}$ is symmetric and positive-definite and the Richardson method with an optimal choice

of γ is used the spectral radius of the iteration matrix can be computed as

$$\rho(\mathbf{G}) = \frac{\kappa(\mathbf{A}) - 1}{\kappa(\mathbf{A}) + 1} \quad (63)$$

indicating that ill-conditioned coefficient matrices (here $\kappa(\square)$ denotes the condition number) of linear systems of equations result in a slow convergence. This obviously limits the applicability for certain areas of application, e.g., large finite element systems.

4.2.2 Jacobi method

The Jacobi method is a simple iterative method that is used to compute the solution of linear systems of equations with a positive-definite *and* strictly diagonally dominant system matrix $\mathbf{A}$, i.e.,

$$\mathbf{x}^T \mathbf{A} \mathbf{x} > 0 \quad \text{and} \quad |a_{ii}| > \sum_{j \neq i} |a_{ij}|. \quad (64)$$

In this case, it is sufficient to choose the following set-up:

$$\tilde{\mathbf{M}}_{\mathbf{J}} = \mathbf{D} = \text{diag}(\mathbf{A}). \quad (65)$$

By substituting Eq. (65) into Eq. (59), we obtain the iteration relation for the Jacobi method as:

$$\mathbf{x}_{\mathbf{J}}^{(i+1)} = \mathbf{D}^{-1} (\mathbf{b} - (\mathbf{L} + \mathbf{U}) \mathbf{x}_{\mathbf{J}}^{(i)}). \quad (66)$$

One important aspect to keep in mind is that the Jacobi method does not converge for every symmetric positive-definite matrix.

In order to improve the comparably poor convergence of the Jacobi method an additional weighting parameter ω can be introduced

$$\tilde{\mathbf{M}}_{\mathbf{wJ}} = \omega \mathbf{D}. \quad (67)$$

Considering the special case of a symmetric and positive-definite coefficient matrix $\mathbf{A}$, it can be shown that the weighted Jacobi method converges if

$$0 < \omega < \frac{1}{2} \rho(\mathbf{D}^{-1} \mathbf{A}) \quad (68)$$

[24] and the optimal choice of ω is provided for

$$\omega_{\mathbf{wJ},\text{opt}} = \frac{1}{2} (\tilde{\lambda}_{\max} - \tilde{\lambda}_{\min}), \quad (69)$$

where $\tilde{\lambda}_{\max}$ and $\tilde{\lambda}_{\min}$ are the largest and smallest eigenvalues of the matrix $\mathbf{D}^{-1} \mathbf{A}$, respectively. A typical choice for the

¹ The spectral radius of a matrix $\mathbf{B}$ is defined as: $\rho(\mathbf{B}) = \max\{|\lambda| : \lambda \in \sigma(\mathbf{B})\}$, where $\sigma(\mathbf{B})$ denotes the spectrum of $\mathbf{B}$, i.e., the set of all eigenvalues of $\mathbf{B}$.

weighting parameter ω is $3/2$ which saves effort in computing the eigenvalues of the system. At this point we can derive the iteration expression as

$$\mathbf{x}_{\text{wJ}}^{(i+1)} = \frac{1}{\omega} \mathbf{D}^{-1} \left(\mathbf{b} - (\mathbf{L} + \mathbf{U}) \mathbf{x}_{\text{wJ}}^{(i)} \right) + \mathbf{I}(1 - \omega) \mathbf{x}_{\text{wJ}}^{(i)}. \quad (70)$$

4.2.3 Gauss–Seidel method

The idea of the Gauss–Seidel iterative solver is closely related to that of the Jacobi method (see Sect. 4.2.2). In contrast to the Jacobi method, each component of the new iterate depends on all previously computed components. Therefore, the following expression is used to set up the iteration:

$$\tilde{\mathbf{M}}_{\text{GS}} = \mathbf{D} + \mathbf{L}. \quad (71)$$

As in the previous section, Eq. (71) is substituted into Eq. (59) and thus the solution for each iteration can be computed as

$$\mathbf{x}_{\text{GS}}^{(i+1)} = (\mathbf{D} + \mathbf{L})^{-1} \left(\mathbf{U} \mathbf{x}_{\text{GS}}^{(i)} + \mathbf{b} \right). \quad (72)$$

Note that the Gauss–Seidel method is applicable to linear systems if the coefficient matrix $\mathbf{A}$ is either strictly diagonally dominant or symmetric positive-definite. This method generally converges faster than the conventional Jacobi method, i.e., without weighting procedure, as presented in Sect. 4.2.2.

4.2.4 Successive over-relaxation

The successive over-relaxation (SOR) method can be directly derived from the Gauss–Seidel scheme, see Sect. 4.2.3, by extrapolation. To this end, the weighted average of the solution of the previous iteration step $\mathbf{x}_{\text{SOR}}^{(i)}$ and the solution of the Gauss–Seidel scheme for the current iteration $\mathbf{x}_{\text{GS}}^{(i+1)}$

$$\mathbf{x}_{\text{SOR}}^{(i+1)} = \omega \mathbf{x}_{\text{GS}}^{(i+1)} + (1 - \omega) \mathbf{x}_{\text{SOR}}^{(i)} \quad (73)$$

is taken. To derive the iteration sequence for the SOR scheme, a different splitting of the coefficient matrix $\mathbf{A}$ compared to the one introduced in Sect. 4.2 is used. Instead, over-relaxation is based on

$$\omega \mathbf{A} = (\mathbf{D} + \omega \mathbf{L}) + (\omega \mathbf{U} + (\omega - 1) \mathbf{D}). \quad (74)$$

In the next step, Eq. (74) is substituted into Eq. (55) and we obtain the following relation to compute the solution iteratively

$$\mathbf{x}_{\text{SOR}}^{(i+1)} = (\mathbf{D} + \omega \mathbf{L})^{-1} \left(\omega \mathbf{b} - [\omega \mathbf{U} + (\omega - 1) \mathbf{D}] \mathbf{x}_{\text{SOR}}^{(i)} \right). \quad (75)$$

We can reach the same solution by defining $\tilde{\mathbf{M}}$ as

$$\tilde{\mathbf{M}}_{\text{SOR}} = \left(\frac{1}{\omega} \mathbf{D} + \mathbf{L} \right). \quad (76)$$

At this point, the question regarding the optimal choice of the extrapolation/relaxation factor ω remains. It can be shown that the optimal value is

$$\omega_{\text{SOR,opt}} = \frac{2}{1 + \sqrt{1 - \mu^2}}, \quad (77)$$

where μ is the spectral radius of the iteration matrix of the Jacobi method $\mathbf{G}_J$

$$\mu = \rho(\mathbf{G}_J) \quad \text{and} \quad \mathbf{G}_J = \mathbf{I} - \mathbf{D}^{-1} \mathbf{A}. \quad (78)$$

It is straightforward to prove that for $\omega = 1$ the SOR is identical to the Gauss–Seidel method. Depending on the problem that is solved, the SOR method might be significantly faster than the Gauss–Seidel method. This has, however, not been observed for the analytical homogenization which is our main concern.

An extension of the SOR method is the symmetric successive over-relaxation (SSOR) scheme requiring a symmetric coefficient matrix $\mathbf{A}$. One step in this approach consists of a standard (forward) SOR computation followed by a backward one. To this end, the approach to splitting the coefficient matrix $\mathbf{A}$ presented in Eq. (74) is adjusted

$$\omega \mathbf{A} = (\mathbf{D} + \omega \mathbf{U}) + (\omega \mathbf{L} + (\omega - 1) \mathbf{D}) \quad (79)$$

and therefore, $\tilde{\mathbf{M}}$ is now defined as

$$\tilde{\mathbf{M}}_{\text{bSOR}} = \left(\frac{1}{\omega} \mathbf{D} + \mathbf{U} \right). \quad (80)$$

Thus, the backward SOR step takes the following form

$$\mathbf{x}_{\text{bSOR}}^{(i+1)} = (\mathbf{D} + \omega \mathbf{U})^{-1} \left(\omega \mathbf{b} - [\omega \mathbf{L} + (\omega - 1) \mathbf{D}] \mathbf{x}_{\text{bSOR}}^{(i)} \right). \quad (81)$$

The equations for the SSOR method can now be derived by substituting the solution obtained with Eq. (75) into the right-hand side of Eq. (81). The final iteration expression can be set up using

$$\tilde{\mathbf{M}}_{\text{SSOR}} = \frac{1}{\omega(2 - \omega)} (\mathbf{D} + \omega \mathbf{L}) \mathbf{D}^{-1} (\mathbf{D} + \omega \mathbf{L})^T. \quad (82)$$

The iteration relation for the SSOR scheme is only given in the form of Eq. (59) due to the length of the resulting

expression

$$\mathbf{x}_{\text{SSOR}}^{(i+1)} = (\mathbf{I} - \tilde{\mathbf{M}}_{\text{SSOR}}^{-1} \mathbf{A}) \mathbf{x}^{(i)} + \tilde{\mathbf{M}}_{\text{SSOR}}^{-1} \mathbf{b}. \quad (83)$$

A suitable value for ω is still computed by means of Eq. (77). Nowadays, the SSOR scheme is primarily used as a preconditioner for other iterative schemes that are suitable for symmetric matrices as its convergence is usually slower than that of the standard SOR approach.

Note that the optimal choice of the iterative method, or in other words, the optimal choice of the iteration matrix $\mathbf{G}$ and the auxiliary solution vector $\mathbf{z}$ are highly depend on the problem under consideration. Thus, for each method different properties of the system matrix $\mathbf{A}$ can be expected and these need to be accounted for.

4.3 Locally synchronous approximation of a single component of the solution vector

The synchronous algorithm is a modification of standard stationary linear iterative methods using an observation which allows the vectors involved in the computation to remain sparse when the iteration matrix $\mathbf{G}$ is sparse (with a maximum of d nonzero components; $d \ll n$, $1/(1 - \|\mathbf{G}\|_2)$ is sufficiently small, and n (number of equations) is large [27]. As solving a system of linear equations is fundamentally a problem involving the full matrix, it is not a trivial task to design an approach that extracts only a single component. The idea used in this contribution for the analytical (Fourier-based) homogenization is based on the works of Lee et al. [27–29]. Without loss of generality (considering the intended applications), the following discussions are limited to (i) positive definite and/or (ii) strictly diagonally dominant coefficient matrices $\mathbf{A}$. In the first case, it is ensured that

$$\|\mathbf{G}\|_2 \leq \rho(\mathbf{G}) < 1 \quad (84)$$

holds, while in the latter case

$$\|\mathbf{G}\|_2 = \rho(\mathbf{G}) < 1. \quad (85)$$

is valid for symmetric iteration matrices.

4.3.1 Neumann series representation

All the iterative methods, introduced in Sects. 4.2.1 to 4.2.4, have in common that their primary goal is to approximate the leading term of the Neumann series

$$\mathbf{x} = \sum_{k=0}^{\infty} \mathbf{G}^k \mathbf{z}. \quad (86)$$

As mentioned before convergence of this series is only ensured if the spectral radius of the iteration matrix $\rho(\mathbf{G})$ is below one. The smaller this value is the faster the iteration converges, i.e., less steps are needed to fall below a prescribed error threshold. Different possible choices of $\mathbf{G}$ and $\mathbf{z}$ have been discussed above and can now be used to construct a local algorithm that does not solve for the entire vector $\mathbf{x}$ but only for a few components x_i that are of interest due to various reasons depending on the intended area of application.

4.3.2 Extraction of a single component

The Neumann series representation of the solution vector $\mathbf{x}$, given in Eq. (86), provides the entire solution for the system of linear equations, while we are only interested in a single component x_i . To extract the component of interest, we simply use the corresponding basis vector (unit vector) of the n -dimensional space which has a value of 1 at the coordinate i and 0 everywhere else

$$x_i = \mathbf{e}_i^T \mathbf{x} \quad \text{with } \mathbf{e}_i = [0_{(1)}, 0_{(2)}, \dots, 0_{(i-1)}, 1_{(i)}, 0_{(i+1)}, \dots, 0_{(n)}]^T, \quad (87)$$

where the subscript $\square_{(k)}$ denotes the index of the component. In the next step, we left-multiply Eq. (86) by $\mathbf{e}_i^T$ and obtain

$$x_i = \left[\sum_{k=0}^{\infty} (\mathbf{e}_i^T \mathbf{G}^k) \right] \mathbf{z}. \quad (88)$$

Thus, the vector $\mathbf{z}$ is only multiplied once at the end of the summation process. Since the Neumann series is an infinite series it has to be truncated after a certain number of steps t and therefore, the equation is split into two parts

$$x_i = \left[\sum_{k=0}^{t-1} (\mathbf{e}_i^T \mathbf{G}^k) \right] \mathbf{z} + \mathbf{e}_i^T \sum_{k=t}^{\infty} \mathbf{G}^k \mathbf{z} = \left[\sum_{k=0}^{t-1} (\mathbf{e}_i^T \mathbf{G}^k) \right] \mathbf{z} + \mathbf{e}_i^T \mathbf{G}^t \mathbf{x}. \quad (89)$$

4.3.3 Algorithm

The complete algorithm including the initialization stage and the update procedure are summarized in Algorithm 1. For the sake of a compact notation, we introduce the residual vector $\mathbf{r}^{(t)}$

$$\mathbf{r}^{(t)} = (\mathbf{e}_i^T \mathbf{G}^t)^T \quad (90)$$

and the estimate vector

$$\mathbf{p}^{(t)} = \left(\sum_{k=0}^{t-1} (\mathbf{e}_i^T \mathbf{G}^k) \right)^T. \quad (91)$$

Thus, the iteration expression is

$$x_i = \mathbf{p}^{(t)T} \mathbf{z} + \mathbf{r}^{(t)T} \mathbf{x}. \quad (92)$$

Algorithm 1 computes the estimate according to

$$\hat{x}_i = \mathbf{p}^{(t)T} \mathbf{z} \quad (93)$$

and thus, the error is given by

$$x_i - \hat{x}_i = \mathbf{r}^{(t)T} \mathbf{x}. \quad (94)$$

That is also the reason why $\mathbf{r}^{(t)}$ is coined residual vector as indicated above. Generally, it is observed that the method terminates within at most

$$n_{\text{iter}} = \frac{\ln(\epsilon)}{\ln(\|\mathbf{G}\|_2)} \quad (95)$$

steps (iterations), and achieves an estimate such that

$$|\hat{x}_i - x_i| \leq \epsilon \|\mathbf{x}\|_2 \quad (96)$$

holds, where ϵ is a prescribed error threshold.

For us, the key property to exploit is the sparsity of the linear system, in conjunction with the damping effect of the iteration matrix on the residual: The initial iteration vector is zero everywhere except for the desired solution component (see Algorithm 1), for which it is one. Hence, a large number of products does not need an explicit evaluation. However, during the iterative process, the vector gets populated, hence more products need to be carried out. Depending on the

Algorithm 1 Algorithm for the locally synchronous iterative solver.

```

1: function LOCALLYSYNCHRONOUSITERATIVE SOLVER( $\mathbf{G}, \mathbf{z}, i, \epsilon$ )
2:   % Input parameter:
3:   %  $\mathbf{G}$ :                                ▷ Iteration matrix
4:   %  $\mathbf{z}$ :                                ▷ Transforms right-hand side vector
5:   %  $i$ :                                ▷ Component of the solution vector
6:   %  $\epsilon$ :                                ▷ Error threshold
7:    $\mathbf{p}^0 = \mathbf{0}; \mathbf{r}^0 = \mathbf{e}_i; t = 0;$ 
8:   while  $\|\mathbf{r}^{(t)}\|_2 \geq \epsilon$  do
9:      $\mathbf{p}^{(t+1)} = \mathbf{p}^{(t)} + \mathbf{r}^{(t)};$ 
10:     $\mathbf{r}^{(t+1)} = \mathbf{G}^T \mathbf{r}^{(t)};$ 
11:     $t = t + 1;$ 
12:   end while
13:   return  $\hat{x}_i = \mathbf{p}^{(t)T} \mathbf{z};$  ▷  $\hat{x}_i$ : Approximation of the  $i$ th component
    of the solution vector
14: end function

```

iteration matrix, the new entries are damped, or propagate through the residual vector. It turns out that the investigated linear system, resulting from the analytical homogenization approach discussed in Sect. 3, is appropriate for this iterative solver. When the iteration matrix $\mathbf{G}$ is sufficiently sparse and its spectral properties are well-behaved, i.e., $d = O(1)$ and $-1/\ln(\|\mathbf{G}\|_2) = O(1)$, then we achieve an approximation for x_i in constant time with respect to the size of the coefficient matrix for large n . For a more detailed discussion of the properties of this algorithm the reader is referred to the original work by Lee et al. [27–29].

The Richardson and Jacobi methods are particularly suitable for the local iterative solver if the coefficient matrix $\mathbf{A}$ is sparse as the iteration matrix $\mathbf{G}$ inherits this property. This will also lead to sparse vector $\mathbf{z}$. Note that the approximation of the leading terms of the Neumann series will be at least as dense as $\mathbf{z}$ resulting in an increasing computational burden if the vector is fully populated or fill-in takes place. According to Ref. [27] bounds on the convergence rates and convergence rate guarantees can be provided if $\|\mathbf{G}\|_2 < 1$ holds.

In order to apply the different iterative algorithms in conjunction with the concept of the locally synchronous solver, we only need to exchange the definitions of the auxiliary iteration variables $\tilde{\mathbf{M}}$ and $\mathbf{z}$ in the procedure given in Algorithm 1. For the sake of convenience, the definitions of these variables are listed in Table 2 including a reference to the corresponding sections and equations.

5 Results

In this section we assess the performance of the presented locally synchronous iterative solver (see Sect. 4.3) in comparison to the default direct and Krylov subspace iteration methods provided in the computer algebra system (CAS) Mathematica.

5.1 Heat conduction problem

Since micro-structural data are generally provided on a Cartesian lattice, the linear system for $\mathbf{L}_\star$ (see Eq. 54) is easily obtained with the help of the discrete Fourier transform (DFT). It provides as many data points in the Fourier space as are used in real space. As an example we consider a two-phase material. We seek the effective heat conductivity $\mathbf{L}^*$ that relates the effective heat flux $\bar{\mathbf{q}}$ to the temperature gradient $\bar{\mathbf{g}}$:

$$\bar{\mathbf{q}} = -\mathbf{L}^* \cdot \bar{\mathbf{g}}. \quad (97)$$

Table 2 Stationary linear iterative methods used for the locally synchronous solver

Method	Auxiliary Iteration Variables: $\tilde{\mathbf{M}}, \mathbf{z}$ $\mathbf{G} = \mathbf{I} - \tilde{\mathbf{M}}^{-1}\mathbf{A}, \mathbf{z} = \tilde{\mathbf{M}}^{-1}\mathbf{b}$	Reference Eq./Sect.
Richardson	$\tilde{\mathbf{M}}_{\text{R}} = \gamma \mathbf{I}$	Eq. (60)/Sect. 4.2.1
Jacobi	$\tilde{\mathbf{M}}_{\text{J}} = \mathbf{D}$	Eq. (65)/Sect. 4.2.2
weighted Jacobi	$\tilde{\mathbf{M}}_{\text{wJ}} = \omega \mathbf{D}$	Eq. (67)/Sect. 4.2.2
Gauss–Seidel	$\tilde{\mathbf{M}}_{\text{GS}} = \mathbf{D} + \mathbf{L}$	Eq. (71)/Sect. 4.2.3
SOR	$\tilde{\mathbf{M}}_{\text{SOR}} = \left(\frac{1}{\omega} \mathbf{D} + \mathbf{L} \right)$	Eq. (76)/Sect. 4.2.4
SSOR	$\tilde{\mathbf{M}}_{\text{SSOR}} = \frac{1}{\omega(2-\omega)} (\mathbf{D} + \omega \mathbf{L}) \mathbf{D}^{-1} (\mathbf{D} + \omega \mathbf{L})^T$	Eq. (82)/Sect. 4.2.4

Locally, we have

$$\mathbf{q}(\mathbf{x}) = -\mathbf{L}_i(\mathbf{x}) \cdot \mathbf{g}(\mathbf{x}), \quad i \in \{1, 2\} \quad (98)$$

in phases 1 and 2, respectively. The second law of thermodynamics requires the heat to flow from regions with higher temperature to those with lower temperature, hence the inequality $\mathbf{q}(\mathbf{x}) \cdot \mathbf{g}(\mathbf{x}) < 0$ holds everywhere. We incorporate this negativity in the above equations explicitly, such that we can work with positive definite conductivities $\mathbf{L}_i$ and thus, a positive definite $\mathbf{L}^*$.

5.2 Verification problem: Laminate structure

To check the implementation, a simple reference problem, i.e., a laminate structure with the isotropic local conductivities

$$\mathbf{L}_i = l_i \mathbf{I}, \quad l_1 = 1, \quad l_2 = 2, \quad (99)$$

has been considered. This example has been chosen as a closed-form solution is available. Parallel to the plane the laminate the effective conductivity is the arithmetic mean value (Voigt average) of the conductivities l_1 and l_2 , while in other planes perpendicular to the lamination plane the effective conductivity is the harmonic mean (Reuss average) of l_1 and l_2

$$\mathbf{L}^* = \begin{bmatrix} v_1 l_1 + v_2 l_2 & & \\ & v_1 l_1 + v_2 l_2 & \\ & & (v_1 l_1^{-1} + v_2 l_2^{-1})^{-1} \end{bmatrix} \mathbf{e}_i \otimes \mathbf{e}_j, \quad (100)$$

where the lamination direction is $\mathbf{e}_3$. The volume fractions $v_1 + v_2 = 1$ correspond to the plies' thicknesses.

As reference conductivity we choose $l_0 = 1.5$. The solution quality only depends on the ability of the discretization to capture the actual volume fractions. In cases where discretization exactly resolve the material distribution, e.g.,

Fig. 2 Sketch of a laminate microstructure with Schematic representation of Eq. (53) for a function on $\mathbb{R}^2$ with 5×5 Fourier coefficients

when the volume fractions are $v_1 = 3/5$ and $v_2 = 2/5$ and a $5 \times 5 \times 5$ discretization of the unit cells is used, the exact results $3/5 + 2 \times 2/5 = 7/5$ and $(3/5 + 1/2 \times 2/5)^{-1} = 5/4$ are obtained.

5.3 Complex microstructure

As the second example, we consider a microstructure whose effective conductivity has to be computed numerically as no analytical solution can be derived. To examine the numerical properties of the worst possible case, we remove any periodicity, material symmetry, coordinate independence (as in the laminate) or symmetric alignment. The phase distribution is generated by periodically repeating an ellipsoid that is larger than the unit cell, see Fig. 4. The ellipsoid which is cut at the boundaries of the unit cell (dimensions: $-1/2 \leq x \leq 1/2$,

Fig. 3 Magnitude of the Fourier coefficients. The coefficients are zero for all wave vectors that are not perpendicular to the plane of lamination, and the higher order Fourier coefficients become smaller as the discretization is increased from $d = 5$ (left) over $d = 15$ (center) to $d = 45$ (right)

$1/2 \leq y \leq 1/2, 1/2 \leq z \leq 1/2$.) is defined by the following expression

$$\left(\frac{1}{4}(-2x + y + 3z) - 1\right)^2 + \left(\frac{1}{4}(x + 3y) - 1\right)^2 + (z - 1)^2 \leq \frac{4}{25}. \quad (101)$$

The ellipsoid penetrates the neighbouring cells, where it overlaps with copies of itself. In this way, we obtain a complex microstructure that has two interpenetrating phases. No rotational symmetry is found in the arrangement of the microstructure, but still a cubic translational symmetry exists, which is unavoidable due to the periodicity of the Fourier base functions. The base vectors $\mathbf{e}_i$ are parallel to the edges of the the cubic unit cell. A discretized version of this microstructure is depicted along with the Fourier coefficients in Fig. 5. The local conductivities

$$\mathbf{L}_1 = \begin{bmatrix} 3 & 1 & 2 \\ 1 & 3 & 1 \\ 2 & 1 & 3 \end{bmatrix} \mathbf{e}_i \otimes \mathbf{e}_j, \quad (102)$$

$$\mathbf{L}_2 = \begin{bmatrix} 2 & 0 & 1 \\ 0 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix} \mathbf{e}_i \otimes \mathbf{e}_j \quad (103)$$

are both orthotropic and not aligned with the cubic periodicity frame of the unit cell. This setup is the most general case that can be considered in the given framework: For second order tensors, the smallest material symmetry group is the orthotropic group, and the microstructure has no rotational symmetry.

The reference heat conduction is chosen as $\mathbf{L}_0 = l_0 \mathbf{I}$ with $l_0 = 3$. This value is relatively close to the entries of $\mathbf{L}_1$ and $\mathbf{L}_2$. If one uses series expansions to estimate $\mathbf{L}^*$, the choice of $\mathbf{L}_0$ controls the rate of convergence. Here, the choice of l_0

Fig. 4 A complex microstructure by overlapping a periodically repeating ellipsoid

affects the condition of the linear system and the convergence of the iterative solver. Listing 1, provided in the appendix of this article, contains the Mathematica-code that generates the linear system to solve. It makes use of the build-in DFT algorithm. We have set up a system of $6 \times d^3$ equations with the edge discretization d (six because $\mathbf{L}_*$ has six independent components, d^3 is the number of points in the unit cell), but are only interested in six solution components, namely the homogeneous part of the $\mathbf{L}_*$ -field, see Eq. (49). However, standard solvers seek the complete solution and therefore, calculation times increase drastically with d . The computations have been performed on a desktop PC with an Intel(R) Core(TM) i7-7700 CPU (3.60GHz) and 32 GB

Fig. 5 Exemplary two-phase material

Fig. 6 Solver times over discretization for different solvers

of RAM. Mathematica's internal solvers, although dealing with sparse numerical data structures need considerable time to set up discretizations as small as $d = 13$. Calling `LinearSolve[]`² with the default behaviour, namely a direct solver, leads to significant computational costs. The solver times for different d -values ranging from 5 to 11 are compared in Fig. 6, while the corresponding convergence of the solution is visualized in Fig. 7.

From the first diagram we can infer that a solution time of approximately 20 minutes is needed for $d = 11$ when employing the internal direct solver, while only 8 minutes

Fig. 7 Convergence of $\|\mathbf{L}_{*d} - \mathbf{L}_{*14}\| / \|\mathbf{L}_{*14}\|$ as the discretization d is increased

are needed by the internal iterative solver. In comparison, the proposed locally synchronous solver (see Sect. 4.3) based on Jacobi-iterations (see Sect. 4.2.2) requires approximately 20 seconds for this problem which is more than 20 times faster. For $d > 11$, the internal solvers run into memory issues, which happens only at $d > 14$ for the locally synchronous solver. The system matrix for a discretization of d generally exhibits a sparsity of approximately 45%, i.e., 45% of the matrix components are nonzero, and therefore has $0.45(6d^3)^2$ entries. Even using sparse data formats, a full-fledged CAS is not an appropriate environment and therefore, tools that are more specialized on numeric computations are required. Nevertheless, the potential of the local solver becomes apparent.

² Depending on the properties of the coefficient matrix of the system of linear equations Mathematica chooses LAPACK routines for dense or a multifrontal direct solver for a sparse matrices.

Fig. 8 Structure of the complex coefficient matrix of size $2058=6 \cdot 7^3$

5.4 Properties of the linear system

A typical system matrix plot is given in Fig. 8. The matrix is unsymmetric and has approximately 40% of nonzero entries, i.e., it is far from being as sparse as the usual finite element or finite difference stiffness matrices. We found the sparsity to depend on the microstructure, the phases' properties, the choice of the reference conductivity l_0 and the discretization. Two extreme cases for the laminate microstructure are illustrated in Figs. 9 and 10, respectively, where the sparsity depends on the parity of d . Note that the right-hand side vector has approximately 94.4% of nonzero entries.

5.5 Comparison to other linear systems

We have applied the same approach for finite element problems, where we tried to solve only for a specific degree of freedom (DOF). It turned out that none of the methods mentioned in Sect. 4 yielded a versatile iterative method that is comparable in terms of performance to standard direct solver. In our studies we also included different pre-conditioning techniques (diagonal pre-conditioner; column pre-conditioner [30]; upper co-diagonal pre-conditioner [31], maximum upper triangular matrix pre-conditioner [32]; incomplete Cholesky decomposition) and algorithms to make the coefficient matrices (strictly) diagonally dominant [33] to no avail. The top priority for all algorithms was placed on a simple and efficient implementation. While the incomplete Cholesky pre-conditioner is comparably effective, it is too costly to be useful in the proposed method. Additionally, minimizing the bandwidth of the system matrix using the reverse Cuthill-McKee method does not signif-

Fig. 9 Matrix structure for $d = 6$ for the laminate structure from Sec. 5.2. Approximately 0.6% of all matrix-entries are non-zero

Fig. 10 Matrix structure for $d = 7$ for the laminate structure from Sec. 5.2. Approximately 38.8% of all matrix-entries are non-zero

icantly improve the overall performance. We believe that the cause for this is that the solutions of boundary value problems require the boundary DOFs to communicate, in a sense. Boundary conditions on one side of the computational domain affect the DOFs on the other side, hence the information needs to propagate through the linear system. Unless the preconditioning is a quasi-inversion of the linear system, the locally synchronous iterative method cannot easily cope with such problems. Here, more sophisticated iterative solvers such as the biconjugate gradient stabilized method

Fig. 11 Spectrum of the iteration matrix $\mathbf{G}$ for different reference conductivities l_0 for the test problem (Sec. 5.3) with $d = 6$. The largest eigenvalue and hence the rate of convergence depends on the choice of l_0

Fig. 12 Spectrum of the iteration matrix $\mathbf{G}$ for different phase contrasts for the lamellar microstructure (Sec. 5.2) with $d = 6$ and where $l_0 = l_1$. For $l_2/l_1 = 8$, the largest eigenvalue is already greater than 1

BiCGSTAB or multi-grid solver have to be employed and coupled with the local iterations (if possible).

On the other hand, in our application to homogenization based on Fourier methods we solve a linear system for Fourier coefficients, but are only interested in the leading order term. therefore, the higher the frequency gets, the smaller is the actual contribution of that coefficient. Hence, we observe a rapid damping of the contribution and the convergence of the local iterative method is ensured.

5.6 Limitations

At this point, important imitations of the presented approach need to be mentioned. This facilitates a general understanding of the locally synchronous method and enables one to choose suitable areas of application. There are basically two facts that need to be considered:

1. Analogous to the MS fixed point solver, high phase contrasts pose a problem for the convergence. We managed to obtain converged solutions for phase contrasts up to 10 by adjusting l_0 . When the phase contrasts becomes larger, the optimal l_0 leaves some eigenvalues close to 1 in the spectrum of the iteration matrix, see Fig. 11, which prevents a convergence of the iterative solver. This is also shown in Fig. 12 for the lamellar structure. For a phase contrast of $l_2/l_1 = 8$ and $l_0 = l_1$, the largest eigenvalue is already larger than 1. In the special case of the laminate, the iteration still converges because no residuum appears in direction of the largest eigenvalue during the iteration. However, in a more general setting, convergence will not be observed.

Adjusting the weight ω of the Jacobi-preconditioning can accelerate the convergence, but does not lead to convergence when the standard Jacobi method with $\omega = 1$ does not converge. It should be noted that this really is a problem of the solver: The linear system solution can be obtained for high phase contrasts with another linear solver. This flexibility is not available in case of the first kind of spectral solver.

2. The proposed approach is very memory intensive, which needs to be reduced in order to become a viable alternative to established numerical methods. It appears that the system matrix has a very specific structure, which may allow for an efficient compression. However, it is still questionable whether this is worth the effort.

6 Summary

Exploiting the orthogonality of rotation- and divergence-free fields algebraically in the Fourier space, we reformulate the difference between the effective conductivity and the reference conductivity as the solution of a large linear system in Fourier space, of which only the homogeneous solution components are needed, i.e., the zeroth Fourier coefficients. Therefore, instead of using a general purpose linear solver we employ a Jacobi iteration only for the solution required solution components. This enables us to apply standard methods to improve the convergence of linear iterative solvers. It turns out that for small phase contrasts, the method works very well and outperforms solvers that give the complete solution vector by several orders of magnitude. However, the iteration matrix has only an advantageous eigenvalue spectrum (which is direly needed for a fast convergence) for low phase contrasts and when an appropriate reference medium is chosen. For phase contrasts above 10, the solution is attainable by other means: either a more sophisticated pre-conditioning is needed or one has to resort to standard solvers. Unfortunately, while the solver is very fast, the linear system has huge memory requirements, such that its construction is not feasible

for useful discretizations. The system matrix is unsymmetric, has complex coefficients, no band structure, and is with approximately 45% of nonzero entries not sparse. This leads to roughly $0.45 \times (k \times d^3)^2$ complex matrix coefficients, where k is the number of components in the constitutive tensor, e.g., $k = 6$ for heat conduction problems, or $k = 21$ in linear elasticity. One can see that the memory requirements are excessive even for small d . Another limitation of the projector approach is that linear projections and linear constitutive laws are combined, such that an extension to nonlinear problems appears impossible.

The above approach differs somewhat from the more usual spectral solvers introduced by Moulinec and Suquet [2], which define a fixed-point iteration for solving specific boundary value problems, from which then the volume average and the effective constitutive tensor is extracted. They solve for a lower-dimensional field, e.g., the strains or temperature gradient instead for the constitutive tensor, and without the need to set up the linear system, which is why this scheme is very memory efficient. The method's good performance relies on the fast Fourier transform that is needed in every iteration step. By linearizing constitutive laws, it is possible to consider nonlinear processes incrementally.

Our approach cannot compete with this method, but is rather complementary. For example, in the Moulinec and Suquet approach, only the volume averages of the solution fields are needed for identifying the effective constitutive tensor. It may be possible to combine the approach to solve only for the volume average in both methods.

Independently, the possibility to solve only for specific variables in linear systems that appear in engineering applications is interesting all by itself. We have shown that some performance gain may be achieved for a particular engineering problem in comparison to approaching the entire linear system directly. Our approach cannot compete with established methods for this particular problem, but it may be used in other engineering contexts. An application could be to calculate results in finite element calculations only at specific, critical locations. One example is the acoustic analysis of noise generation, e.g., engine sound, where the resulting sound pressure level is commonly only needed at few locations in the vicinity of the radiator [34]. Currently, linear solvers for specific variables seem to be employed in problems from information science only [27,35].

Acknowledgements Funding was supported by Deutsche Forschungsgemeinschaft (Grant No. 411252165).

Funding Open Access funding enabled and organized by Projekt DEAL.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the

source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

Appendix A. Mathematica scripts

Mathematica scripts for the analytical (Fourier) homogenization technique presented in Sect. 3 including the locally synchronous solver based on Jacobi-iterations detailed in Sect. 4:

```

Remove["Global*"]
(* Local conductivities and ref. conductivity *)
L1 = {{3, 1, 2}, {1, 3, 1}, {2, 1, 3}};
L2 = {{2, 0, 1}, {0, 2, 1}, {1, 1, 2}};
l0 = 3;
L0 = l0 IdentityMatrix[3];

(* Verification against the laminate solution, use ALT+/ to (un)comment *)
(* L1 = 1 IdentityMatrix[3];
L2 = 4 IdentityMatrix[3]; *)

(* Parameters *)
r = 0.4;
size = 6;
center = (size + 1)/2;

(* Build the indicator function of microstructure *)
unitcell[x_] := Boole[(x[[1]] + 3 x[[2]])/4 - center)^2 +
  ((x[[2]] + 3 x[[3]] - 2 x[[1]])/4 - center)^2 +
  (x[[3]] - center)^2 <= (r size)^2;

(* Periodic shifts *)
shifts = Tuples[{-size, 0, size}, 3];

(* Unit cell plus neighbouring cells *)
ID[x_] := Max[Table[unitcell[x + shifts[[i]]], {i, 1, 27}]]

(* Verification against the laminate solution, use ALT+/ to (un)comment *)
(* ID[x_] := Boole[x[[1]] <= center]; *)

(* Discretize microstructure and perform Fourier transform *)
data = Table[ID[{i, j, k}], {L2 - L1 + L1 - L0, {j, size}, {k, size}, {i, size}];
fdata = Table[Fourier[data[[;;, ;, ;, ;, i, j]],
  FourierParameters -> {1, 1}]/size^3, {i, 1, 3}, {j, 1, 3}];

(* Declare variables *)
vars = Table[Lstar[Min[{m, n}], Max[{m, n}], i, j, k],
  {i, size}, {j, size}, {k, size}, {m, 3}, {n, 3}];

(* Define projection tensor K *)
K[k_] := Outer[Times, k, k]/k.k;

(* List of all k-vectors, excluding {0,0,0} *)
ks = DeleteCases[Flatten[Table[{i, j, k}, {i, size}, {j, size}, {k, size}], 2], {1, 1, 1}];

(* Construct the linear system matrix and right handside vector *)
nd = 6*size^3 (* overall size *)
rhs = SparseArray[{}, {nd}];(* create empty rhs *)

(* Create index translation table *)
count = 1;
indexing = Table[count++, {i, size}, {j, size}, {k, size}, {m, 3}, {n, m, 3}];
For[c1 = 1, c1 <= size, c1++,

```



```

54 For[c2 = 1, c2 <= size, c2++,
55   For[c3 = 1, c3 <= size, c3++,
56     PrependTo[indexing[[c1, c2, c3, 2]], indexing[[c1, c2, c3, 1, 2]]];
57     PrependTo[indexing[[c1, c2, c3, 3]], indexing[[c1, c2, c3, 2, 3]]];
58     PrependTo[indexing[[c1, c2, c3, 3]], indexing[[c1, c2, c3, 1, 3]]]]];

60 (* Collect coefficients of L in this order *)
61 idlist = {{1, 1}, {1, 2}, {1, 3}, {2, 2}, {2, 3}, {3, 3}};

62 curlist = Reap[Monitor[(* Cycle through eqs and identify coefficients *)
63   For[c1 = 1, c1 <= size, c1++,
64     For[c2 = 1, c2 <= size, c2++,
65       For[c3 = 1, c3 <= size, c3++,
66         expr = vars[[c1, c2, c3]] + (* Evaluate tensorial equation *)
67           Sum[fdata[;;, ;, ;,
68             Mod[c1 - ks[[i, 1]], size] + 1,
69             Mod[c2 - ks[[i, 2]], size] + 1,
70             Mod[c3 - ks[[i, 3]], size] + 1]]
71             .K[ks[[i]] - {1, 1, 1}].vars[[ks[[i, 1]], ks[[i, 2]], ks[[i, 3]]]]/10,
72             {i, Length[ks]]] -
73             fdata[;;, ;, ;, c1, c2, c3]];
74   For[jj = 1, jj <= 6, jj++, (* Collect coefficients for the components
75     11,12,13,22,23,33 *)
76     i1 = idlist[[jj, 1]];
77     i2 = idlist[[jj, 2]];
78     column = indexing[[c1, c2, c3, i1, i2]];
79     ex = Expand[expr[[i1, i2]]];
80     rhs[column] += ex[[1]]; (* The first term is the absolute part,
81     *)
82     len = Length[ex]; (* the rest is a linear comb. of the vars.
83     *)
84     data = Reap[For[ii = 2, ii <= len, ii++, (* collect the coefficients
85       in a row vector *)
86       tmp = FactorList[ex[[ii]]];
87       tmp2 = tmp[[2, 1]];
88       row = indexing[[tmp2[[3]], tmp2[[4]], tmp2[[5]], tmp2[[1]], tmp2
89         [[2]]];
90       Sow[row -> tmp[[1, 1]]];
91       Sow[SparseArray[Normal[data[[2, 1]], {nd}]]], (* Collect the row
92       vectors in a list *)
93       Row[{ProgressIndicator[column, {1, nd}], column, len}, " "]]][[2,
94       1]]; (* Progress bar *)

95 (* Convert to sparse matrix *)
96 coeffmat = SparseArray[curlist]

97 (* More explicit but less efficient generation of the linear system, (un)
98   comment with ALT + / *)
99 (* eqs = {}];
100 cs = Flatten[Table[{i, j, k}, {i, size}, {j, size}, {k, size}], 2];
101 Table[AppendTo[eqs, vars[[c[[1]], c[[2]], c[[3]]]] +
102   Sum[fdata[;;, ;, ;,
103     Mod[c[[1]] - k[[1]], size] + 1,
104     Mod[c[[2]] - k[[2]], size] + 1,
105     Mod[c[[3]] - k[[3]], size] + 1]]
106     .K[k - {1, 1, 1}].vars[[k[[1]], k[[2]], k[[3]]]]/10, {k, ks}] -
107     fdata[;;, ;, ;, c[[1]], c[[2]], c[[3]]], {c, cs}];
108 eqlst = Flatten[Table[eqs[[i, m, n]] == 0, {i, Length[eqs]}, {m, 3}, {n,
109   m, 3}];
110 vlist = DeleteDuplicates[Flatten[vars]];
111 sys = CoefficientArrays[eqlst, vlist];
112 rhs = sys[[1]];
113 coeffmat = sys[[2]]; *)

114 (* These indices correspond to L-star *)
115 indices = {1, 2, 3, 4, 5, 6};

```

Listing 1 Mathematica-code that generates the linear system.

```

result = {10, 0, 0, 10, 0, 10} - LinearSolve[coeffmat, rhs][[indices]] //
Chop

```

Listing 2 Direct equation solver.

```

result = {10, 0, 0, 10, 0, 10} - LinearSolve[coeffmat, rhs, Method -> "
Krylov"][[indices]] // Chop

```

Listing 3 Iterative equation solver.

```

(* Local solver *)
Jaci = SparseArray[DiagonalMatrix[Table[coeffmat[[i, i]], {i, 1, Length[
rhs}]]]];
InvJaci = SparseArray[DiagonalMatrix[1/Table[coeffmat[[i, i]], {i, 1,
Length[rhs}]]]];
G = -InvJaci.(coeffmat - Jaci);
z = InvJaci.rhs;
result = {};
For[i = 1, i <= Length[indices], i++,
  index = indices[[i]];
  p = SparseArray[{}], {Length[rhs]}; (* Initialize r and p *)
  r = p;
  r[[index]] = 1;
  While[Norm[r] > 10^-10,
    p = p + r;
    r = r.G;];
  AppendTo[result, p.z]];
{10, 0, 0, 10, 0, 10} - result // Chop

```

Listing 4 Local synchronous solver.

References

- Pierard O, Friebe C, Doghri I (2004) Mean-field homogenization of multi-phase thermo-elastic composites: a general framework and its validation. *Composites Science and Technology* 64(10):1587–1603 <https://doi.org/10.1016/j.compscitech.2003.11.009>. <http://www.sciencedirect.com/science/article/pii/S0266353803004494>
- Moulinec H, Suquet P (1995) A fft-based numerical method for computing the mechanical properties of composites from images of their microstructures. In: Pyrz R (ed) *IUTAM Symposium on Microstructure-Property Interactions in Composite Materials*. Springer, Netherlands, Dordrecht, pp 235–246
- Moulinec H, Suquet P (1998) A numerical method for computing the overall response of nonlinear composites with complex microstructure. *Computer Methods in Applied Mechanics and Engineering* 157(1):69–94
- Wicht D, Schneider M, Böhlke T (2020) On Quasi-Newton methods in fast Fourier transform-based micromechanics. *International Journal for Numerical Methods in Engineering* 121 (8) (2020) 1665–1694. [arXiv:https://arxiv.org/https://onlinelibrary.wiley.com/doi/pdf/10.1002/nme.6283](https://arxiv.org/https://onlinelibrary.wiley.com/doi/pdf/10.1002/nme.6283), <https://doi.org/10.1002/nme.6283>. <https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.6283>
- Vondřejc J, Zeman J, Marek I (2012) Analysis of a fast fourier transform based method for modeling of heterogeneous materials. In: Lirkov I, Margenov S, Waśniewski J (eds) *Large-Scale Scientific Computing*. Springer, Berlin Heidelberg, Berlin, Heidelberg, pp 515–522
- Willot B, Abdallah Y-P, Pellegrini P (2014) Fourier-based schemes with modified green operator for computing the electrical response of heterogeneous media with accurate local fields. *International Journal for Numerical Methods in Engineering* 98 (7) (2014) 518–533. <http://arxiv.org/https://onlinelibrary.wiley.com/doi/pdf/10.1002/nme.4641>, <https://doi.org/10.1002/nme.4641>. <https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.4641>
- Willot F (2015) Fourier-based schemes for computing the mechanical response of composites with accurate local fields. *Comptes Rendus Mécanique* 343(3):232–245 <https://doi.org/10.1016/j.crme.2014.12.005>. <http://www.sciencedirect.com/science/article/pii/S1631072114002149>
- Mishra N, Vondřejc J, Zeman J (2016) A comparative study on low-memory iterative solvers for fft-based homogenization of periodic media. *Journal of Computational Physics* 321:151–168 <https://doi.org/10.1016/j.jcp.2016.05.041>. <http://www.sciencedirect.com/science/article/pii/S0021999116301863>

9. To V-T, Monchiet V, To Q (2017) An FFT method for the computation of thermal diffusivity of porous periodic media. *Acta Mechanica* 228(9):3019–3037
10. Eisenlohr P, Diehl M, Lebensohn R, Roters F (2013) A spectral method solution to crystal elasto-viscoplasticity at finite strains. *International Journal of Plasticity* 46:37–53
11. Zeman J, Vondřejc J, Novák J, Marek I (2010) Accelerating a fft-based solver for numerical homogenization of periodic media by conjugate gradients. *Journal of Computational Physics* 229(21):8065–8071
12. Brisard S, Dormieux L (2010) Fft-based methods for the mechanics of composites: A general variational framework. *Computational Materials Science* 49(3):663–671 <https://doi.org/10.1016/j.commatsci.2010.06.009>. <http://www.sciencedirect.com/science/article/pii/S0927025610003563>
13. Milton G (2002) *The Theory of Composites*. Cambridge University Press,
14. Nemat-Nasser S, Taya M (1981) On effective moduli of an elastic body containing periodically distributed voids. *Quarterly of Applied Mathematics* 39(1):43–59
15. Nemat-Nasser S, Iwakuma T, Hejazi M (1982) On composites with periodic structure. *Mechanics of Materials* 1(3):239–267
16. Iwakuma T, Nemat-Nasser S (1983) Composites with periodic microstructure. *Computers & Structures* 16(1):13–19
17. Nemat-Nasser S, Yu N, Hori M (1993) Solids with periodically distributed cracks. *International Journal of Solids and Structures* 30(15):2071–2095
18. Fotiu PA, Nemat-Nasser S (1996) Overall properties of elastic-viscoplastic periodic composites. *International Journal of Plasticity* 12(2):163–190
19. Chen Y (2004) *Percolation and Homogenization Theories for Heterogeneous Materials*, Ph.D. thesis at Massachusetts Institute of Technology
20. Morawiec A (2004) *Orientations and Rotations-Computations in Crystallographic Textures*. Springer,
21. Torquato S (1997) Effective stiffness tensor of composite media-I. Exact series expansions. *Journal of the Mechanics and Physics of Solids* 45(9):1421–1448
22. Vondřejc J (2013) *FFT-based method for homogenization of periodic media: Theory and applications*, dissertation, Czech Technical University in Prague. <https://pdfs.semanticscholar.org/5cf3/a69eedfb1e6ee260e30a02a28a899faea46.pdf>
23. Stoer J, Bulirsch R (2007) *Numerische Mathematik 1*. Springer, Berlin
24. Saad Y (2003) *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics. <https://doi.org/10.1137/1.9780898718003>
25. Stoer J, Bulirsch R (2005) *Numerische Mathematik 2*. Springer Verlag. <https://doi.org/10.1007/978-3-540-45390-1>
26. Hackbusch W (2016) *Iterative Solution of Large Sparse Systems of Equations*. Springer International Publishing. <https://doi.org/10.1007/978-3-319-28483-5>
27. Lee CE, Ozdaglar AE, Shah D Solving systems of linear equations: Locally and asynchronously, [arXiv:abs/1411.2647](https://arxiv.org/abs/1411.2647)
28. Lee CE, Ozdaglar AE, Shah D Solving for a single component of the solution to a linear system, asynchronously
29. Ozdaglar A, Shah D, Yu C Lee (2019) Asynchronous approximation of a single component of the solution to a linear system, *IEEE Transactions on Network Science and Engineering*, 1–12 <https://doi.org/10.1109/TNSE.2019.2894990>
30. Milaszewicz JP (1987) Improving Jacobi and Gauss-Seidel iterations. *Linear Algebra and its Applications* 93:161–170. [https://doi.org/10.1016/S0024-3795\(87\)90321-1](https://doi.org/10.1016/S0024-3795(87)90321-1)
31. Gunawardena AD, Jain SK, Snyder L (1991) Modified iterative methods for consistent linear systems. *Linear Algebra and its Applications* 154–156:123–143. [https://doi.org/10.1016/0024-3795\(91\)90376-8](https://doi.org/10.1016/0024-3795(91)90376-8)
32. Kotakemori H, Harada K, Morimoto M, Niki H (2002) A comparison theorem for the iterative method with the preconditioner (I+Smax). *Journal of Computational and Applied Mathematics* 145(2):373–378. [https://doi.org/10.1016/S0377-0427\(01\)00588-X](https://doi.org/10.1016/S0377-0427(01)00588-X)
33. Urekw TJ, Rencis JJ (1993) The importance of diagonal dominance in the iterative solution of equations generated from the boundary element method. *International Journal for Numerical Methods in Engineering* 36(20):3509–3527. <https://doi.org/10.1002/nme.1620362007>
34. Duveigneau F, Liefold S, Höchstetter M, Verhey JL, Gabbert U (2016) Analysis of simulated engine sounds using a psychoacoustic model. *Journal of Sound and Vibration* 366:544–555. <https://doi.org/10.1016/j.jsv.2015.11.034>
35. Andersen R, Borgs C, Chayes J, Hopcraft J, Mirrokni VS, Teng S-H (2007) Local computation of PageRank contributions, in: A. Bonato, F. R. K. Chung (Eds.), *Algorithms and Models for the Web-Graph*, Springer Berlin Heidelberg, pp. 150–165. <https://doi.org/10.1007/978-3-540-77004-6>

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.