Available online at www.sciencedirect.com
ScienceDirect
Comput. Methods Appl. Mech. Engrg. 364 (2020) 112890
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

FFT-based homogenisation accelerated by low-rank tensor approximations

Jaroslav Vondřejc[a,*], Dishi Liu[a], Martin Ladecký[b], Hermann G. Matthies[a]

[a] *Technische Universität Braunschweig, Institute of Scientific Computing, Mühlenpfordtstrasse 23, 38106 Braunschweig, Germany*
[b] *Czech Technical University in Prague, Faculty of Civil Engineering, Thákurova 7/2077, 166 29 Prague 6, Czech Republic*

Received 17 July 2019; received in revised form 30 November 2019; accepted 30 January 2020
Available online 28 February 2020

Abstract

Fast Fourier transform (FFT) based methods have turned out to be an effective computational approach for numerical homogenisation. In particular, Fourier–Galerkin methods are computational methods for partial differential equations that are discretised with trigonometric polynomials. Their computational effectiveness benefits from efficient FFT based algorithms as well as a favourable condition number. Here these kinds of methods are accelerated by low-rank tensor approximation techniques for a solution field using canonical polyadic, Tucker, and tensor train formats. This reduced order model also allows to efficiently compute suboptimal global basis functions without solving the full problem. It significantly reduces computational and memory requirements for problems with a material coefficient field that admits a moderate rank approximation. The advantages of this approach against those using full material tensors are demonstrated using numerical examples for the model homogenisation problem that consists of a scalar linear elliptic variational problem defined in two and three dimensional settings with continuous and discontinuous heterogeneous material coefficients. This approach opens up the potential of an efficient reduced order modelling of large scale engineering problems with heterogeneous material.

Keywords: Fourier–Galerkin method; Fast Fourier transform; Low-rank approximations; Reduced order modelling; Homogenisation

1. Introduction

FFT-based methods. A fast Fourier transform (FFT) based method has been introduced as an efficient algorithm for numerical homogenisation in 1994 by Moulinec and Suquet [1]. The method, that has application in multiscale problems, represents an alternative discretisation approach to the finite element method. The effectiveness of FFT-based homogenisation relies on the facts that the system matrix is never assembled, the matrix–vector product in linear iterative solvers is provided very efficiently by FFT, and the condition number is independent of discretisation parameters.

Since the seminal paper in 1994 the methodology has been significantly developed. Originally the approach has been based on Lippmann–Schwinger equation, which is a formulation incorporating Green's function for an auxiliary homogeneous problem. Its connection to a standard variational formulation has been discovered

* Corresponding author.
E-mail address: j.vondrejc@tu-bs.de (J. Vondřejc).

https://doi.org/10.1016/j.cma.2020.112890

in [2] by using the fact that Green's function is a projection on compatible fields (i.e. gradient fields in scalar elliptic problems), see [3]. It has allowed to fully remove the reference conductivity tensor from the formulation, and interpreted the method from the perspective of finite elements also in nonlinear problems [4,5]. Moreover, the standard primal–dual variational formulations allow to compute guaranteed bounds on effective material properties [6], which provides tighter bounds than the Hashin–Shtrikman functional.

Significant attention has been focused on developing discretisation approaches that justify the original FFT-based homogenisation algorithm. Many efforts have been made on discretisation with trigonometric polynomials, starting with [7] and followed by [4,6,8,9]. Other discretisation approaches are based on pixel-wise constant basis functions [10,11], linear hexahedral elements [12], or finite differences [13,14]. The variational formulations also allowed to derive convergence of approximate solutions to the continuous one [2,9,10].

The various discretisation approaches have been studied along with linear and non-linear solvers [4,5,7,11,15–19]. Other research directions focus, for example, on multiscale methods [20–22], highly non-linear problems in solid mechanics [23–26], and parameter estimation features FFT and model reduction [27].

Low-rank approximations. The general idea of low-rank approximations is to express or compress tensors with fewer parameters, which can lead to a huge reduction in requirements for computer memory and possible significant computational speed-up. For matrices as second order tensors, the optimal low-rank approximation in mean square sense is based on the truncated singular value decomposition (SVD). A computationally cheaper choice is Cross Approximation [28,29] which has only linear complexity in matrix size N. Low-rank formats or tensors of order larger than two include the canonical polyadic (CP), Tucker, and hierarchical schemes such as the tensor train and the quantic tensor-train form of [30,31]. Low-rank formats are not only needed to compress the data tensor as the final delivered result of high-dimensional numerical modellings, but are also preferred to approximate tensors in the numerical solution process. In [32] the proper generalised decomposition is adopted for the construction of low-rank tensors in CP and Tucker formats in a numerical homogenisation from high-resolution images. It is also possible to compute the tensors directly in low-rank formats, which can be provided by a suitable solver [33–37]. The rank one tensors in low-rank approximations can be seen as suboptimal global basis functions.

However, the need to compute with tensors in low-rank formats requires one to deal with operations such as addition, element-wise multiplication, or Fourier transformation. Since the low-rank tensors are described with fewer parameters, the computational complexities are typically reduced, which may lead to significant speed-up of computations. However, performing such operations with tensors in low-rank format, it typically happens that the representation rank of the tensors grows, which calls for their truncation, i.e. their approximation or reparametrisation with fewer parameters while keeping a reasonable accuracy [38–40]. This truncation of tensors may be viewed as a generalisation of the rounding of numbers, which occurs when working with floating point formats. In general, the applications of low-rank approximations are very broad, e.g. for stochastic problems with high number of random parameters [37,41–43], acceleration of solutions to PDEs [44,45], or model order reduction [46], but its application to FFT-based homogenisation is new. However, an alternative low-rank representation has been studied recently in [47].

Structure of the paper. In Section 2, two state-of-the-art Fourier–Galerkin methods are described for a model homogenisation problem of a scalar elliptic equation. In particular, the two discretisation methods based on numerical and exact integration are described along with their corresponding linear systems. Then in Section 3 the low-rank approximation techniques are summarised and their application within a Fourier–Galerkin method is discussed. In Section 4, the effectiveness of low-rank approximations is demonstrated on several numerical examples.

Notation. We will denote vectors and matrices by boldface letters: $\boldsymbol{a} = (a_i)_{i=1}^d \in \mathbb{R}^d$ or $\boldsymbol{A} = (A_{ij})_{i,j=1}^d \in \mathbb{R}^{d\times d}$. Matrix–matrix and matrix–vector multiplications are denoted as $\boldsymbol{C} = \boldsymbol{A}\boldsymbol{B}$ and $\boldsymbol{c} = \boldsymbol{A}\boldsymbol{b}$, which in Einstein summation notation reads $C_{ik} = A_{ij}B_{jk}$ and $b_i = A_{ij}b_j$ respectively. The element-wise (Hadamard) product of two vectors is denoted as $\boldsymbol{a} \odot \boldsymbol{b} = (a_i b_i)_{i=1}^d$. The Euclidean inner product will be referred to as $\boldsymbol{a} \cdot \bar{\boldsymbol{b}} = \sum_i a_i \bar{b}_i$, and the induced norm as $\|\boldsymbol{a}\| = \sqrt{\boldsymbol{a} \cdot \bar{\boldsymbol{a}}}$. Vectors, matrices, and tensors such as x, b, and $\mathbf{A}$ arising from discretisation will be denoted by the bold sans-serif font in order to highlight their special structure. For $\boldsymbol{N} = (N_1, \dots, N_d) \in \mathbb{N}^d$, the components of a tensor $\mathbf{A} \in \mathbb{R}^{\boldsymbol{N}} = \bigotimes_{\alpha=1}^d \mathbb{R}^{N_\alpha}$ of order d will be denoted as $\mathsf{A}[k_1, \dots, k_d]$. The multiindex notation will be also incorporated to simplify the components of the tensors, e.g. $\mathsf{A}[k_1, \dots, k_d] = \mathsf{A}[\boldsymbol{k}]$ for a multi-index $\boldsymbol{k} = [k_1, \dots, k_d]$. The space $\mathbb{R}^{\boldsymbol{N}}$, composed of tensors of order d, can be considered as a vector space, which allows to talk about its dimension as the number of basis vectors, i.e. $\dim \mathbb{R}^{\boldsymbol{N}} = \prod_{\alpha=1}^d N_\alpha$.

The space of square integrable $\mathcal{Y}$-periodic functions defined on a periodic cell $\mathcal{Y} = (-\frac{1}{2}, \frac{1}{2})^d$ is denoted as $L^2(\mathcal{Y})$. The analogous space $L^2(\mathcal{Y}; \mathbb{R}^d)$ collects $\mathbb{R}^d$-valued functions $\boldsymbol{v} : \mathcal{Y} \to \mathbb{R}^d$ with components v_i from $L^2(\mathcal{Y})$. Finally, $H^1_0(\mathcal{Y}) = \{v \in L^2(\mathcal{Y}) \mid \nabla v \in L^2(3\mathcal{Y}; \mathbb{R}^d), \int_{\mathcal{Y}} v(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = 0\}$ denotes the Sobolev space of periodic functions with zero mean; note that the Sobolev functions are also differentiable on the boundary.

2. Homogenisation by Fourier–Galerkin methods

2.1. Model problem

A model problem in homogenisation [48] consists of a scalar linear elliptic variational problem defined on a unit domain $\mathcal{Y} = (-\frac{1}{2}, \frac{1}{2})^d$ in a spatial dimension d (we consider both $d = 2$ and $d = 3$) with material coefficients $\boldsymbol{A} : \mathcal{Y} \to \mathbb{R}^{d\times d}$, which are required to be essentially bounded, symmetric, and uniformly elliptic. This means that for almost all $\boldsymbol{x} \in \mathcal{Y}$, there are constants $0 < c_A \le C_A < +\infty$ such that

$$\boldsymbol{A}(\boldsymbol{x}) = \boldsymbol{A}^T(\boldsymbol{x}), \quad c_A\|\boldsymbol{v}\|^2 \le \boldsymbol{A}(\boldsymbol{x})\boldsymbol{v}\cdot\boldsymbol{v} \le C_A\|\boldsymbol{v}\|^2 \quad \text{for all } \boldsymbol{v} \in \mathbb{R}^d. \tag{1}$$

The homogenisation problem is focused on the computation of effective material properties $\boldsymbol{A}_{\mathrm{H}} \in \mathbb{R}^{d\times d}$. Its variational formulation is based on the minimisation of a microscopic energetic functional for constant vectors $\boldsymbol{E} \in \mathbb{R}^d$, which represents an *average* of the macroscopic gradient, as

$$\boldsymbol{A}_{\mathrm{H}}\boldsymbol{E}\cdot\boldsymbol{E} = \min_{v\in H^1_0(\mathcal{Y})} a\big(\boldsymbol{E}+\nabla v, \boldsymbol{E}+\nabla v\big), \tag{2}$$

where the bilinear form $a : L^2(\mathcal{Y}; \mathbb{R}^d) \times L^2(\mathcal{Y}; \mathbb{R}^d) \to \mathbb{R}$ is defined as

$$a\big(\boldsymbol{e}, \boldsymbol{w}\big) := \int_{\mathcal{Y}} \boldsymbol{A}(\boldsymbol{x})\boldsymbol{e}(\boldsymbol{x})\cdot\boldsymbol{w}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}. \tag{3}$$

The minimisation Sobolev space $H^1_0(\mathcal{Y})$ consists of zero-mean $\mathcal{Y}$-periodic microscopic fields $v : \mathbb{R}^d \to \mathbb{R}$, which have locally square integrable weak gradient and finite L^2-norm on $\mathcal{Y}$; together with (1) it satisfies the existence of a unique minimiser. Note that the minimisation problem (2) corresponds to the scalar elliptic partial differential equation $-\nabla\cdot[\boldsymbol{A}(\boldsymbol{x})\nabla u(\boldsymbol{x})] = f(\boldsymbol{x})$ with a special right-hand side $f(\boldsymbol{x}) = -\nabla\cdot\boldsymbol{A}(\boldsymbol{x})\boldsymbol{E}$ and periodic boundary conditions.

2.2. Fourier–Galerkin methods

Alternatively, the minimisers in (2) are described by a weak formulation: find $u \in H^1_0(\mathcal{Y})$ such that

$$a\big(\nabla u, \nabla v\big) = -a\big(\boldsymbol{E}, \nabla v\big) \quad \forall\, v \in H^1_0(\mathcal{Y}). \tag{4}$$

This formulation is the starting point for a discretisation using Galerkin approximations, when the trial and test spaces are substituted with finite dimensional ones. We choose to discretise the function space using trigonometric polynomials, which leads to a Fourier–Galerkin method.

In order to compute the effective matrix $\boldsymbol{A}_{\mathrm{H}}$ one has to solve d minimisation problems or weak formulation for different $\boldsymbol{E}$, which are usually taken as the canonical basis of $\mathbb{R}^d$. Here we consider exclusively $\boldsymbol{E} = (\delta_{1,i})_{i=1}^d \in \mathbb{R}^d$ (i.e. in 3D $\boldsymbol{E} = [1, 0, 0]$); therefore, the $(1, 1)$-component of the homogenised properties will be of particular interest, i.e. $\boldsymbol{A}_{\mathrm{H}}\boldsymbol{E}\cdot\boldsymbol{E} = \boldsymbol{A}_{\mathrm{H},11} =: A_{\mathrm{H}}$.

2.2.1. Trigonometric polynomials

The Fourier–Galerkin method, [2,8,49] is built on discretisations using the space of *trigonometric polynomials*

$$\mathcal{T}_N = \Big\{ \sum_{\boldsymbol{k}\in\mathbb{Z}_N} \widehat{v}[\boldsymbol{k}]\varphi^{\boldsymbol{k}} \;\mid\; \widehat{v}[\boldsymbol{k}] \in \mathbb{C}, \text{ and } \widehat{v}[\boldsymbol{k}] = \overline{\widehat{v}[-\boldsymbol{k}]} \Big\},$$

where $\varphi^{\boldsymbol{k}}(\boldsymbol{x}) = \exp(2\pi\mathrm{i}\,\boldsymbol{k}\cdot\boldsymbol{x})$ are the well-known Fourier basis functions and $\mathbb{Z}_N = \{\boldsymbol{k} \in \mathbb{Z}^d\,;\, |k_i| < \frac{N_i}{2} \text{ for } i = 1, \dots, d\}$ is an index set. The number of discretisation points $\boldsymbol{N} = [N, \dots, N] \in \mathbb{R}^d$ in this work take only odd values because an even N introduces Nyquist frequencies that have to be omitted to obtain a conforming approximation, see [6] for details.

There are also other natural basis vectors $\varphi_N^{\boldsymbol{k}} : \mathcal{Y} \to \mathbb{R}$, the so-called fundamental trigonometric polynomials. They are expressed as a linear combination

$$\varphi_N^{\boldsymbol{k}}(\boldsymbol{x}) = \frac{1}{|\boldsymbol{N}|_{\Pi}} \sum_{\boldsymbol{m}\in\mathbb{Z}_{\boldsymbol{N}}} \omega_{\boldsymbol{N}}^{-\boldsymbol{km}} \varphi^{\boldsymbol{m}}(\boldsymbol{x}) \text{ for } \boldsymbol{x} \in \mathcal{Y}$$

of Fourier basis function $\varphi^{\boldsymbol{m}}$ with complex-valued weights $\omega_{\boldsymbol{N}}^{\boldsymbol{mk}} = \exp\big(2\pi \mathrm{i} \sum_{\alpha=1}^{d} \frac{m_\alpha k_\alpha}{N_\alpha}\big)$ for $\boldsymbol{m}, \boldsymbol{k} \in \mathbb{Z}_{\boldsymbol{N}}$. The weights are from the discrete Fourier transform (DFT) matrices in $\mathbb{C}^{\boldsymbol{N}\times\boldsymbol{N}}$ with components

$$\mathcal{F}_{\boldsymbol{N}}[\boldsymbol{m},\boldsymbol{k}] = \frac{1}{|\boldsymbol{N}|_{\Pi}} \omega_{\boldsymbol{N}}^{-\boldsymbol{mk}}, \quad \mathcal{F}_{\boldsymbol{N}}^{-1}[\boldsymbol{m},\boldsymbol{k}] = \omega_{\boldsymbol{N}}^{\boldsymbol{mk}} \quad \text{for } \boldsymbol{m},\boldsymbol{k} \in \mathbb{Z}_{\boldsymbol{N}}.$$

The coefficients of trigonometric polynomials in the two different bases are connected by the discrete Fourier transform (DFT), particularly expressed as

$$v(\boldsymbol{x}) = \sum_{\boldsymbol{k}\in\mathbb{Z}_{\boldsymbol{N}}} \widehat{v}[\boldsymbol{k}]\varphi^{\boldsymbol{k}}(\boldsymbol{x}) = \sum_{\boldsymbol{k}\in\mathbb{Z}_{\boldsymbol{N}}} \boldsymbol{v}[\boldsymbol{k}]\varphi_{\boldsymbol{N}}^{\boldsymbol{k}}(\boldsymbol{x}) \quad \text{and} \quad \widehat{\boldsymbol{v}} = \mathcal{F}_{\boldsymbol{N}}\boldsymbol{v}.$$

Due to the Dirac-delta property $\varphi_{\boldsymbol{N}}^{\boldsymbol{l}}(\boldsymbol{x}_{\boldsymbol{N}}^{\boldsymbol{k}}) = \delta_{\boldsymbol{kl}}$ of the fundamental trigonometric polynomials on a regular grid of points $\boldsymbol{x}_{\boldsymbol{N}}^{\boldsymbol{k}} = \frac{\boldsymbol{k}}{\boldsymbol{N}_\alpha}$ for $\boldsymbol{k},\boldsymbol{l} \in \mathbb{Z}_{\boldsymbol{N}}$, the coefficients of the trigonometric polynomials are equal to the function values at the grid points, i.e. $\boldsymbol{v}[\boldsymbol{k}] = v(\boldsymbol{x}_{\boldsymbol{N}}^{\boldsymbol{k}})$.

Differential operators are naturally applied on trigonometric polynomials. In particular the gradient

$$\nabla v(\boldsymbol{x}) = \sum_{\boldsymbol{k}\in\mathbb{Z}_{\boldsymbol{N}}} \hat{v}[\boldsymbol{k}]\nabla\varphi^{\boldsymbol{k}}(\boldsymbol{x}) = \sum_{\boldsymbol{k}\in\mathbb{Z}_{\boldsymbol{N}}} 2\pi\mathrm{i}\boldsymbol{k}\hat{v}[\boldsymbol{k}]\varphi^{\boldsymbol{k}}(\boldsymbol{x}),$$

corresponds to the application of the operator $\widehat{\nabla}_{\boldsymbol{N}} : \mathbb{C}^{\boldsymbol{N}} \to \mathbb{C}^{d\times\boldsymbol{N}}$ on Fourier coefficients as $(\widehat{\nabla}_{\boldsymbol{N}}\hat{v})[\alpha,\boldsymbol{k}] = 2\pi\mathrm{i}k_\alpha\hat{v}[\boldsymbol{k}]$. The adjoint operator $\widehat{\nabla}_{\boldsymbol{N}}^* : \mathbb{C}^{d\times\boldsymbol{N}} \to \mathbb{C}^{\boldsymbol{N}}$ corresponding to the divergence is then expressed as

$$(\widehat{\nabla}_{\boldsymbol{N}}^*\hat{\boldsymbol{w}})[\boldsymbol{k}] = \sum_{\alpha=1}^{d} -2\pi\mathrm{i}k_\alpha\hat{\boldsymbol{w}}[\alpha,\boldsymbol{k}].$$

Then the gradient operator can be expressed with respect to the basis with fundamental trigonometric polynomials as

$$\nabla v(\boldsymbol{x}) = \sum_{\boldsymbol{k}} \big(\boldsymbol{\mathcal{F}}_{\boldsymbol{N}}^{-1}\widehat{\nabla}_{\boldsymbol{N}}\mathcal{F}_{\boldsymbol{N}}\boldsymbol{v}\big)[\boldsymbol{k}]\varphi_{\boldsymbol{N}}^{\boldsymbol{k}}(\boldsymbol{x}) \tag{5}$$

where the d-fold discrete Fourier transform (emphasised with bold) $\boldsymbol{\mathcal{F}}_{\boldsymbol{N}} = \mathbb{C}^{d\times\boldsymbol{N}} \to \mathbb{C}^{d\times\boldsymbol{N}}$ acts individually on each component of the vector field $(\boldsymbol{\mathcal{F}}_{\boldsymbol{N}}\boldsymbol{w})[\alpha] = \mathcal{F}_{\boldsymbol{N}}\boldsymbol{w}[\alpha]$ for $\alpha = 1,\dots,d$.

The numerical treatment of the weak formulation (4) or a corresponding Galerkin approximation requires the use of numerical integration. In this manuscript we incorporate two versions: an exact integration [8] as described in Section 2.2.3, and a numerical integration as described in Section 2.2.2.

2.2.2. The Fourier–Galerkin method with numerical integration (GaNi)

This numerical integration based on the rectangle (or the mid-point) rule corresponds to the original Moulinec–Suquet algorithm [1,50], as the resulting discrete solution vectors fully coincide. This approach, applied to the bilinear form (3) on regular grids, reads

$$a(\boldsymbol{e},\boldsymbol{w}) \approx a_{\boldsymbol{N}}(\boldsymbol{e}_{\boldsymbol{N}},\boldsymbol{w}_{\boldsymbol{N}}) = \sum_{\boldsymbol{k}\in\mathbb{Z}_{\boldsymbol{N}}} \boldsymbol{A}(\boldsymbol{x}_{\boldsymbol{N}}^{\boldsymbol{k}})\boldsymbol{e}_{\boldsymbol{N}}(\boldsymbol{x}_{\boldsymbol{N}}^{\boldsymbol{k}}) \cdot \boldsymbol{w}_{\boldsymbol{N}}(\boldsymbol{x}_{\boldsymbol{N}}^{\boldsymbol{k}}) = \big(\widetilde{\boldsymbol{A}}\boldsymbol{e},\boldsymbol{w}\big)_{\mathbb{R}^{d\times\boldsymbol{N}}},$$

where $\boldsymbol{e}$ and $\boldsymbol{w}$ store the function values on the grid (e.g. $\boldsymbol{e}[\alpha,\boldsymbol{k}] = e_{\boldsymbol{N},\alpha}(\boldsymbol{x}_{\boldsymbol{N}}^{\boldsymbol{k}})$), and $\widetilde{\boldsymbol{A}} \in \mathbb{R}^{d\times d\times\boldsymbol{N}\times\boldsymbol{N}}$ is a block diagonal tensor with components

$$\widetilde{\boldsymbol{A}}[\alpha,\beta,\boldsymbol{k},\boldsymbol{l}] = \delta_{\boldsymbol{kl}}A_{\alpha\beta}(\boldsymbol{x}_{\boldsymbol{N}}^{\boldsymbol{k}});$$

but one only needs to store the diagonals, which can be done in a tensor of shape $d \times d \times \boldsymbol{N}$.

The numerical integration leads to an approximate formulation of the Galerkin approximation of (4):

$$\text{find } u \in \mathcal{T}_{\boldsymbol{N}}: \quad a_{\boldsymbol{N}}\big(\nabla u_{\boldsymbol{N}},\nabla v_{\boldsymbol{N}}\big) = -a_{\boldsymbol{N}}\big(\boldsymbol{E},\nabla v_{\boldsymbol{N}}\big), \quad \forall v_{\boldsymbol{N}} \in \mathcal{T}_{\boldsymbol{N}};$$

note that the approximation is exact for constant material coefficients $\boldsymbol{A}$. This formulation, that can be seen also as a collocation method [7], is equivalent to the original Moulinec and Suquet formulation [1] in the sense that the solution vectors coincide [2]. However, the formulation here builds on the variational formulation [2] solved for the potential field (instead of gradient one).

The combination of numerical integration and differentiation of trigonometric polynomials (5) allows to approximate the bilinear form in terms of the nodal values of potential fields

$$a_N\big(\nabla u_N, \nabla v_N\big) = \big(\widetilde{\boldsymbol{A}}\boldsymbol{\mathcal{F}}_N^{-1}\widehat{\nabla}_N\mathcal{F}_N u, \boldsymbol{\mathcal{F}}_N^{-1}\widehat{\nabla}_N\mathcal{F}_N v\big)_{\mathbb{R}^{d\times N}}.$$

In order to deduce the linear system, all operators acting on test vectors v_N are moved to the trial vector u_N as adjoint operators to reveal the linear system in the original space

$$\mathcal{F}_N^{-1}\widehat{\nabla}_N^*\boldsymbol{\mathcal{F}}_N\widetilde{\boldsymbol{A}}\boldsymbol{\mathcal{F}}_N^{-1}\widehat{\nabla}_N\mathcal{F}_N u = -\mathcal{F}_N^{-1}\widehat{\nabla}_N^*\boldsymbol{\mathcal{F}}_N\widetilde{\boldsymbol{A}}E,$$

where $E \in \mathbb{R}^{d\times N}$ is constant with components $E[\alpha, \boldsymbol{k}] = E_\alpha$. One may notice that the system can be solved in Fourier space to save one computation of FFT and its inverse, which leads to the linear system in Fourier space

$$\widehat{\nabla}_N^*\boldsymbol{\mathcal{F}}_N\widetilde{\boldsymbol{A}}\boldsymbol{\mathcal{F}}_N^{-1}\widehat{\nabla}_N\widehat{u} = -\widehat{\nabla}_N^*\boldsymbol{\mathcal{F}}_N\widetilde{\boldsymbol{A}}E. \tag{6}$$

2.2.3. Fourier-Galerkin method with exact integration (GA)

For many types of material coefficients (1) and basis functions, there is a possibility to integrate the bilinear forms in the weak formulation exactly, which leads to a Galerkin approximation with exact integration

$$\text{find } u \in \mathcal{T}_N: \quad a\big(\nabla u_N, \nabla v_N\big) = a\big(\boldsymbol{E}, \nabla v_N\big) \quad \forall v_N \in \mathcal{T}_N. \tag{7}$$

However, the exact integration of the Fourier–Galerkin formulation, in contrast to FEM, leads to a full linear system, which can be overcome with a double-grid integration with projection (DoGIP) [6,8]. The DoGIP is a general method applicable also within the finite element method [51]. The original evaluation of the material law on a grid of size $\boldsymbol{N}$ is reformulated as an evaluation on a double grid $2\boldsymbol{N}-\boldsymbol{1}$ with modified material coefficients; they can be expressed as a modification of the original material coefficients.

The main idea relies on expressing gradients of the trial and a test function together

$$\nabla u_N(\boldsymbol{x}) \otimes \nabla v_N(\boldsymbol{x}) = \boldsymbol{e}_N(\boldsymbol{x}) \otimes \boldsymbol{w}_N(\boldsymbol{x}) = \sum_{\boldsymbol{k}\in\mathbb{Z}_{2N-1}} \boldsymbol{e}[:, \boldsymbol{k}] \otimes \boldsymbol{w}[:, \boldsymbol{k}]\varphi^{\boldsymbol{k}}_{2N-1}(\boldsymbol{x})$$

with respect to the basis of the double grid space consisting of trigonometric polynomials with doubled frequencies $\mathcal{T}_{2N-1}$; the arrays $\boldsymbol{e}$ and $\boldsymbol{w}$ store the values of the trigonometric polynomials on the double grid, e.g. $\boldsymbol{e}[\alpha, \boldsymbol{k}] = \boldsymbol{e}_{N,\alpha}(\boldsymbol{x}^{\boldsymbol{k}}_{2N-1})$ for $\alpha \in \{1, \dots, d\}$ and $\boldsymbol{k} \in \mathbb{Z}_{2N-1}$. Then the bilinear form can be expressed on the double grid

$$a\big(\boldsymbol{e}_N, \boldsymbol{w}_N\big) = \sum_{\boldsymbol{k}\in\mathbb{Z}_{2N-1}} \int_{\mathcal{Y}} \boldsymbol{A}(\boldsymbol{x})\varphi^{\boldsymbol{k}}_{2N-1}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} : \boldsymbol{e}_{2N-1}[:, \boldsymbol{k}] \otimes \boldsymbol{w}_{2N-1}[:, \boldsymbol{k}] = \big(\boldsymbol{A}\boldsymbol{e}, \boldsymbol{w}\big)_{\mathbb{R}^{d\times(2N-1)}}$$

where the operator : is a double contraction between two matrices of size $d \times d$ and the material coefficients are defined as

$$\boldsymbol{A}[\alpha, \beta, \boldsymbol{k}, \boldsymbol{l}] = \delta_{\boldsymbol{kl}} \int_{\mathcal{Y}} A_{\alpha\beta}(\boldsymbol{x})\varphi^{\boldsymbol{k}}_{2N-1}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \quad \text{for } \alpha, \beta \in \{1, \dots, d\} \text{ and } \boldsymbol{k}, \boldsymbol{l} \in \mathbb{Z}_{2N-1}.$$

This integration can be performed exactly for a large class of material coefficients. In particular in [6,8], square or circular inclusions have been considered, as well as image-based composites, materials with coefficients constant or bilinear over pixels (voxels in 3D). Moreover, the evaluation of modified material coefficients can be performed effectively by FFT.

In order to derive the linear system, we have to still describe the interpolation from the original to the double grid space. As the spaces of trigonometric polynomials are nested $\mathcal{T}_N \subset \mathcal{T}_M$ for $\boldsymbol{N} < \boldsymbol{M}$ (element-wise), we can just inject the polynomial to the bigger space by adding trigonometric polynomials with zero Fourier coefficients. This can be represented by the zero-padding injection operator $\mathcal{I} : \mathbb{C}^{d\times N} \to \mathbb{C}^{d\times(2N-1)}$, defined as

$$(\mathcal{I}\widehat{\boldsymbol{w}})[:, \boldsymbol{k}] = \begin{cases} \widehat{\boldsymbol{w}}[:, \boldsymbol{k}], & \text{for } \boldsymbol{k} \in \mathbb{Z}_N \\ \boldsymbol{0} & \text{for } \boldsymbol{k} \in \mathbb{Z}_{2N-1}\backslash\mathbb{Z}_N. \end{cases}$$

Its adjoint operator $\mathcal{I}^* : \mathbb{C}^{d\times(2N-1)} \to \mathbb{C}^{d\times N}$ just removes the frequencies $\boldsymbol{k} \in \mathbb{Z}_{2N-1}\backslash\mathbb{Z}_N$, i.e. projects on the $\boldsymbol{k} \in \mathbb{Z}_N$.

This allows us to deduce the linear system with exact integration

$$\widehat{\nabla}_N^*\mathcal{I}^*\boldsymbol{\mathcal{F}}_{2N-1}\boldsymbol{A}\boldsymbol{\mathcal{F}}_{2N-1}^{-1}\mathcal{I}\widehat{\nabla}_N\widehat{u} = -\widehat{\nabla}_N^*\mathcal{I}^*\boldsymbol{\mathcal{F}}_{2N-1}\boldsymbol{A}E, \tag{8}$$

which has very similar structure compared to the scheme based on numerical integration (6).

2.3. Preconditioning

Following the recent paper [52], the preconditioning of both linear systems (6) and (8) is based on a Laplacian expressed in the Fourier domain as

$$\widehat{P}[\boldsymbol{k},\boldsymbol{l}] = \delta_{\boldsymbol{k}\boldsymbol{l}}\boldsymbol{k}\cdot\boldsymbol{l} \quad \text{for } \boldsymbol{k},\boldsymbol{l} \in \mathbb{Z}_N,$$

which is a simple diagonal preconditioner. Its inverse is given by the Moore–Penrose pseudoinverse $\widehat{P}^{-1}[\boldsymbol{k},\boldsymbol{l}] = \delta_{\boldsymbol{k}\boldsymbol{l}}\frac{1}{\boldsymbol{k}\cdot\boldsymbol{k}}$ for $\boldsymbol{k} \in \mathbb{Z}_N\backslash\{\boldsymbol{0}\}$ and $\widehat{P}^{-1}[\boldsymbol{0},\boldsymbol{0}] = 0$; the latter condition enforces the zero-mean property of the approximated vectors. The preconditioned systems are explicitly stated for both discretisation schemes

$$\widehat{P}^{-1}\widehat{\nabla}_N^*\boldsymbol{\mathcal{F}}_N\widetilde{\boldsymbol{A}}\boldsymbol{\mathcal{F}}_N^{-1}\widehat{\nabla}_N\widehat{u} = -\widehat{P}^{-1}\widehat{\nabla}_N^*\boldsymbol{\mathcal{F}}_N\widetilde{\boldsymbol{A}}E, \tag{9a}$$

for the preconditioning of (6), and

$$\widehat{P}^{-1}\widehat{\nabla}_N^*\mathcal{I}^*\boldsymbol{\mathcal{F}}_{2N-1}\boldsymbol{A}\boldsymbol{\mathcal{F}}_{2N-1}^{-1}\mathcal{I}\widehat{\nabla}_N\widehat{u} = -\widehat{P}^{-1}\widehat{\nabla}_N^*\mathcal{I}^*\boldsymbol{\mathcal{F}}_{2N-1}\boldsymbol{A}E. \tag{9b}$$

for the preconditioning of (8).

3. FFT-based methods with low-rank approximations

Applying low-rank approximation techniques is of particular interest for problems with a huge number of degrees of freedom. The low-rank approximations can not only furnish a posterior data compression of the solution array, but also reduce computational complexity by exploiting low-rank format representations in the solution process. For the latter one needs some operations such as additions, element-wise multiplication, and the fast Fourier transform (FFT) to be implemented on tensors in low-rank format. In this section we introduce an FFT-based solution process incorporating low-rank representations of tensors. In the following Section 3.1, the low-rank approximation formats are summarised along the corresponding operations; details can be found in textbooks or in Appendix. Then the application of low-rank approximation for the Fourier–Galerkin method is described and discussed in Section 3.2, and the suitable linear solvers in Section 3.3.

3.1. Overview of low-rank formats

Here we give a brief introduction of three types of low-rank tensors that are applied in this work, they are of canonical polyadic (CP), Tucker, and tensor train format respectively. The CP format is only used for tensors of order two because of its intrinsic difficulty in finding optimal approximation for tensors with higher order. The necessity and impact of rank truncation is also emphasised. Interested readers are provided by more details about the operations on the low-rank tensors in Appendix.

3.1.1. Canonical polyadic format

A CP r-term approximation of a tensor $v \in \mathbb{K}^{N_1\times\cdots\times N_d}$ (the field $\mathbb{K}$ is $\mathbb{R}$ or $\mathbb{C}$) is a sum of r rank-1 tensors. In this work the CP format is only used for tensors of order two ($d = 2$), i.e. matrices. In this case the representation has the form:

$$v \approx \widetilde{v} = \sum_{i=1}^{r} c[i]b^{(1)}[i] \otimes b^{(2)}[i],$$

where $c \in \mathbb{R}^r$ stores the coefficients, and $b^{(j)} \in \mathbb{K}^{r\times N_j}$. A low-rank representation for order-2 tensors (matrices) can be obtained by various matrix factorising methods, among which the Singular Value Decomposition (SVD) is

prominent as it provides a factorisation that minimises the Frobenius-norm error of an r-term approximation. The level of compression (reduction of memory requirements) depends on the rank r. In order to find a solution in such a low-rank form, it requires to perform several operations occurring in the Fourier–Galerkin method, particularly the FFT and element-wise multiplication.

The linearity and the tensor-product structure of the Fourier transform facilitates to express d-dimensional FFT of a tensor (of order d) as the sum of tensor products of 1-dimensional FFTs, i.e.,

$$\mathcal{F}_N(\widetilde{v}) = \sum_{i=1}^{r} c[i]\mathcal{F}_{N_1}(b^{(1)}[i]) \otimes \mathcal{F}_{N_2}(b^{(2)}[i]).$$

For the same number of tensor components in all directions j, i.e. $N_j = N$, this d-dimensional FFT algorithm has a complexity $\mathcal{O}(drN \log N)$, which scales much better than the $\mathcal{O}(dN^d \log N)$ for the full tensor, when the rank r is kept low. Note that this operation does not change the rank of a transformed tensor.

Another two operations that occur in the Fourier–Galerkin method are the addition and the element-wise (Hadamard) multiplication of two tensors in low-rank format. In the case of the CP format it is computed as:

$$\widetilde{v} + \widetilde{w} = \sum_{i=1}^{r} c_v[i] \left(b_v^{(1)}[i] \otimes b_v^{(2)}[i]\right) + \sum_{k=1}^{s} c_w[k] \left(b_w^{(1)}[k] \otimes b_w^{(2)}[k]\right),$$

$$\widetilde{v} \odot \widetilde{w} = \sum_{i=1}^{r}\sum_{k=1}^{s} c_v[i]c_w[k] \left(b_v^{(1)}[i] \odot b_w^{(1)}[k]\right) \otimes \left(b_v^{(2)}[i] \odot b_w^{(2)}[k]\right).$$

While addition of two tensor costs no floating point operations and only requires more memory, the element-wise multiplication has a complexity of $\mathcal{O}(rsdN)$, which is significantly less than the $\mathcal{O}(N^d)$-complexity for full tensors, especially when the ranks r and s are much smaller than N.

3.1.2. Tucker format

The decomposition of higher order tensors has many variants. The Tucker format representation is linked to the definition of a tensor subspace $\mathcal{V} = \bigotimes_{j=1}^{d} \mathcal{V}^j$ where $\mathcal{V}^j$ is a subspace of $\mathbb{R}^{N_j}$ generated by the span of vectors $\{b^{(j)}[i] \mid i = 1, \ldots, r_j\}$; these vectors, which may be a frame, are typically chosen as an orthogonal or orthonormal basis. The Tucker format is then a linear combination of tensor products of all possible combinations of basis vectors in different directions, i.e.

$$v \approx \sum_{i_1=1}^{r_1} \cdots \sum_{i_d=1}^{r_d} c[i_1, \ldots, i_d] \bigotimes_{j=1}^{d} b^{(j)}[i_j] \in \mathbb{R}^N,$$

where the core $c \in \bigotimes_{\alpha=1}^{d} \mathbb{R}^{r_\alpha}$ is a tensor of order d. The CP format is then a special form of the Tucker format with a diagonal core. Note that naturally there can be different number of basis vectors in different directions.

3.1.3. The Tensor train (TT) format

The tensor train is another format which is suitable for the decomposition of higher order tensors. The idea is based on recursive decompositions done sequentially along the *tensor's* individual spatial dimensions. For tensors of order 3, the decomposition of the tensor of size $N \times N \times N$ is computed in two steps. Using the standard SVD algorithm, the decomposition is first computed on the reshaped matrix of size $N \times N^2$. It is followed by the decomposition of the reshaped right-singular vectors, i.e. of the matrix of size $N \times N$. The above recursive decomposition thus leads to

$$v = \sum_{i_1=1}^{r_1}\sum_{i_2=1}^{r_2} b^{(1)}[1, :, i_1] \otimes b^{(2)}[i_1, :, i_2] \otimes b^{(3)}[i_2, :, 1], \tag{10}$$

where the vectors $b^{(j)}[i_{j-1}, :, i_j] \in \mathbb{R}^{N_j}$ are vectors in direction j. The tensor's components can be explicitly written for $\boldsymbol{k} = (k_1, k_2, k_3)$ as

$$v[\boldsymbol{k}] = \sum_{i_1=1}^{r_1}\sum_{i_2=1}^{r_2} b^{(1)}[1, k_1, i_1]b^{(2)}[i_1, k_2, i_2]b^{(3)}[i_2, k_3, 1].$$

For $d = 2$ it is identical to the CP format.

The tensor train format in (10) is again expressed as a linear combination of rank one tensors, on which a d-dimensional FFT can be applied through a series of one-dimensional FFTs on the train *carriages* along the second index, i.e. applied on the vectors $\boldsymbol{b}^{(j)}[i_{j-1},:,i_j] \in \mathbb{K}^{N_j}$ for all i_j. The operations addition or element-wise multiplication are discussed in detail in Appendix A.3.

3.1.4. Rank truncation

Rank truncation is the way to reduce computational complexity by a reasonable compromise in the precision of the low-rank approximations. It is particularly necessitated by the fact that operations on low-rank tensors like addition and element-wise multiplication usually inflate the representation rank $\boldsymbol{r}$, potentially at a very fast rate, which is detrimental to a fast computation. On the other hand, in the resulted representation, a large part of the $\boldsymbol{r}$ terms are not essential and can be given up without or with minor loss of accuracy, if done correctly.

Rank truncations of tensors in the three low-rank formats are all based on QR decomposition, SVD, or high order SVD (HOSVD) [53], which provide optimal or suboptimal truncations and error estimates.

Other truncations are also possible. Particularly, the element-wise multiplication of two tensors with rank r results in a tensor of rank $s = r^2$, which is truncated with computational complexity $O(Ns^2)$ for CP and Tucker and $O(Ns^3)$ for TT format. In case of higher rank r of the original tensors, the truncation becomes a computational bottleneck. To speed up the basis orthogonalisation procedure, the basis with relatively small norms can also be removed before the orthogonalisation to trade accuracy for efficiency. This is usually beneficial in an iterative solver.

We supplement a more detailed introduction to the truncation procedure in each low-rank format in Appendix.

3.2. Applications of low-rank approximations on the linear systems

Here, we discuss the application of low-rank formats to speed up solvers of the linear systems (9), which are again stated here for the reader's convenience

$$\overbrace{\widehat{\boldsymbol{P}}^{-1}\widehat{\nabla}_N^*\boldsymbol{\mathcal{F}}_N\widetilde{\boldsymbol{A}}\boldsymbol{\mathcal{F}}_N^{-1}\widehat{\nabla}_N}^{\widetilde{C}}\widehat{u} = \overbrace{-\widehat{\boldsymbol{P}}^{-1}\widehat{\nabla}_N^*\boldsymbol{\mathcal{F}}_N\widetilde{\boldsymbol{A}}E}^{\widetilde{b}}, \tag{11a}$$

$$\underbrace{\widehat{\boldsymbol{P}}^{-1}\widehat{\nabla}_N^*\mathcal{I}^*\boldsymbol{\mathcal{F}}_{2N-1}\boldsymbol{A}\boldsymbol{\mathcal{F}}_{2N-1}^{-1}\mathcal{I}\widehat{\nabla}_N}_{C}\widehat{u} = \underbrace{-\widehat{\boldsymbol{P}}^{-1}\widehat{\nabla}_N^*\mathcal{I}^*\boldsymbol{\mathcal{F}}_{2N-1}\boldsymbol{A}E}_{b}. \tag{11b}$$

The solution vector u (or their Fourier coefficients $\widehat{u} = \mathcal{F}_N u$) stores the values of the trigonometric polynomial on the d-dimensional regular discretisation grid. Therefore the solution vector can be naturally represented as a tensor of order d, which allows a low-rank representation. In order to avoid the computation of the full tensor and its decomposition, the low-rank tensor $\widehat{u}$ is computed by a suitable iterative solver introduced in 3.3. It requires to perform matrix vector multiplication for a low-rank tensor v, which is approximated as

$$\widetilde{C}v \approx \mathcal{T}\widehat{\boldsymbol{P}}^{-1}\mathcal{T}\widehat{\nabla}_N^*\boldsymbol{\mathcal{F}}_N\mathcal{T}\widetilde{\boldsymbol{A}}\boldsymbol{\mathcal{F}}_N^{-1}\widehat{\nabla}_N v, \tag{12a}$$

$$Cv \approx \mathcal{T}\widehat{\boldsymbol{P}}^{-1}\mathcal{T}\widehat{\nabla}_N^*\mathcal{I}^*\boldsymbol{\mathcal{F}}_{2N-1}\mathcal{T}\boldsymbol{A}\boldsymbol{\mathcal{F}}_{2N-1}^{-1}\mathcal{I}\widehat{\nabla}_N v; \tag{12b}$$

similarly, the right-hand side of the linear systems is approximated by a low-rank tensors

$$\widetilde{b} = -\mathcal{T}\widehat{\boldsymbol{P}}^{-1}\widehat{\nabla}_N^*\boldsymbol{\mathcal{F}}_N\widetilde{\boldsymbol{A}}E, \tag{12c}$$

$$b = -\mathcal{T}\widehat{\boldsymbol{P}}^{-1}\widehat{\nabla}_N^*\mathcal{I}^*\boldsymbol{\mathcal{F}}_{2N-1}\boldsymbol{A}E. \tag{12d}$$

These approximations involve several operations in low-rank formats such as differentiation, divergence, Fourier transform, and the truncation operator $\mathcal{T}$, which keeps the rank $\boldsymbol{r}$ at an affordable level. The operations are tabulated in Table 1 together with the corresponding implementations in low-rank format and their impact on the rank $\boldsymbol{r}$.

Since the material coefficients $\widetilde{\boldsymbol{A}}$, $\boldsymbol{A}$ and the preconditioner P^{-1} are diagonal or block-diagonal for non-isotropic material coefficients, the related matrix–vector multiplications are implemented as element-wise multiplications, which inevitably inflates the representation rank of the tensors in low-rank format. We apply a rank truncation after each multiplication to keep the computational complexity at a relatively low level, while maintaining reasonable accuracy in the solution.

Table 1
Operations and their implementations in low-rank formats.

Operation	Low-rank tensor implementation	Rank $\boldsymbol{r}$
Differentiation (gradient)	Element-wise multiplication	Remains unchanged
Divergence	Element-wise multiplication and addition	Is increased
Evaluation of material law	Element-wise multiplication	Is increased
d-dimensional FFT	Series of 1D FFTs	Remains unchanged
Preconditioning	Element-wise multiplication	Is increased

The application of the gradient and divergence in Fourier space is also implemented as element-wise multiplications. The differentiation operator for trigonometric polynomials is by nature a rank-1 tensor in the form

$$\widehat{\nabla}_{\boldsymbol{N}} = [2\pi \mathrm{i}\boldsymbol{K}_1 \otimes \mathbf{1} \otimes \mathbf{1}, \mathbf{1} \otimes 2\pi \mathrm{i}\boldsymbol{K}_2 \otimes \mathbf{1}, \mathbf{1} \otimes \mathbf{1} \otimes 2\pi \mathrm{i}\boldsymbol{K}_3]$$

in the 3D setting, where $\boldsymbol{K}_\alpha = (k \in \mathbb{Z};\ |k| < N/2)$ is a vector of all discrete frequencies in direction α. So the corresponding element-wise multiplication keeps the rank of tensors unchanged. However, for the divergence the contraction along the first component of $\widehat{\nabla}_{\boldsymbol{N}}$ is provided by the operation addition of two low-rank formats, which increases the rank, and hence a truncation has to be performed.

The last operation that occurs in the system is the d-dimensional fast Fourier transform (FFT) which is efficiently evaluated using 1-dimensional FFTs. Moreover the rank of the tensor remains the same in this operation.

3.3. Linear solvers

For the full solver we have used preconditioned conjugate gradients, which is considered to be the best available solver for FFT-based homogenisation [17,52]. However, the linear systems with low-rank approximations require solvers that are insensitive to small perturbations, as the matrix–vector product is computed only approximately, due to the truncation of tensors. Therefore conjugate gradient method that builds on the orthogonalisation of Krylov subspace vectors using a short-term recurrence relation is inappropriate.

The systems with low-rank approximations are solved here with minimal residual iteration [54] which is closely related to Richardson iteration. The latter is well established in the FFT-based community, as it corresponds to the original Moulinec–Suquet algorithm. Both methods solve the linear system $\boldsymbol{C}\boldsymbol{u} = \boldsymbol{d}$, see (11) and (12) for details, by the iteration

$$\boldsymbol{u}_{(i+1)} = \boldsymbol{u}_{(i)} + \omega \underbrace{(\boldsymbol{d} - \boldsymbol{C}\boldsymbol{u}_{(i)})}_{\boldsymbol{r}_{(i)}} = (\boldsymbol{I} - \omega\boldsymbol{C})\boldsymbol{u}_{(i)} + \omega\boldsymbol{d}.$$

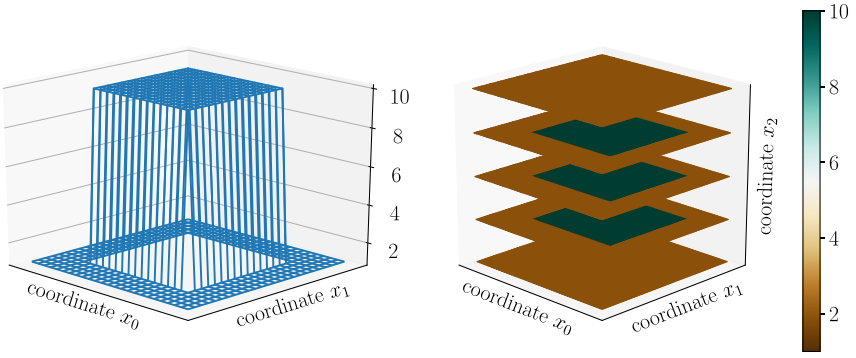
In the Richardson iteration, the parameter ω is chosen such that the iteration matrix $(\boldsymbol{I} - \omega\boldsymbol{C})$ has a norm smaller than one to guarantee convergence. A fixed value ω is set on the basis of a priori knowledge about the extreme eigenvalues of the system matrix $\boldsymbol{C}$, i.e.

$$\omega = \frac{2}{\lambda_{\min}(\boldsymbol{C}) + \lambda_{\max}(\boldsymbol{C})},$$

because it satisfies the minimal norm of the iterative matrix as proposed in [50] for FFT-based homogenisation. Here $\lambda_{\min}(\boldsymbol{C})$ denotes the smallest positive eigenvalue, as the system matrix is only positive semidefinite. In particular, the linear systems in (9) contain one zero eigenvalue corresponding to the constant fields, while the linear systems that are formulated in traditional FFT-based homogenisation for gradients fields contain many zero eigenvalues corresponding to the eigenspace composed of divergence-free fields. In both cases the solver produces the solution in the space of compatible fields.

In the minimal residual iteration, the parameter ω is chosen at each iteration as the minimiser of the next residual $\boldsymbol{r}_{(i+1)}$ over all increments of $\boldsymbol{u}$ in the direction of $\boldsymbol{r}_{(i)}$., i.e.

$$\boldsymbol{u}_{(i+1)} = \boldsymbol{u}_{(i)} + \omega_{(i)}\boldsymbol{r}_{(i)}, \quad \text{with } \omega_{(i)} = \frac{(\boldsymbol{C}\,\boldsymbol{r}_{(i)}, \boldsymbol{r}_{(i)})}{\|\boldsymbol{C}\,\boldsymbol{r}_{(i)}\|^2}.$$



(a) Square inclusion (□) in 2D (b) Square (cube) inclusion (□) in 3D

Fig. 1. Material coefficients (13) of the square and the cube inclusion defined by (14).

We adopt the latter method in this work, because of our observation that the minimal residual iteration is more robust than Richardson iteration, for which we have observed a divergence when a massive truncation has been used during the iterations. For a low-rank approximation of a solution vector, note that the solver has to deal with the matrix vector product $\mathsf{C}u_{(i)}$, which is computed only approximately (12) to limit the growth of the solution rank. The rank also grows by the operation addition during the iteration. Therefore, a truncation is included at each step of the low-rank variant of the minimal residual iteration, i.e.

$$u_{(i+1)} = \mathcal{T}[u_{(i)} + \omega_{(i)}(\mathsf{d} - \mathsf{C}u_{(i)})].$$

4. Numerical results

The methodology described in the previous sections is tested on several numerical examples with material parameters defined in Section 4.1. We compare two numerical homogenisation schemes: the Fourier–Galerkin method with numerical integration (GaNi) and a version with exact integration (Ga), described in Sections 2.2.3 and 2.2.2. The preconditioned linear systems stated in (9) are solved by conjugate gradient method. The same systems that are equipped with low-rank tensor approximations are solved by the minimal residual iteration, which is discussed in Section 3.3.

The numerical results were calculated using software FFTHomPy (FFT-based Homogenisation in Python), which is freely available at https://github.com/vondrejc/FFTHomPy; the software contains examples, which are described in the following sections.

4.1. Material parameters

Here, we present two material examples on which we did numerical tests. The first is defined as

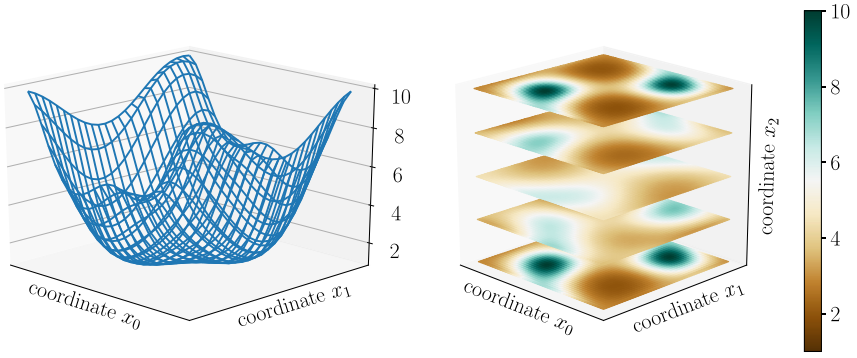
$$\boldsymbol{A}_{\square}(\boldsymbol{x}) = \boldsymbol{I}(1 + \rho\chi(\boldsymbol{x})) \tag{13}$$

where $\boldsymbol{I} \in \mathbb{R}^{d\times d}$ is the identity matrix and the parameter $\rho = 10$ corresponds to a material contrast. The function $\chi : \mathcal{Y} \to \mathbb{R}^d$ describing the topology of the inclusions is defined on a unit cell $\mathcal{Y} = (-\frac{1}{2}, \frac{1}{2})^d$ as

$$\chi(\boldsymbol{x}) = \begin{cases} 1 & \text{for } \boldsymbol{x} \text{ such that } x_i < 0.3 \text{ for } i = 1, \dots, d, \\ 0 & \text{otherwise,} \end{cases} \tag{14}$$

which is also depicted in 2D in Fig. 1. The corresponding low-rank approximations have rank 2 for all three formats (CP, Tucker, tensor train).

As a second example, one sample of a stochastic material has been obtained using the truncated Karhunen–Loève expansion [55] of the squared exponential Matérn covariance function [56]. In order to obtain positive definite



(a) Stochastic material (S) in 2D (b) Stochastic material (S) in 3D

Fig. 2. One sample of the stochastic material defined by (15).

material coefficients, the exponential function has been applied on the expansion, which leads to the following form

$$\boldsymbol{A}_{\mathrm{S}}(\boldsymbol{x}) = \boldsymbol{I} \exp\big(C + D \sum_{\boldsymbol{k}\in I} c[k]\varphi^{\boldsymbol{k}}(\boldsymbol{x})\big). \tag{15}$$

The most important modes of the expansion have been selected (20 modes in $2D$ and 26 modes in 3D) and the corresponding frequencies are collected in the index set I. The coefficients $c[k]$ for $k \in I$ have been sampled from uniform distribution on the interval $[-0.5, 0.5]$. The constants C and D scale the material coefficients such that the minimal eigenvalue of $\boldsymbol{A}$ is 1, and the maximal 10. The particular sample that is used for the computation is plotted in Fig. 2. The material coefficients were approximated in low-rank formats with a rank set to 10. For a comparison to the full solution, the full material coefficients have been recovered in order to compute exactly the same problem.

All the numerical problems have been computed with the same number of discretisation grids in each direction $\boldsymbol{N} = [N, \ldots, N] \in \mathbb{R}^d$.

4.2. Behaviour of linear systems during iterations

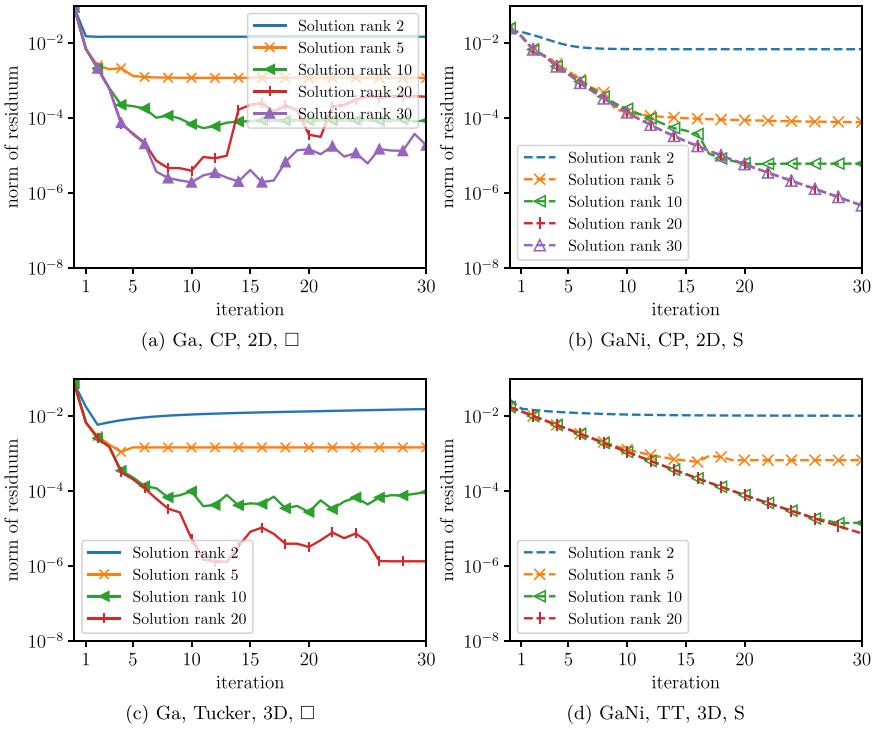
The evolution of the norm during the minimal residual iteration is investigated because it describes well the character of the low-rank approximations. The numerical results in Fig. 3 depict the Euclidean norm of the residuum $r = d - Cu_{(i)}$

$$\|r\| = \Big(\sum_{\boldsymbol{k}\in\mathbb{Z}_{\boldsymbol{N}}} \big|r[\boldsymbol{k}]\big|^2\Big)^{\frac{1}{2}}$$

because it corresponds to the L^2-norm of the corresponding trigonometric polynomial. Note that since the problem is solved in Fourier space, the residuum components agree with the Fourier coefficients of the corresponding trigonometric polynomial.

Although the truncation of the growing tensor's rank can be provided by a tolerance to an approximation error, it is difficult to set up the parameters properly during the solver. Particularly it may happen that the rank significantly increases resulting in unnecessary computational demands, especially when the tensors are far away from the solution. Therefore the truncation has been performed to a fixed rank. The solution which is from a large dimensional space $\mathbb{R}^{\boldsymbol{N}}$ with the dimension $\prod_{\alpha=1}^{d} N_\alpha$ is approximated with a significantly smaller number of parameters. Therefore there is always a residual error which can be diminished only by an increasing rank of the low-rank formats. Note that the rank-one tensors occurring in all three low-rank formats are automatically computed by a solver and are thus suboptimal global basis vectors for the particular problem. Therefore the method can be seen as a model order reduction technique.

From the results in Fig. 3, we can observe that solutions with higher rank have larger potential in reducing the norm of residuum regardless of the discretisation method (Ga and GaNi), material problem (□ and S), or the



(a) Ga, CP, 2D, □ (b) GaNi, CP, 2D, S

(c) Ga, Tucker, 3D, □ (d) GaNi, TT, 3D, S

Fig. 3. Evolution of the norm of residua during minimal residuum iteration; computed in 2D for $N = 1215$ and in 3D for $N = 135$.

low-rank format (CP, Tucker, TT). This proposes a rank adapting solver that starts with a lower solution rank and increases the rank during the iterations. We also notice that the norms of residuum during iterations decrease with higher rate for the problem with the square inclusion (material □), however, the rate is more stable for the material S. Although, the material □ was systematically computed with GaNi method and material S with Ga, which is in accordance with the recommendation in [57], the discretisation method has no influence on the character of the behaviour during iterations. These findings are in agreement with [37] analysing the stochastic linear systems and solvers approximated with low-rank approximations.

Note that the computation of the Frobenius norm of tensors in Tucker format is computationally demanding. Therefore, we have used the equivalent Frobenius norm of the Tucker's core, which can be computed much faster.

4.3. Algebraic error of the low-rank approximations

In Fig. 4, the approximation properties of the low-rank formats are depicted. As a criterion, the relative algebraic error between the homogenised properties of low-rank solution $A_{\mathrm{H},\boldsymbol{N},r}$ and of the full solution $A_{\mathrm{H},\boldsymbol{N}}$ has been used, i.e.

$$\text{relative error} = \frac{A_{\mathrm{H},\boldsymbol{N}} - A_{\mathrm{H},\boldsymbol{N},r}}{A_{\mathrm{H},\boldsymbol{N}}}. \tag{16}$$

This is chosen because the error in the homogenised properties corresponds to the square of the energetic semi-norm (norm on zero-mean fields) of the algebraic error between the full solution and the low-rank approximation

$$\|u_{\boldsymbol{N}} - u_{\boldsymbol{N},r}\|_A^2 = a\big(\nabla u_{\boldsymbol{N}} - \nabla u_{\boldsymbol{N},r}, \nabla u_{\boldsymbol{N}} - \nabla u_{\boldsymbol{N},r}\big) = A_{\mathrm{H},\boldsymbol{N},r} - A_{\mathrm{H},\boldsymbol{N}};$$

for the derivation see [57, Appendix D]. We also note that the full solution $u_{\boldsymbol{N}}$ has been computed using conjugate gradients with high accuracy (tolerance 10^{-8} on the norm of the residuum) to obtain a solution that is close to the exact one. The low-rank solution has been obtained from minimal residual iteration, which was stopped when the residuum failed to be decreased. The minimal residual iteration was used to provide low-rank solution with the minimal norm of residuum.

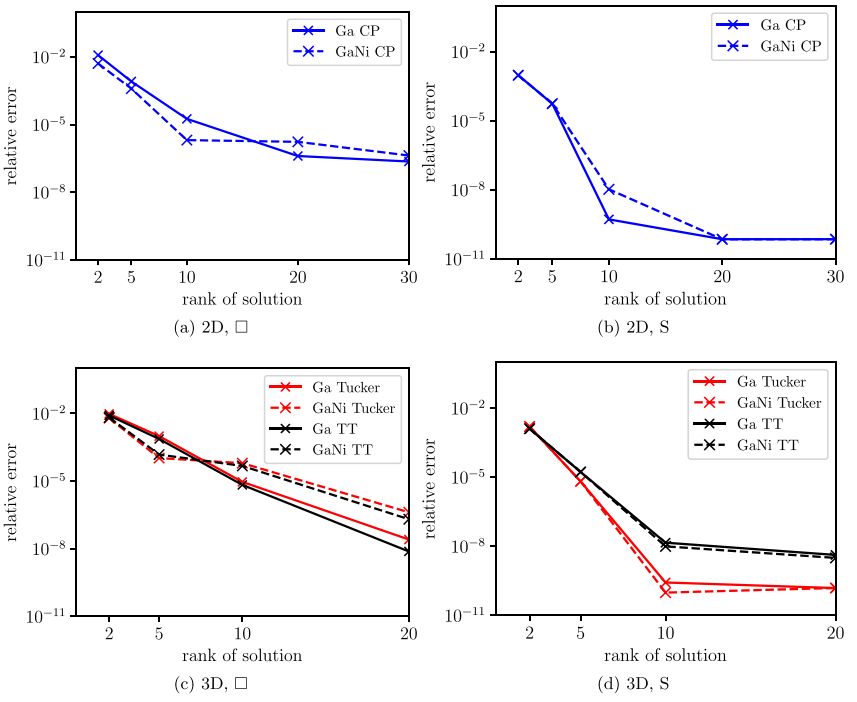


Fig. 4. Relative errors (16) of low-rank solutions computed in 2D for $N = 1215$ and in 3D for $N = 135$.

We can observe that the results are again similar regardless of the discretisation method (Ga and GaNi), material problem (□ and S), or the low-rank format (CP, Tucker, TT). An increase in the solution rank leads to a significant reduction of the relative error. However, the low-rank approximations of the material S reach the threshold error corresponding to the full approximate solution obtained from the conjugate gradients. It also shows that the low-rank method is more accurate for a problem with continuous material property (material S) than for the one with discontinuous coefficients (material □).

4.4. Memory and computational efficiencies

Here, we discuss the computational and memory requirements to resolve the linear system using low-rank approximations. Additionally to the previous examples, the CPU times and approximation properties of low-rank formats were tested for an anisotropic material. The heterogeneous material coefficients $\boldsymbol{A}_{\square}$ and $\boldsymbol{A}_{\mathrm{S}}$ were modified by adding a spatially constant anisotropic material tensor $\boldsymbol{B}$, i.e.

$$\widetilde{\boldsymbol{A}}_{\bullet}(\boldsymbol{x}) = \boldsymbol{A}_{\bullet}(\boldsymbol{x}) + \boldsymbol{B},$$

where the matrices

$$\boldsymbol{B} = \begin{pmatrix} 5.5 & -4.5 \\ -4.5 & 5.5 \end{pmatrix}, \quad \boldsymbol{B} = \begin{pmatrix} 4.25 & -3.25 & -1.25\sqrt{2} \\ -3.25 & 4.25 & 1.25\sqrt{2} \\ -1.25\sqrt{2} & 1.25\sqrt{2} & 7.5 \end{pmatrix}$$

have eigenvalues $(1, 10)$ in 2D and $(1, 5, 10)$ in 3D.

As we are using several low-rank formats and several operations on them, the computational complexities and memory requirements are summarised in Tables 2 and 3. The memory requirements of the FFT-based systems are controlled by memory requirements for material coefficients, preconditioner, solution vector, and possibly other vectors needed to store as a requirement of the linear solver. Provided that the ranks are kept small, the memory of low-rank solvers scales linearly with N, while full solver scales with N^d, which makes the method effective particularly for tensor with high order.

Table 2
Asymptotic computational complexities in terms of floating point multiplications. The operations are performed on full tensors of order d and shape $(N, \dots, N)$, and on the same tensors in their CP, Tucker, and tensor-train (TT) formats with maximum rank r and s (s for the second operand in a binary operation).

Formats	Operations		
	Element-wise product	FFT_d	Truncation
Full	N^d	$\mathcal{O}(N^d \log N)$	–
CP	$dNrs$	$\mathcal{O}(dNr \log N)$	$\mathcal{O}(dNr^2)$
Tucker	$dNrs + r^d s^d$	$\mathcal{O}(dNr \log N)$	$\mathcal{O}(dNr^2 + r^{d+1})$
TT	$dNr^2 s^2$	$\mathcal{O}(dNr^2 \log N)$	$\mathcal{O}(dNr^3)$

Table 3
Memory requirements to store tensors of order d with shape $(N, \dots, N)$ for full, CP, Tucker, and tensor-train (TT) formats with maximum rank r.

Format	Memory requirements
Full	N^d
CP	dNr
Tucker	$dNr + r^d$
TT	$2Nr + (d-2)Nr^2$

Table 4
Rank r of low-rank solutions that reach the same accuracy as the full solution for various values of N, for isotropic and anisotropic material □. The full solver has been computed with grid size $(N, \dots, N)$ while sparse solver with $(3N, \dots, 3N)$. The stopping criterion in the conjugate gradient method for the full solver was set to 10^{-6} on the norm of residuum.

	2D, □				3D, □				
N	45	135	405	1215	5	15	45	135	175
r (isotropic)	3	3	5	7	3	3	3	5	5
r (anisotropic)	5	11	21	31	3	3	5	11	11

We compare the CPU time of full and low-rank solvers for homogenisation with exact integration (Ga) on the same level of accuracy measured by the energetic norm. It is achieved by the following procedure. The reference full solution was computed using conjugate gradient method on the regular grid $(N, \dots, N)$ with the tolerance 10^{-6} on the norms of residua. In order to achieve the same accuracy as the full solution, the low-rank solver was run on a bigger grid $(\alpha N, \dots, \alpha N)$ with the multiplier $\alpha = 3$. The rank of low-rank approximations was increased step-by-step until it achieved a required error tolerance defined in (16). The iterations of the low-rank solver (for a given rank) are stopped when the residuum fails to decrease. This procedure, which creates a great possibility for a rank reduction in the low-rank solution, is applicable only for problems that allow an exact integration of material coefficients (here material □).

The results in Fig. 5 show that the CPU time scales as N^d for a full solution, and almost linearly for low-rank solutions on the isotropic material □. In the anisotropic cases the time costs of low-rank solutions are relatively higher but still cheaper than that of the full solution. The difference in these two cases is due to the different ranks of the low-rank solutions. For isotropic material □, the solution rank increases only slowly with N, while for its anisotropic counterpart the rank increases at a faster rate (as tabulated in Table 4). In general, the results show that the low-rank solver is significantly faster for larger N, despite being run on a larger computational grid.

We also did the comparison of both solvers for the isotropic and anisotropic material S. However, the smooth material S is better suited for the homogenisation with the numerical integration (GaNi), see the comparison in [57]. Therefore, the comparison of the solvers is run on the same discretisation grid. The ranks of the low-rank solution are chosen such that it achieves a relative error (as defined in (16)) 10^{-3} or 10^{-6}. The ranks remain stable when N increases, which makes the CPU time of the low-rank solver almost linear in N, as shown in Fig. 6.

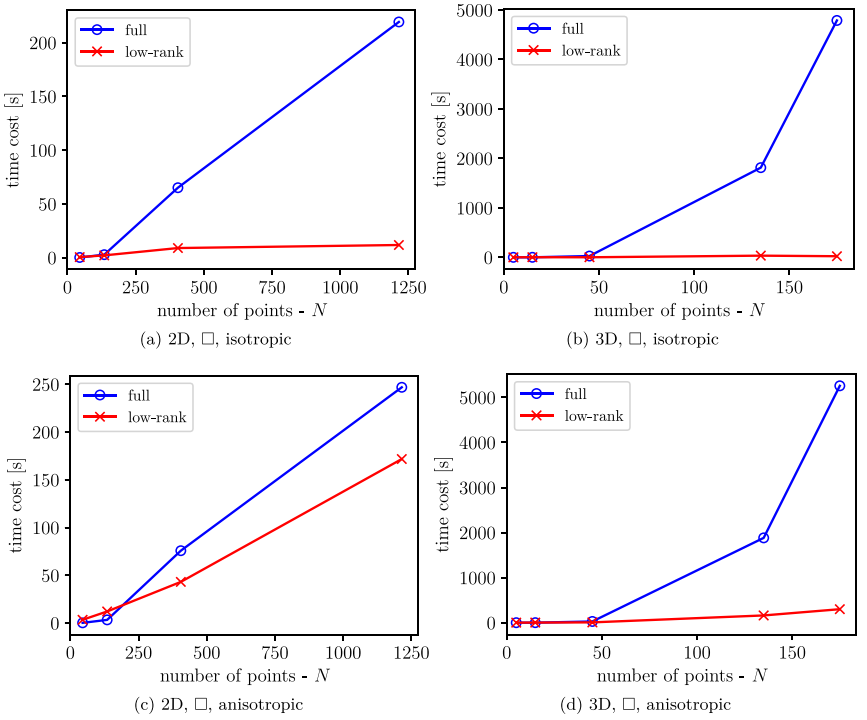


Fig. 5. The CPU time of Ga solver to solve the problem with isotropic (1st row) and anisotropic (2nd row) material □. The full solution has been computed on a grid of size $(N, \ldots, N)$ while the low-rank solution on the grid $(3N, \ldots, 3N)$ with various solution ranks to achieve the same level of accuracy as the full scheme. The stopping criterion of conjugate gradients for the full solver was set to 10^{-6} on the norm of residuum.

5. Conclusion

This paper is focused on the acceleration of Fourier–Galerkin methods using low-rank tensor approximations for spatially 2-dimensional and 3-dimensional problems of numerical homogenisation. The efficiency of this approach builds on the incorporation of fast Fourier transform (FFT) and low-rank tensor approximation into the iterative linear solvers. The computational complexity is reduced to be quasilinear in the size of the discretisation and linear in spatial dimension d, since on a low-rank tensor of order d, the d-dimensional FFT can be performed as a series of one-dimensional FFTs. In this paper three formats – canonical polyadic (CP), Tucker, and tensor train (TT) – have been considered, and all of them show similar advantage in saving the computational cost.

The main results are summarised as the following:

- The incorporation of low-rank tensor approximations leads to a significant reduction of memory and computational cost in the solution of the homogenisation problems.
- The method is more suitable for material coefficients with relatively smaller rank. The low-rank approximation solvers computationally benefit from the better asymptotic behaviour, see Tables 2 and 3. The advantage is accentuated for problems of a higher spatial dimension d leading to tensors with order d.
- The low-rank approximation can be seen as a model order reduction technique.

Since the low-rank approximation provides a significant memory reduction it allows to compute the solution on a finer grid. Therefore, the proposed method based on low-rank approximation may provide more accurate solution than the conventional method based on full tensors, especially when the material is of a relatively small rank.

Acknowledgements

Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) — project number MA2236/27-1. Martin Ladecký was supported by the Czech Science Foundation through projects No. GAČR

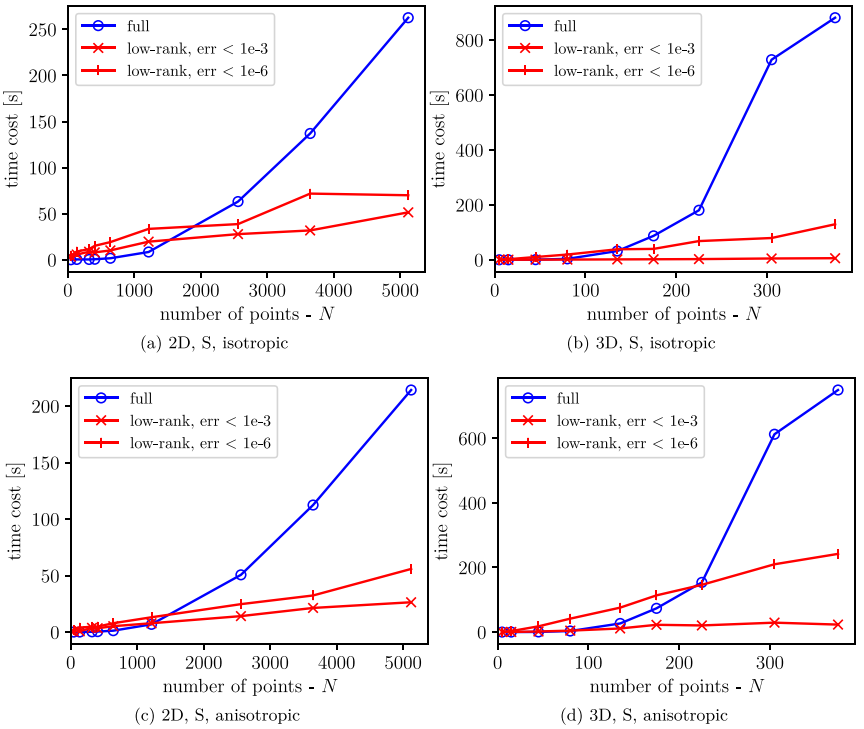


Fig. 6. The CPU time of GaNi solver to solve the problem with isotropic (1st row) and anisotropic (2nd row) material S. Both the full and low-rank solutions are computed on the same grid. The stopping criterion of conjugate gradients for the full solver was set to 10^{-6} on the norm of residuum. The minimal residuum iteration for the low-rank approximation was computed with various ranks to achieve a required error tolerance defined in (16).

17-04150J, by the Center of Advanced Applied Sciences (CAAS), Czech Republic, financially supported by the European Regional Development Fund (through project No. CZ.02.1.01/0.0/0.0/16_019/0000778), and by the Grant Agency of the CTU in Prague, Czech Republic, grant No. SGS19/002/OHK1/1T/11.

Appendix. Low-rank tensor approximations

Here we provide more details of the low-rank tensor approximations techniques utilised in this paper. This includes the approximation in CP, Tucker and tensor train formats.

A.1. The canonical polyadic format

A canonical polyadic (CP) or r-term representation v_r of a tensor $v \in \mathbb{K}^{N_1 \times \cdots \times N_d}$ ($\mathbb{K}$ is either $\mathbb{R}$ or $\mathbb{C}$) is a sum of r rank-1 tensors, i.e.

$$v \approx v_r = \sum_{i=1}^{r} c[i] \bigotimes_{j=1}^{d} b^{(j)}[i] \tag{17}$$

with $b^{(j)} \in \mathbb{K}^{r \times N_j}$ and $\bigotimes$ denotes tensor product. This format has linear storage size $r \sum_{j=1}^{d} N_j$. But for $d \geq 3$ and a given r, the construction of an error minimising v_r is not always feasible [53, Proposition 9.10] because the space of CP format tensor with fixed r is not closed [53, Lemma 9.11].

A.1.1. Element-wise multiplication

The element-wise (Hadamard) product of two tensors of ranks r and s in CP format is computed as:

$$v_r \odot w_s = \sum_{i=1}^{r}\sum_{k=1}^{s} c_v[i]c_w[k]\bigotimes_{j=1}^{d}\left(b_v^{(j)}[i]\odot b_w^{(j)}[k]\right).$$

This operation has complexity $rs\sum_{j=1}^{d} N_j$ and the product has a new rank rs.

A.1.2. Fourier transform

Due to the linearity and tensor structure of Fourier transform, a d-dimensional Fourier transform $\mathcal{F}_{\boldsymbol{N}}$ of a tensor with mode sizes $\boldsymbol{N}\in\mathbb{N}^d$ in its CP format is broken down to a series of 1-d Fourier transform, i.e.,

$$\mathcal{F}_{\boldsymbol{N}}(v_r) = \sum_{i=1}^{r} c[i]\bigotimes_{j=1}^{d}\mathcal{F}_{N_j}(b^{(j)}[i]).$$

Hence a FFT on a CP tensor has a complexity $drN\log N$.

A.1.3. Rank truncation

Operations (e.g. element-wise multiplication) applied on tensors in CP format usually inflate the representation rank. This calls for a truncation to a prescribed rank or error tolerance.

For $d=2$, this reduction is done by rank truncation based on QR decomposition and singular value decomposition (SVD). Let the matrices $\boldsymbol{B}^{(j)}\in\mathbb{K}^{N_j\times r}$ collect the vectors $\{b^{(j)}[i]\}_{i=1}^{r}$ for the jth dimension, we have their re-orthogonalisations $\boldsymbol{B}^{(1)}=\boldsymbol{Q}^{(1)}\boldsymbol{R}^{(1)}$ and $\boldsymbol{B}^{(2)}=\boldsymbol{Q}^{(2)}\boldsymbol{R}^{(2)}$ by QR decompositions. A SVD $\boldsymbol{R}^{(1)}\boldsymbol{R}^{(2)}=\boldsymbol{U}^{(1)}\boldsymbol{\Sigma}(\boldsymbol{U}^{(2)})^{\top}$ facilitates the truncation. Suppose $\boldsymbol{U}_k^{(1)}$, $\boldsymbol{U}_k^{(2)}$ and $\boldsymbol{\Sigma}_k$ are the truncated ones with rank $k\le r$, the truncated form of the CP representation (17) is

$$v_k = \sum_{i=1}^{k} c[i]\widehat{b}^{(1)}[i]\otimes\widehat{b}^{(2)}[i]$$

where $\widehat{b}^{(1)}[i]$, $\widehat{b}^{(2)}[i]$ are the columns of $\boldsymbol{Q}^{(1)}\boldsymbol{U}_k^{(1)}$, $\boldsymbol{Q}^{(2)}\boldsymbol{U}_k^{(2)}$ respectively, and $c[i]$ are the diagonal entries of $\boldsymbol{\Sigma}_k$.

For $d\ge 3$, the k-rank form could be obtained by numerical error minimising procedures [53], e.g. Alternative Least-Squares method. But there is no guarantee that the procedures would converge, and if they would, there is no guarantee that they converge to the global optimum. This is due to the non-closedness of the set of rank-r CP tensors with $d\ge 3$.

A.2. Tucker format

A Tucker format representation (or tensor subspace representation) of a tensor $v\in\mathbb{K}^{N_1\times\cdots\times N_d}\in\mathcal{V}$ is a linear combination of frames (usually orthogonal bases) of the tensor space $\mathcal{V}$. Suppose $\mathcal{V}=\bigotimes_{j=1}^{d}\mathcal{V}^j$, the subspace $\mathcal{V}^j$ has basis vectors $\{b^{(j)}[i_j]\in\mathbb{K}^{N_j}: 1\le i_j\le r_j\}$ with ranks $\boldsymbol{r}=(r_1,\dots,r_d)$. The tensors $\bigotimes_{j=1}^{d} b^{(j)}[i_j]$ for all $1\le i_j\le r_j$ form the bases of the space $\mathcal{V}$. Then we have a unique coefficient $c[i_1,i_2,\dots,i_d]$ for every $v\in\mathcal{V}$ such that

$$v\approx v_{\boldsymbol{r}} = \sum_{i_1=1}^{r_1}\cdots\sum_{i_d=1}^{r_d} c_v[i_1,i_2,\dots,i_d]\bigotimes_{j=1}^{d} b_v^{(j)}[i_j], \tag{18}$$

where $c\in\mathbb{K}^{r_1\times\cdots\times r_d}$ is called the core tensor. Given any prescribed rank vector $\boldsymbol{r}$, an error minimising approximation $v_{\boldsymbol{r}}$ can be found by a *high-order singular value decomposition* (HOSVD) [58]. When the vectors $\{b^{(j)}[i_j]\in\mathbb{K}^{N_j}: 1\le i_j\le r_j\}$ form only a frame of the subspace $\mathcal{V}$ (e.g. after addition of two tensors), the core tensor is not unique, however, a representation with orthogonal bases can be obtained by applying QR decomposition to the frames and HOSVD to the accordingly updated core.

A.2.1. Element-wise multiplication

Let another Tucker format tensor with rank $\boldsymbol{s} \in \mathbb{N}^d$ be defined as

$$w_{\boldsymbol{s}} = \sum_{\boldsymbol{k}}^{\boldsymbol{s}} c_w[\boldsymbol{k}] \bigotimes_{j=1}^{d} b_w^{(j)}[k_j]$$

the element-wise (Hadamard) product of $v_{\boldsymbol{r}}$ and $w_{\boldsymbol{s}}$ has also a Tucker format

$$v_{\boldsymbol{r}} \odot w_{\boldsymbol{s}} = \sum_{\boldsymbol{l}}^{\boldsymbol{t}} c[\boldsymbol{l}] \bigotimes_{j=1}^{d} b^{(j)}[l_j]$$

where $\boldsymbol{t} = \boldsymbol{r} \odot \boldsymbol{s}$ and $c = c_v \otimes c_w$, i.e. the Kronecker product of the two coefficient tensors. So for any $1 \le j \le d$, the index l_j is related to i_j and k_j by $l_j = \overline{i_j k_j} = i_j r_j + k_j$, and $b^{(j)}$ is obtained from $b_v^{(j)}$ and $b_w^{(j)}$ through

$$b^{(j)}[l_j] = b^{(j)}[\overline{i_j k_j}] = b_v^{(j)}[i_j] \odot b_w^{(j)}[k_j] \quad \text{for} \quad 1 \le i_j \le r_j,\ 1 \le k_j \le s_j$$

Let $N = \max_i N_i$, $r = \max_i r_i$ and $s = \max_i s_i$, the computational complexity of the element-wise product is bounded by $dNrs + r^d s^d$, in which the first term is the cost for computing $\{b^{(j)}[l_j] : 1 \le l_j \le \boldsymbol{t}_j\}_{j=1}^d$, and the second for the Kronecker product of coefficient tensors.

A.2.2. Fourier transform

The Fourier transform of $v_{\boldsymbol{r}}$ is

$$\mathcal{F}_N(v_{\boldsymbol{r}}) = \sum_{\boldsymbol{i}}^{\boldsymbol{r}} c[\boldsymbol{i}] \bigotimes_{j=1}^{d} \mathcal{F}_{N_j}(b^{(j)}[i_j])$$

which only involves the basis vectors. If FFT is applied, the complexity is of order $\mathcal{O}(dr N \log N)$.

A.2.3. Rank truncation

The Tucker representation (18) can be obtained either by a HOSVD applied on a full tensor or by an operation (e.g. element-wise multiplication) over other Tucker operands. In the first case, an error minimising rank truncation is readily available due to the property of HOSVD:

$$\sigma_1^{(j)} \ge \sigma_2^{(j)} \ge \cdots \ge \sigma_{r_j}^{(j)}, \quad \text{for } j = 1, \ldots, d,$$

where $\sigma_{i_j}^{(j)}$ is the 2-norm of the i_j-th slice of the core tensor c cut on the jth dimension. If the truncation rank is $k_j < r_j$, the error of the truncated representation $v_{\boldsymbol{k}}$ is bounded by

$$\|v_{\boldsymbol{r}} - v_{\boldsymbol{k}}\| \le \left[\sum_{j=1}^{d} \sum_{i=k_j+1}^{r_j} (\sigma_i^{(j)})^2 \right]^{1/2}.$$

In the second case the bases $\{b^{(j)}\}_{j=1}^d$ have to be re-orthogonalised first, and then a HOSVD of the updated core tensor is to be made to facilitate the truncation as in the first case. This procedure [53, as detailed in] is analogues to the re-orthogonalisation and SVD for the 2D CP format representations, but with higher tensor order.

A.3. Tensor train format

A tensor train (TT) representation [39] of a tensor $v \in \mathbb{K}^{N_1 \times \cdots \times N_d}$ can be expressed as a series of consecutive contractions of order-3 tensors $b^{(j)} \in \mathbb{K}^{r_{j-1} \times N_j \times r_j}$ for $j = 1, \ldots, d$, which are the *carriages* of the tensor train. An equivalent expression in the form of tensor products is

$$v \approx v_{\boldsymbol{r}} = \sum_{i_1=1}^{r_1} \cdots \sum_{i_{d-1}=1}^{r_{d-1}} b_v^{(1)}[1, :, i_1] \otimes b_v^{(2)}[i_1, :, i_2] \otimes \cdots \otimes b_v^{(d)}[i_{d-1}, :, 1]$$

$\boldsymbol{r}$ is the TT-rank of v with a constrain $r_0 = r_d = 1$ to keep the elements of v scalars. The TT format is stable in the sense that for any prescribed $\boldsymbol{r}$ an error minimising $v_{\boldsymbol{r}}$ can always be constructed by a series of SVDs on consecutive matricisations of v.

A.3.1. Element-wise multiplication

Let another TT format tensor with rank $\boldsymbol{s}$ be defined as

$$\mathsf{w}_{\boldsymbol{s}} = \sum_{i_1=1}^{s_1} \cdots \sum_{i_{d-1}=1}^{s_{d-1}} \mathsf{b}_{\mathsf{w}}^{(1)}[1,:,i_1] \otimes \mathsf{b}_{\mathsf{w}}^{(2)}[i_1,:,i_2] \otimes \cdots \otimes \mathsf{b}_{\mathsf{w}}^{(d)}[i_{d-1},:,1]$$

with $\mathsf{b}_{\mathsf{w}}^{(j)} \in \mathbb{K}^{s_{j-1} \times N_j \times s_j}$. The element-wise product of $\mathsf{v}_{\boldsymbol{r}}$ and $\mathsf{w}_{\boldsymbol{s}}$ can also be expressed in TT format:

$$\mathsf{v}_{\boldsymbol{r}} \odot \mathsf{w}_{\boldsymbol{s}} = \sum_{i_1=1}^{t_1} \cdots \sum_{i_{d-1}=1}^{t_{d-1}} b^{(1)}[1,:,i_1] \otimes b^{(2)}[i_1,:,i_2] \otimes \cdots \otimes b^{(d)}[i_{d-1},:,1]$$

where $\boldsymbol{t} = \boldsymbol{r} \odot \boldsymbol{s}$ and $b^{(j)} = \mathsf{b}_{\mathsf{v}}^{(j)} * \mathsf{b}_{\mathsf{w}}^{(j)}$. Here the $*$ denotes one type of Khatri–Rao product [59] which makes Kronecker product only in the first and third dimensions, i.e. it yields an order 3 tensor $b^{(j)} \in \mathbb{K}^{r_{j-1}s_{j-1} \times N_j \times r_j s_j}$. The complexity of the element-wise product is of order $\mathcal{O}(dNr^2s^2)$ with N, r and s as defined in Appendix A.2.

A.3.2. Fourier transform

The Fourier transform of $\mathsf{v}_{\boldsymbol{r}}$ can also be carried out by doing 1-D transforms on each *carriage*:

$$\mathcal{F}_N(\mathsf{v}_{\boldsymbol{r}}) = \sum_{i_1=1}^{r_1} \cdots \sum_{i_{d-1}=1}^{r_{d-1}} \mathcal{F}_{N_1}(\mathsf{b}_{\mathsf{v}}^{(1)}[1,:,i_1]) \otimes \mathcal{F}_{N_2}(\mathsf{b}_{\mathsf{v}}^{(2)}[i_1,:,i_2]) \otimes \cdots \otimes \mathcal{F}_{N_d}(\mathsf{b}_{\mathsf{v}}^{(d)}[i_{d-1},:,1])$$

in which the $\mathcal{F}_{N_j}(\cdot)$ is made on the fibres along the second mode. If FFT is applied here, the number of operations is of order $\mathcal{O}(dr^2N \log N)$.

A.3.3. Rank truncation

The tensor train representation (10) can be obtained either by transforming a full tensor into tensor train format by using $d-1$ sequential SVDs applied on auxiliary matrices of the tensor (known as TT-SVD) [39], or as a result of operations (e.g. additions or multiplications) over tensor train operands. In the first case, an error minimising rank truncation could be directly carried out in the TT-SVD process. The truncation has an error bound $(\sum_{k=1}^{d-1} \epsilon_k^2)^{1/2}$, where ϵ_k is the Frobenius norm error introduced by the truncation of the kth SVD. In the second case, a re-orthogonalisation has to be done in the first place, this is followed by $d-1$ sequential SVDs on unfolded *carriages*. This process is known as TT-truncation (also called rounding).

For the first case, the complexity of truncation is the same as that for the TT-SVD, which is of order $\mathcal{O}(N^{d+1})$. A cheaper alternative for TT-SVD is TT-cross approximation as introduced in [38]. The complexity of TT-truncation in the second case is of order $\mathcal{O}(dNr^3)$.

References

[1] H. Moulinec, P. Suquet, A fast numerical method for computing the linear and nonlinear mechanical properties of composites, C. R. Acad. Sci., Paris II 318 (11) (1994) 1417–1423.

[2] J. Vondřejc, J. Zeman, I. Marek, An FFT-based Galerkin method for homogenization of periodic media, Comput. Math. Appl. 68 (3) (2014) 156–173, http://dx.doi.org/10.1016/j.camwa.2014.05.014.

[3] G.W. Milton, The Theory of Composites, in: Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press, Cambridge, UK, 2002.

[4] J. Zeman, T.W.J. de Geus, J. Vondřejc, R.H.J. Peerlings, M.G.D. Geers, A finite element perspective on non-linear FFT-based micromechanical simulations, Internat. J. Numer. Methods Engrg. 111 (10) (2017) 903–926, http://dx.doi.org/10.1002/nme.5481.

[5] T.W.J. de Geus, J. Vondřejc, J. Zeman, R.H.J. Peerlings, M.G.D. Geers, Finite strain FFT-based non-linear solvers made simple, Comput. Methods Appl. Mech. Engrg. 318 (2017) 412–430, http://dx.doi.org/10.1016/j.cma.2016.12.032.

[6] J. Vondřejc, J. Zeman, I. Marek, Guaranteed upper-lower bounds on homogenized properties by FFT-based Galerkin method, Comput. Methods Appl. Mech. Engrg. 297 (2015) 258–291, http://dx.doi.org/10.1016/j.cma.2015.09.003.

[7] J. Zeman, J. Vondřejc, J. Novák, I. Marek, Accelerating a FFT-based solver for numerical homogenization of periodic media by conjugate gradients, J. Comput. Phys. 229 (21) (2010) 8065–8071, http://dx.doi.org/10.1016/j.jcp.2010.07.010.

[8] J. Vondřejc, Improved guaranteed computable bounds on homogenized properties of periodic media by the Fourier–Galerkin method with exact integration, Internat. J. Numer. Methods Engrg. 107 (13) (2016) 1106–1135, http://dx.doi.org/10.1002/nme.5199.

[9] M. Schneider, Convergence of FFT-based homogenization for strongly heterogeneous media, Math. Methods Appl. Sci. 38 (13) (2014) 2761–2778, http://dx.doi.org/10.1002/mma.3259.

[10] S. Brisard, L. Dormieux, Combining Galerkin approximation techniques with the principle of Hashin and Shtrikman to derive a new FFT-based numerical method for the homogenization of composites, Comput. Methods Appl. Mech. Eng. 217–220 (2012) 197–212, http://dx.doi.org/10.1016/j.cma.2012.01.003.
[11] S. Brisard, L. Dormieux, FFT-based methods for the mechanics of composites: A general variational framework, Comput. Mater. Sci. 49 (3) (2010) 663–671, http://dx.doi.org/10.1016/j.commatsci.2010.06.009.
[12] M. Schneider, D. Merkert, M. Kabel, FFT-based homogenization for microstructures discretized by linear hexahedral elements, Internat. J. Numer. Methods Engrg. 109 (10) (2017) 1461–1489, http://dx.doi.org/10.1002/nme.5336.
[13] F. Willot, Fourier-based schemes for computing the mechanical response of composites with accurate local fields, C. R. Méc. 343 (2015) 232–245, http://dx.doi.org/10.1016/j.crme.2014.12.005.
[14] F. Willot, B. Abdallah, Y.-P. Pellegrini, Fourier-based schemes with modified Green operator for computing the electrical response of heterogeneous media with accurate local fields, Internat. J. Numer. Methods Engrg. 98 (7) (2014) 518–533, http://dx.doi.org/10.1002/nme.4641.
[15] D.J. Eyre, G.W. Milton, A fast numerical scheme for computing the response of composites using grid refinement, Eur. Phys. J. Appl. Phys. 6 (1) (1999) 41–47.
[16] H. Moulinec, P. Suquet, G.W. Milton, Convergence of iterative methods based on Neumann series for composite materials: Theory and practice, Internat. J. Numer. Methods Engrg. 114 (10) (2018) 1103–1130, http://dx.doi.org/10.1002/nme.5777.
[17] N. Mishra, J. Vondřejc, J. Zeman, A comparative study on low-memory iterative solvers for FFT-based homogenization of periodic media, J. Comput. Phys. 321 (2016) 151–168, http://dx.doi.org/10.1016/j.jcp.2016.05.041.
[18] M. Kabel, T. Böhlke, M. Schneider, Efficient fixed point and Newton–Krylov solvers for FFT-based homogenization of elasticity at large deformations, Comput. Mech. 54 (6) (2014) 1497–1514, http://dx.doi.org/10.1007/s00466-014-1071-8.
[19] M. Schneider, An FFT-based fast gradient method for elastic and inelastic unit cell homogenization problems, Comput. Methods Appl. Mech. Engrg. 315 (2017) 846–866, http://dx.doi.org/10.1016/j.cma.2016.11.004.
[20] J. Kochmann, B. Svendsen, S. Reese, L. Ehle, S. Wulfinghoff, J. Mayer, Efficient and accurate two-scale FE-FFT-based prediction of the effective material behavior of elasto-viscoplastic polycrystals, Comput. Mech. 61 (6) (2017) 751–764, http://dx.doi.org/10.1007/s00466-017-1476-2.
[21] F.S. Göküzüm, M.A. Keip, An algorithmically consistent macroscopic tangent operator for FFT-based computational homogenization, Internat. J. Numer. Methods Engrg. 113 (4) (2018) 581–600, http://dx.doi.org/10.1002/nme.5627.
[22] F. Dietrich, D. Merkert, B. Simeon, Derivation of higher-order terms in FFT-based numerical homogenization, in: Lecture Notes in Computational Science and Engineering, vol. 126, 2019, pp. 289–297, http://dx.doi.org/10.1007/978-3-319-96415-7_25.
[23] N. Bertin, L. Capolungo, A FFT-based formulation for discrete dislocation dynamics in heterogeneous media, J. Comput. Phys. 355 (2018) 366–384, http://dx.doi.org/10.1016/J.JCP.2017.11.020.
[24] T.W.J. de Geus, R.H.J. Peerlings, M.G.D. Geers, Competing damage mechanisms in a two-phase microstructure: How microstructure and loading conditions determine the onset of fracture, Int. J. Solids Struct. 97 (2016) 687–698, http://dx.doi.org/10.1016/j.ijsolstr.2016.03.029.
[25] M. Boeff, F. Gutknecht, P.S. Engels, A. Ma, A. Hartmaier, Formulation of nonlocal damage models based on spectral methods for application to complex microstructures, Eng. Fract. Mech. 147 (2015) 373–387, http://dx.doi.org/10.1016/j.engfracmech.2015.06.030.
[26] J. Segurado, R.A. Lebensohn, J. Llorca, Computational homogenization of polycrystals, Adv. Appl. Mech. 51 (2018) 1–114, http://dx.doi.org/10.1016/bs.aams.2018.07.001.
[27] C. Garcia-Cardona, R. Lebensohn, M. Anghel, Parameter estimation in a thermoelastic composite problem via adjoint formulation and model reduction, Internat. J. Numer. Methods Engrg. 112 (6) (2017) 578–600, http://dx.doi.org/10.1002/nme.5530.
[28] S.A. Goreinov, E.E. Tyrtyshnikov, N.L. Zamarashkin, A theory of pseudoskeleton approximations, Linear Algebra Appl. 261 (1–3) (1997) 1–21, http://dx.doi.org/10.1016/S0024-3795(96)00301-1.
[29] M. Bebendorf, Approximation of boundary element matrices, Numer. Math. 86 (4) (2000) 565–589, http://dx.doi.org/10.1007/PL00005410.
[30] W. Hackbusch, Numerical tensor calculus, Acta Numer. 23 (2014) 651–742, http://dx.doi.org/10.1017/S0962492914000087.
[31] T.G. Kolda, B.W. Bader, Tensor decompositions and applications, SIAM Rev. 51 (3) (2009) 455–500, http://dx.doi.org/10.1137/07070111X.
[32] L. Giraldi, A. Nouy, G. Legrain, P. Cartraud, Tensor-based methods for numerical homogenization from high-resolution images, Comput. Methods Appl. Mech. Engrg. 254 (2013) 154–169, http://dx.doi.org/10.1016/j.cma.2012.10.012.
[33] D. Kressner, C. Tobler, Low-rank tensor Krylov subspace methods for parametrized linear systems, SIAM J. Matrix Anal. Appl. 32 (4) (2011) 1288–1316, http://dx.doi.org/10.1137/100799010.
[34] C. Tobler, Low-Rank Tensor Methods for Linear Systems and Eigenvalue Problems (Ph.D. thesis), ETH Zürich, 2012.
[35] S.V. Dolgov, TT-GMRES: Solution to a linear system in the structured tensor format, Russ. J. Numer. Anal. Math. Modelling 28 (2) (2013) 149–172, http://dx.doi.org/10.1515/rnam-2013-0009.
[36] J. Ballani, L. Grasedyck, A projection method to solve linear systems in tensor format, Numer. Linear Algebra Appl. 20 (1) (2013) 27–43, http://dx.doi.org/10.1002/nla.1818.
[37] H.G. Matthies, E. Zander, Solving stochastic systems with low-rank tensor compression, Linear Algebra Appl. 436 (10) (2012) 3819–3838, http://dx.doi.org/10.1016/j.laa.2011.04.017.
[38] I. Oseledets, E. Tyrtyshnikov, TT-cross approximation for multidimensional arrays, Linear Algebra Appl. 432 (1) (2010) 70–88, http://dx.doi.org/10.1016/J.LAA.2009.07.024.
[39] I.V. Oseledets, Tensor-train decomposition, SIAM J. Sci. Comput. 33 (5) (2011) 2295–2317, http://dx.doi.org/10.1137/090752286.
[40] D. Bigoni, A.P. Engsig-Karup, Y.M. Marzouk, Spectral tensor-train decomposition, SIAM J. Sci. Comput. 38 (4) (2016) A2405–A2439, http://dx.doi.org/10.1137/15M1036919.

[41] M. Espig, W. Hackbusch, A. Litvinenko, H.G. Matthies, P. Wähnert, Efficient low-rank approximation of the stochastic Galerkin matrix in tensor formats, Comput. Math. Appl. 67 (4) (2014) 818–829, http://dx.doi.org/10.1016/j.camwa.2012.10.008.
[42] A. Nouy, Low-rank methods for high-dimensional approximation and model order reduction, in: Model Reduction and Approximation: Theory and Algorithms, Society for Industrial and Applied Mathematics, 2015, pp. 1–73, http://dx.doi.org/10.1007/978-3-319-11259-6_21-1.
[43] B.N. Khoromskij, C. Schwab, Tensor-structured Galerkin approximation of parametric and stochastic elliptic PDEs, SIAM J. Sci. Comput. 33 (1) (2011) 364–385, http://dx.doi.org/10.1137/100785715.
[44] B.N. Khoromskij, S.I. Repin, A fast iteration method for solving elliptic problems with quasiperiodic coefficients, Russ. J. Numer. Anal. Math. Modelling 30 (6) (2015) 329–344, http://dx.doi.org/10.1515/rnam-2015-0030.
[45] B. Khoromskij, S. Repin, Rank structured approximation method for quasi-periodic elliptic problems, Comput. Methods Appl. Math. 17 (3) (2017) 457–477, http://dx.doi.org/10.1515/cmam-2017-0014.
[46] A. Nouy, Low-rank tensor methods for model order reduction, in: Handbook of Uncertainty Quantification, Springer International Publishing, Cham, 2015, pp. 1–26, http://dx.doi.org/10.1007/978-3-319-11259-6_21-1.
[47] J. Kochmann, K. Manjunatha, C. Gierden, S. Wulfinghoff, B. Svendsen, S. Reese, A simple and flexible model order reduction method for FFT-based homogenization problems using a sparse sampling technique, Comput. Methods Appl. Mech. Engrg. 347 (2019) 622–638, http://dx.doi.org/10.1016/j.cma.2018.11.032.
[48] A. Bensoussan, J.-L. Lions, G. Papanicolaou, Asymptotic Analysis for Periodic Structures, North Holland, Amsterdam, 1978.
[49] J. Saranen, G. Vainikko, Periodic Integral and Pseudodifferential Equations with Numerical Approximation, in: Springer Monographs Mathematics, Berlin, Heidelberg, 2002.
[50] H. Moulinec, P. Suquet, A numerical method for computing the overall response of nonlinear composites with complex microstructure, Comput. Methods Appl. Mech. Engrg. 157 (1–2) (1998) 69–94, http://dx.doi.org/10.1016/S0045-7825(97)00218-1.
[51] J. Vondřejc, Double-grid quadrature with interpolation-projection (DoGIP) as a novel discretisation approach: An application to FEM on simplexes, Comput. Math. Appl. 78 (11) (2019) 3501–3513, http://dx.doi.org/10.1016/j.camwa.2019.05.021.
[52] M. Ladecký, I. Pultarová, J. Vondřejc, J. Zeman, Preconditioning the spectral fourier method for homogenization of periodic media, 2019, arXiv:https://mat.fsv.cvut.cz/nales/preprints/preprinty/2019/preprinty2019.html.
[53] W. Hackbusch, Tensor Spaces and Numerical Tensor Calculus, Springer Science & Business Media, Berlin, Heidelberg, 2012, http://dx.doi.org/10.1007/978-3-540-78862-1.
[54] Y. Saad, Iterative Methods for Sparse Linear Systems, second ed., SIAM, Philadelphia, PA, USA, 2003.
[55] R.J. Adler, J.E. Taylor, Random Fields and Geometry, Springer, New York, 2009.
[56] B. Matérn, Spatial Variation 36 of Lecture Notes in Statistics, Springer New York, New York, NY, 1986, http://dx.doi.org/10.1007/978-1-4615-7892-5.
[57] J. Vondřejc, T.W.J. de Geus, Energy-based comparison between the Fourier–Galerkin method and the finite element method, J. Comput. Appl. Math. 374 (2020) 112585, http://dx.doi.org/10.1016/j.cam.2019.112585.
[58] L. De Lathauwer, B. De Moor, J. Vandewalle, A multilinear singular value decomposition, SIAM J. Matrix Anal. Appl. 21 (4) (2000) 1253–1278, http://dx.doi.org/10.1137/S0895479896305696.
[59] C.G. Khatri, C.R. Rao, Solutions to some functional equations and their applications to characterization of probability distributions, Sankhya A 30 (2) (1968) 167–180, http://dx.doi.org/10.2307/25049527.