Computing (2009) 84:49–67
DOI 10.1007/s00607-008-0018-3

Application of hierarchical matrices for computing the Karhunen–Loève expansion

B. N. Khoromskij · A. Litvinenko · H. G. Matthies

Received: 26 August 2008 / Accepted: 30 September 2008 / Published online: 31 October 2008

Abstract Realistic mathematical models of physical processes contain uncertainties. These models are often described by stochastic differential equations (SDEs) or stochastic partial differential equations (SPDEs) with multiplicative noise. The uncertainties in the right-hand side or the coefficients are represented as random fields. To solve a given SPDE numerically one has to discretise the deterministic operator as well as the stochastic fields. The total dimension of the SPDE is the product of the dimensions of the deterministic part and the stochastic part. To approximate random fields with as few random variables as possible, but still retaining the essential information, the Karhunen–Loève expansion (KLE) becomes important. The KLE of a random field requires the solution of a large eigenvalue problem. Usually it is solved by a Krylov subspace method with a sparse matrix approximation. We demonstrate the use of sparse hierarchical matrix techniques for this. A log-linear computational cost of the matrix-vector product and a log-linear storage requirement yield an efficient and fast discretisation of the random fields presented.

Keywords Hierarchical matrix · Data-sparse approximation · Karhunen–Loève expansion · Uncertainty quantification · Random fields · Eigenvalue computation

Mathematics Subject Classification (2000) 60H15 · 60H35 · 65N25

B. N. Khoromskij
Max-Planck-Institut für Mathematik in den Naturwissenschaften,
Leipzig, Germany

A. Litvinenko (✉) · H. G. Matthies
Institut für Wissenschaftliches Rechnen,
Hans-Sommer Str. 65, 38106 Brunswick, Germany
e-mail: a.litvinenko@tu-braunschweig.de

1 Introduction

During the last few years there is a great interest in numerical methods for solving stochastic PDEs and ODEs [2,3,10,25,35,37–39]. Examples are stochastic Navier–Stokes equations, stochastic plasticity equations and stochastic aerodynamic equations. Very often these equations contain parameters, right-hand sides, initial or boundary conditions which have a stochastic nature. Typical examples are conductivity coefficients in groundwater flow problems, plasticity of the material and parameters in turbulence modelling. To solve the problem, the given stochastic differential or integral equation has to be discretised. For the discretisation of the deterministic part one can use any known technique (finite element, finite differences or finite volumes). For the discretisation of random fields the Karhunen–Loève expansion (KLE) [28] is usually used. Another important application of the KLE is the direct computation of higher order moments of the solution without computing the solution per se [29,36].

Each random field is characterised by its covariance function. To discretise this random field one has to solve an eigenproblem for a Fredholm integral operator with the covariance function as the kernel. In a straight-forward discretisations, the matrix is dense and hence the computational cost is $\mathcal{O}(n^3)$ FLOPS, where n is the number of degrees of freedom (dof) in the computational domain. For special cases, when the covariance function is stationary (i.e. $\mathrm{cov}(x, y) = \mathrm{cov}(x - y)$) and the computational domain is an axiparallel rectangle with uniform and axiparallel triangulation the Fast Fourier technique [12] can be applied with the computational cost $\mathcal{O}(n \log n)$. In [23] the authors introduced the so-called Hierarchical Kronecker Tensor (HKT) format for sparse approximation of integral operators. The matrix-vector product in the HKT format can be done in $\mathcal{O}(dn^{1/d} \log n)$ FLOPS, where d is the dimension of the domain. For more general cases of the covariance matrix, for a non-rectangular domain or for a non-axiparallel triangulation, the FFT is not applicable and a data sparse technique should be applied (e.g., the $\mathcal{H}$-matrix technique [17,18,20,21]).

In [37], the authors compute the KLE by the Fast Multipole method with an iterative Krylov eigensolver. In [10] a brief overview of how boundary value problems with random data may be solved using the stochastic FEM is described. In the same paper the authors apply $\mathcal{H}$-matrices and the Lanczos-based thick-restart method [42] for computing the KLE of random fields.

In the current paper, we consider the application of the $\mathcal{H}$-matrix method in a systematic way. The rest of this paper is structured as follows. In Sect. 2, we set up the problem and recall the Karhunen–Loève expansion. The $\mathcal{H}$-matrix technique is presented in Sect. 3. In particular, we prove the asymptotic smoothness of the arising covariance functions. The $\mathcal{H}$-matrix approximation of covariance functions is shown in Sect. 4. Finally, in Sect. 5, we provide numerical results for solving an eigenproblem with predefined $\mathcal{H}$-matrix-vector product.

2 Background

Nowadays the trend of numerical mathematics is often trying to resolve inexact mathematical models by very exact deterministic numerical methods. The reason is that

almost each mathematical model contains uncertainties in the coefficients, right-hand side, boundary conditions, initial data as well as in the geometry. Such type of uncertainties can be modelled by random fields. In [2,3,25,29,35,37,39], the authors consider the following stochastic elliptic boundary value problem

$$\begin{aligned} -\operatorname{div}(\kappa(x,\omega)\nabla u) &= f(x,\omega) \quad \text{in } \mathcal{G}\times\Omega, \quad \mathcal{G}\subset\mathbb{R}^d, \\ u &= g(x,\omega) \quad \text{on } \partial\mathcal{G}\times\Omega, \end{aligned} \tag{1}$$

where the conductivity coefficient $\kappa(x,\omega)$, the right-hand side $f(x,\omega)$, the boundary data $g(x,\omega)$ and the solution $u(x,\omega)$ are random fields. The computational domain $\mathcal{G}$ is a bounded domain, $x\in\mathcal{G}$ and ω belongs to the space of random events Ω.

To guarantee the positive definiteness and regularity of the operator in (1) it is assumed that

$$0<\kappa_{\min}\le\kappa(x,\omega)\le\kappa_{\max}<\infty, \quad \text{a.e. on } \mathcal{G}\times\Omega.$$

We assume that there is a triplet $(\Omega,\Sigma,\mathbb{P})$, where Ω is a set of random elementary events, Σ is the σ-algebra of Borel subsets of Ω and $\mathbb{P}$ a probability measure. We assume also that the random fields $\kappa(\cdot,\omega):\Omega\to L^\infty(\mathcal{G})$, $f(\cdot,\omega):\Omega\to L^2(\mathcal{G})$ and $g(\cdot,\omega):\Omega\to L^2(\partial\mathcal{G})$ have finite variance.

Let us as an example consider the random field $\kappa(x,\omega)$. The mean value $\overline{\kappa}(x,\omega)$ and the covariance function $\operatorname{cov}_\kappa(x,y)$, $x,y\in\mathbb{R}^d$, should be provided. By definition, the covariance function is symmetric and positive semi-definite. One can classify all covariance functions into the three following groups:

1. Directionally independent (isotropic) and translation invariant (stationary or homogeneous), i.e. $\operatorname{cov}(x,y)=\operatorname{cov}(|x-y|)$.
2. Directionally dependent (anisotropic) and stationary or homogeneous, i.e. $\operatorname{cov}(x,y)=\operatorname{cov}(x-y)$.
3. Instationary and non-homogeneous, i.e. of a general type.

The covariance functions of types (1) and (2), discretised on an axiparallel rectangular grid, result in (block) Toeplitz matrices. These matrices can be further extended to (block) circulant ones. The matrix vector multiplication in the class of (block) circulant matrices can be performed by the Fast Fourier Transformation (FFT) very efficiently. In the case of a general grid as well as in the third case, the discretised covariance matrix is not a Toeplitz one and the FT cannot be applied. Thus, we need a general data sparse format to store covariance matrices.

For the numerical solution of (1), the presented random fields need to be discretised both in the stochastic and in the spatial dimension. One of the main tools here is the Karhunen–Loève expansion (KLE) [28]. Thus, an effective and "sparse" computation of the KLE is a key point in solving Eq. (1) [31]. Let us define the following operator T which will be needed for computing the KLE of $\kappa(x,\omega)$:

$$T:L^2(\mathcal{G})\to L^2(\mathcal{G}), \quad (T\phi)(x):=\int_{\mathcal{G}}\operatorname{cov}_\kappa(x,y)\phi(y)\mathrm{d}y.$$

For $\mathrm{cov}_\kappa \in L^2(\mathcal{G}\times\mathcal{G})$, the operator T is compact and selfadjoint [40], in fact Hilbert–Schmidt. As the covariance function cov_κ is symmetric positive semi-definite, hence so is T. Thus, the eigenfunctions ϕ_ℓ of the following Fredholm integral equation of the second kind

$$T\phi_\ell = \lambda_\ell \phi_\ell, \quad \phi_\ell \in L^2(\mathcal{G}),\ \ell \in \mathbb{N}, \tag{2}$$

are mutually orthogonal and define a basis of $L^2(\mathcal{G})$ (for more details see [19,33]). The eigenvalues λ_ℓ are real, non-negative and can be arranged decreasingly $\lambda_1 \geq \lambda_2 \geq \cdots \geq 0$ [33]. From Mercer's theorem [33,40], it follows that for a continuous cov_κ the eigenfunctions are continuous and the convergence of

$$\mathrm{cov}_{\kappa_m}(x,y) = \sum_{\ell=0}^{m-1} \lambda_\ell \phi_\ell(x)\phi_\ell(y)$$

as $m \to \infty$ to the exact covariance function cov_κ is absolute and uniform on $\mathcal{G}\times\mathcal{G}$ [33]. The convergence rates can be estimated through the smoothness of the covariance function [37].

By definition, the KLE of $\kappa(x,\omega)$ is the following series

$$\begin{aligned} \kappa(x,\omega) &= \mu_\kappa(x) + \sum_{\ell=1}^{\infty} \sqrt{\lambda_\ell}\phi_\ell(x)\xi_\ell(\omega), \quad \text{where} \\ \mu_\kappa(x) &= E_\kappa(x), \quad \xi_\ell(\omega) = \frac{1}{\sqrt{\lambda_\ell}} \int_{\mathcal{G}} (\kappa(x,\omega) - \mu_\kappa(x))\phi_\ell(x)\mathrm{d}x, \end{aligned} \tag{3}$$

$E_\kappa(x)$ is the mean value of $\kappa(x,\omega)$, λ_ℓ and ϕ_ℓ are the eigenvalues and the eigenvectors of problem (2) and $\xi_\ell(\omega)$ uncorrelated random variables. For numerical purposes one truncates the KLE (3) to a finite number m of terms. In the case of a Gaussian random field, the ξ_ℓ are independent standard normal random variables. In the case of a non-Gaussian random field, the ξ_ℓ are uncorrelated but not independent, and can be approximated in a set of new independent Gaussian random variables [24,41], e.g.

$$\xi_\ell(\omega) = \sum_{\alpha\in\mathcal{J}} \kappa^{(\alpha)} H_\alpha(\boldsymbol{\theta}(\omega)),$$

where $\boldsymbol{\theta}(\omega) = (\theta_1(\omega), \theta_2(\omega), \ldots)$, $\kappa^{(\alpha)}$ are coefficients, H_α, $\alpha \in \mathcal{J}$, is a Hermitian basis and $\mathcal{J} := \{\alpha | \alpha = (\alpha_1, \ldots, \alpha_j, \ldots),\ \alpha_j \in \mathbb{N}_0\}$ a multi-index set. For the purpose of actual computation, truncate the polynomial chaos expansion (PCE) [24,41] after finitely many terms, e.g.

$$\alpha \in \mathcal{J}_{M,p} := \{\alpha \in \mathcal{J} \mid \gamma(\alpha) \leq M, |\alpha| \leq p\}, \gamma(\alpha) := \max\{_J \in \mathbb{N} \mid \alpha_J > 0\}.$$

In [33] it is shown that the m-term KL truncation is best in Hilbert–Schmidt norm.

As soon as the m-term KLE of the conductivity $\kappa(x,\omega)$ is computed and the random variables $\xi_\ell(\omega) \in \mathbb{R}^{|J|}$ are discretised [2,11,25], one can obtain, after applying the stochastic Galerkin approximation method [30] and truncated PCE, the following equation

$$\mathbf{Ku} = \left[\sum_{\ell=0}^{m-1} \sum_{\gamma \in J_{M,p}} \boldsymbol{\Delta}^{(\gamma)} \otimes \boldsymbol{K}_\ell\right] \mathbf{u} = \mathbf{f}, \tag{4}$$

where $\boldsymbol{\Delta}^{(\gamma)}$ are some discrete operators which come from the Hermitian algebra and can be computed analytically [25,29,30]. The sparsity pattern of $\boldsymbol{\Delta}^{(\gamma)}$ depends on how many terms were used in the PCE. Note that the matrices $\boldsymbol{K}_\ell \in \mathbb{R}^{n\times n}$ allow for data sparse approximations, in particular the hierarchical ($\mathcal{H}$) matrix approximation. Note that the iterative solvers, used for the solution of (4), do not require that the matrices $\boldsymbol{K}_\ell$ are stored explicitly.

Now one can see that the accuracy of the discretisation of (1) depends on the convergence rate of $\kappa_m(x,\omega)$ with respect to $m \to \infty$. Thus, cheap and accurate computing of the KLE approximation of the given random fields is required. Further, in this paper, we combine the $\mathcal{H}$-matrix data representation together with Krylov solvers for the efficient computation of the m-term KLEs of the given random fields and the solution.

2.1 FE discretisation of Eq. (2)

Further in the paper, we will use the bold font for defining discretised objects, e.g. $\boldsymbol{u} \in \mathbb{R}^n$ or $\boldsymbol{C} \in \mathbb{R}^{n\times n}$.

In general, the eigenvalue problem (2) needs to be solved numerically and standard techniques [1,19,32] may be used for this. We consider the following Galerkin discretisation of the operator in (2). Let $I = \{1,\dots,n\}$. Assume that b_1,…,b_n are the nodal basis functions with respect to the nodes $x_1,\dots,x_n \in \mathcal{G} \subset \mathbb{R}^d$, i.e. $b_i(x_j) = \delta_{ij}$, $i,j \in I$. Let $V_h = \operatorname{span}\{b_1,\dots,b_n\}$ and for the stochastic variables we introduce $\boldsymbol{\zeta} = (\zeta_1,\dots,\zeta_n)^T$, $\zeta_i(\omega) := \kappa(x_i,\omega)$, $i \in I$.

The interpolation of $\kappa(x,\omega)$ in the FE basis above is then

$$\kappa^h(x,\omega) = \sum_{i=1}^{n} b_i(x)\zeta_i(\omega) = \boldsymbol{b}(x)\boldsymbol{\zeta}(\omega), \quad \boldsymbol{b}(x) = (b_1(x),\dots,b_n(x)).$$

The covariance function of κ^h is

$$\operatorname{cov}_{\kappa^h}(x,y) = \sum_{i=1}^{n}\sum_{j=1}^{n} b_i(x)\boldsymbol{C}_{ij}b_j(y) = \boldsymbol{b}(x)\boldsymbol{C}\boldsymbol{b}(y)^T, \quad \text{with } \boldsymbol{C}_{ij} = \operatorname{cov}_\kappa(x_i,y_j). \tag{5}$$

Note that this discretisation may use a different grid than the discretisation of the spatial part in (1). Applying (5) and

$$\phi_\ell(y) = \sum_{j=1}^{n} b_j(y)\phi_{j\ell} = \boldsymbol{b}(y)\boldsymbol{\phi}_\ell, \quad \boldsymbol{\phi}_\ell := (\phi_{1\ell}, \ldots, \phi_{n\ell})^T$$

to the eigenvalue problem

$$\int_{\mathcal{G}} \mathrm{cov}_\kappa(x, y)\phi_\ell(y)\mathrm{d}y = \lambda_\ell \phi_\ell(x), \tag{6}$$

we obtain

$$\int_{\mathcal{G}} \boldsymbol{b}(x)\boldsymbol{C}\boldsymbol{b}(y)^T \boldsymbol{b}(y)\boldsymbol{\phi}_\ell \mathrm{d}y = \lambda_\ell \boldsymbol{b}(x)\boldsymbol{\phi}_\ell.$$

The weak formulation (Galerkin weighting) gives

$$\int_{\mathcal{G}}\int_{\mathcal{G}} \boldsymbol{b}(x)^T \boldsymbol{b}(x)\boldsymbol{C}\boldsymbol{b}(y)^T \boldsymbol{b}(y)\boldsymbol{\phi}_\ell \mathrm{d}y\mathrm{d}x = \int_{\mathcal{G}} \lambda_\ell \boldsymbol{b}(x)^T \boldsymbol{b}(x)\boldsymbol{\phi}_\ell \mathrm{d}x,$$

or

$$\boldsymbol{W}\boldsymbol{\phi}_\ell = \lambda_\ell \boldsymbol{M}\boldsymbol{\phi}_\ell,$$

where the matrix $\boldsymbol{W}$ and mass matrix $\boldsymbol{M}$ are defined as follows

$$\boldsymbol{W}_{ij} := \sum_{k,\nu} \int_{\mathcal{G}}\int_{\mathcal{G}} b_i(x)b_k(x)\boldsymbol{C}_{k\nu}b_j(y)b_\nu(y)\mathrm{d}x\mathrm{d}y, \quad \mathcal{G} \subset \mathbb{R}^d, \quad k, \nu \in I,$$

$$\boldsymbol{M}_{ij} = \int_{\mathcal{G}} b_i(x)b_j(x)\mathrm{d}x, \quad i, j \in I.$$

Recall that the matrix $\boldsymbol{W}$ is symmetric positive semi-definite and dense. The mass matrix $\boldsymbol{M}$ is symmetric positive definite and may be sparse. Now, the discrete eigenvalue problem looks like

$$\boldsymbol{W}\boldsymbol{\phi}_\ell = \lambda_\ell^h \boldsymbol{M}\boldsymbol{\phi}_\ell, \quad \boldsymbol{W} = \boldsymbol{M}\boldsymbol{C}\boldsymbol{M}, \quad \boldsymbol{C}_{ij} = \mathrm{cov}_\kappa(x_i, y_j). \tag{7}$$

Here the matrix $\boldsymbol{M}$ is stored in the usual data sparse format and the matrix $\boldsymbol{C}$ is approximated in the $\mathcal{H}$-matrix format (see Sect. 3). If not the complete spectrum is of interest, but only a part of it then the needed computational resources can be drastically reduced [4]. To compute m eigenvalues ($m \ll n$) and corresponding eigenvectors we apply an iterative Krylov subspace (Lanczos) eigenvalue solver for symmetric matrices [4,26,27,34,42]. This eigensolver requires only matrix-vector multiplications. All matrix-vector multiplications are performed in the $\mathcal{H}$-matrix format which will cost $\mathcal{O}(n \log n)$. Note that to solve the symmetric problem (7) often a third party eigensolver

requires the user to define the matrix-vector products $\boldsymbol{w} = \boldsymbol{M}^{-1}\boldsymbol{W}\boldsymbol{v}$ and $\boldsymbol{w} = \boldsymbol{M}\boldsymbol{v}$. The same problem can be written in the form

$$\boldsymbol{C}\boldsymbol{M}\boldsymbol{\phi}_i = \lambda_i \boldsymbol{\phi}_i, \tag{8}$$

where the product $\boldsymbol{C}\boldsymbol{M}$ is selfadjoint with respect to the new scalar product $(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j)_{\boldsymbol{M}} = (\boldsymbol{M}\boldsymbol{\phi}_i, \boldsymbol{\phi}_j)$.

3 $\mathcal{H}$-Matrix technique

Usually the mass matrix $\boldsymbol{M}$ is stored in a sparse matrix format, which requires linear complexity. The covariance matrix $\boldsymbol{C}$ is not sparse and, in general, requires $\mathcal{O}(n^2)$ units of memory for the storage and $\mathcal{O}(n^2)$ FLOPS for the matrix-vector multiplication. In this section it will be shown how to approximate general covariance matrices with the $\mathcal{H}$-matrix format [17,18,20,22]. The $\mathcal{H}$-matrix technique is nothing but a hierarchical division of a given matrix into subblocks and further approximation of the majority of them by low-rank matrices (Fig. 2). To define which subblocks can be approximated well by low-rank matrices and which not, a so-called admissibility condition is used. When decomposition into subblocks is done an important question is, how to compute the low-rank approximations. For this purpose we offer to use the ACA algorithm [5,7–9,15] which does the job with a linear complexity.

3.1 Admissibility conditions

Originally the $\mathcal{H}$-matrix technique was developed for the approximation of stiffness matrices coming from partial differential and integral equations [9,17,20]. Typical kernels of integral equations are the following Green functions:

$$\chi(x, y) := \frac{1}{|x - y|^{d-2}},\ x, y \in \mathbb{R}^d,\ d \geq 3 \quad \text{or} \quad \chi(x, y) := \log|x - y|,\ x, y \in \mathbb{R}^2, \tag{9}$$

with singularities at $x = y$. The idea behind $\mathcal{H}$-matrices is to approximate blocks far from diagonal (far from the singularity) by low-rank matrices. The admissibility condition (criteria) is used to divide a given matrix into subblocks and define which subblocks can be approximated well by low-rank matrices and which not. Let us explain how to obtain an admissibility condition for the functions in (9).

Let I be an index set of all degrees of freedom. Denote for each index $i \in I$ corresponding to a basis function b_i the support $\mathcal{G}_i := \operatorname{supp} b_i \subset \mathbb{R}^d$. Now we define two trees which are necessary for the definition of hierarchical matrices. These trees are labeled trees where the label of a vertex t is denoted by $\hat{t}$.

Definition 3.1 (Cluster Tree $T_{I\times I}$)[17,20] A finite tree T_I is a cluster tree over the index set I if the following conditions hold:

- I is the root of T_I and a subset $\hat{t} \subseteq I$ holds for all $t \in T_I$.

- If $t \in T_I$ is not a leaf, then the set of sons $\mathrm{sons}(t)$ contains disjoint subsets of I and the subset $\hat{t}$ is the disjoint union of its sons, $\hat{t} = \bigcup_{s\in \mathrm{sons}(t)} \hat{s}$.
- If $t \in T_I$ is a leaf, then $|\hat{t}| \le n_{min}$ for a fixed number n_{min}.

Definition 3.2 (Block Cluster Tree $T_{I\times I}$)[17,20] Let T_I be a cluster tree over the index set I. A finite tree $T_{I\times I}$ is a block cluster tree based on T_I if the following conditions hold:

- $\mathrm{root}(T_{I\times I}) = I \times I$.
- Each vertex b of $T_{I\times I}$ has the form $b = (\tau, \sigma)$ with clusters $\tau, \sigma \in T_I$.
- For each vertex (τ, σ) with $\mathrm{sons}(\tau, \sigma) \neq \varnothing$, we have

$$\mathrm{sons}(\tau,\sigma) = \begin{cases} (\tau, \sigma') : \sigma' \in \mathrm{sons}(\sigma), & \text{if } \mathrm{sons}(\tau) = \varnothing \wedge \mathrm{sons}(\sigma) \neq \varnothing \\ (\tau', \sigma) : \tau' \in \mathrm{sons}(\tau), & \text{if } \mathrm{sons}(\tau) \neq \varnothing \wedge \mathrm{sons}(\sigma) = \varnothing \\ (\tau', \sigma') : \tau' \in \mathrm{sons}(\tau), \sigma' \in \mathrm{sons}(\sigma), & \text{otherwise} \end{cases}$$

- The label of a vertex (τ, σ) is given by $\widehat{(\tau, \sigma)} = \hat{\tau} \times \hat{\sigma} \subseteq I \times I$.

We can see that $\widehat{\mathrm{root}(T_{I\times I})} = I \times I$. This implies that the set of leaves of $T_{I\times I}$ is a partition of $I \times I$.

We generalise $\mathcal{G}_i$ to clusters $\tau \in T_I$ by setting $\mathcal{G}_\tau := \bigcup_{i\in\tau} \mathcal{G}_i$, i.e., $\mathcal{G}_\tau$ is the minimal subset of $\mathbb{R}^d$ that contains the supports of all basis functions b_i with $i \in \tau$.

Suppose that $\mathcal{G}_\tau \subset \mathbb{R}^d$ and $\mathcal{G}_\sigma \subset \mathbb{R}^d$ are compact and $\chi(x, y)$ is defined for $(x, y) \in \mathcal{G}_\tau \times \mathcal{G}_\sigma$ with $x \neq y$. The standard assumption on the kernel function in the $\mathcal{H}$-matrix theory is asymptotic smoothness of $\chi(x, y) \in C^\infty(\mathcal{G}_\tau \times \mathcal{G}_\sigma)$, i.e, that

$$|\partial_x^\alpha \partial_y^\beta \chi(x, y)| \le C_1 |\alpha + \beta|! C_0^{|\alpha+\beta|} \|x - y\|^{-|\alpha+\beta|-\gamma}, \quad \alpha, \beta \in \mathbb{N},$$

holds for constants C_1, C_0 and $\gamma \in \mathbb{R}$. This estimation is used to control the error ϵ_q from the Taylor expansion

$$\chi(x, y) = \sum_{\alpha\in\mathbb{N}_0^d, |\alpha|\le q} (x - x_0)^\alpha \frac{1}{\alpha!} \partial_x^\alpha \chi(x_0, y) + \epsilon_q.$$

Let S be an integral operator with an asymptotically smooth kernel χ in the domain $\mathcal{G}_\tau \times \mathcal{G}_\sigma$:

$$(Sv)(x) = \int_{\mathcal{G}_\sigma} \chi(x, y) v(y) dy, \quad x \in \mathcal{G}_\tau.$$

Suppose that $\chi_k(x, y)$ is an approximation of χ in $\mathcal{G}_\tau \times \mathcal{G}_\sigma$ of the separate form (e.g., Taylor or Lagrange polynomials):

$$\chi_k(x, y) = \sum_{\nu=1}^{k} \varphi_\nu(x) \psi_\nu(y), \tag{10}$$

where k is the rank of separation. We are aiming at an approximation of the form (10) such that exponential convergence

$$\|\chi - \chi_k\|_{\infty,\mathcal{G}_\tau \times \mathcal{G}_\sigma} \leq \mathcal{O}(\eta^k) \tag{11}$$

holds. For this purpose we introduce the following admissibility condition.

Definition 3.3 The standard admissibility condition (Adm_η) for two domains B_τ and B_σ (which actually correspond to two clusters τ and σ) is defined as follows

$$\min\{\mathrm{diam}(B_\tau), \mathrm{diam}(B_\sigma)\} \leq \eta \mathrm{dist}(B_\tau, B_\sigma), \tag{12}$$

where $B_\tau,\ B_\sigma \subset \mathbb{R}^d$ are axis-parallel bounding boxes of the clusters τ and σ such that $\mathcal{G}_\tau \subset B_\tau$ and $\mathcal{G}_\sigma \subset B_\sigma$.

Lemma 3.1 *The function $\chi(x, y) = e^{-|x-y|}$ converges exponentially, i.e. $\exists \eta$ such that for $\chi_k(x, y)$ from (10) holds*

$$\|\chi(x, y) - \chi_k(x, y)\| \leq \mathcal{O}(\eta^k). \tag{13}$$

Proof Let $x, y \in \mathcal{G} := [0, 1]$, $x \in \tau := [a, b]$, $0 \leq a < b \leq 1$, and $y \in \sigma := [c, d]$, $b \leq c < d \leq 1$. After introduction of the new variable $t := x - y$, we obtain $\chi(t) := e^{-t}$ with $t \in [c - b, d - a]$. The Taylor series of $\chi(t)$ in point $t_0 := \frac{(c-b)+(d-a)}{2}$ is

$$\begin{aligned}\chi(t) &= e^{-t_0}\left(1 + \sum_{j=1}^{\infty} \frac{(-1)^j}{j!}(t - t_0)^j\right) \\ &= e^{-t_0}\left(1 + \sum_{j=1}^{k} \frac{(-1)^j}{j!}(t - t_0)^j + \frac{(-1)^{k+1}}{(k+1)!}(\tilde{t} - t_0)^{k+1}\right),\end{aligned}$$

where $\tilde{t} \in [c - b, d - a]$. Let $\varepsilon := e^{-t_0}\frac{(-1)^{k+1}}{(k+1)!}(\tilde{t} - t_0)^{k+1}$, $L_1 := c - b$, $L_2 := d - a$ then

$$\begin{aligned}|\varepsilon| \leq e^{-t_0}\frac{(L_2 - L_1)^{k+1}}{(k+1)!} &\leq e^{-t_0} \cdot \frac{(L_2 - L_1)^{L_2 - L_1}}{(L_2 - L_1)!}\frac{(L_2 - L_1)^{k+1-(L_2-L_1)}}{(L_2 - L_1 + 1) \cdot \cdots \cdot (k+1)} \\ &\leq C \cdot \eta^{k+1-(L_2-L_1)},\end{aligned}$$

where $C := e^{-t_0}\frac{(L_2-L_1)^{L_2-L_1}}{(L_2-L_1)!}$ and $\eta := \frac{L_2-L_1}{L_2-L_1+1} < 1$. □

We will say that a pair (τ, σ) of clusters τ and $\sigma \in T_I$ is admissible if the condition (12) is satisfied. The admissibility condition indicates blocks which allow rank-k approximation and which not (see Fig. 2). The blocks for which condition (12) is true (called admissible blocks) are approximated by rank-k matrices. All other blocks are computed as usual.

In order to get a simpler partitioning (see an example in Fig. 2, right), we define the weaker admissibility condition Adm_W for a pair (τ, σ):

$$\text{Block } b = \tau \times \sigma \in T_{I\times I} \quad \text{is weak admissible} \Leftrightarrow (\text{(b is a leaf) or } \sigma \neq \tau), \tag{14}$$

where τ, σ are assumed to belong to the same level of $T_{I\times I}$.

The covariance functions which are considered in this paper (see Sect. 4) do not have singularities like in (9) and this is why more appropriate admissibility conditions are required. Different types of covariance functions require different admissibility conditions. The development of new admissibility condition is not an easy task and it is out of frame of this paper.

Let us consider properties of functions depending on $(x-y)$, i.e. $\chi(x, y) = s(x-y)$. If $x \in B_x$ and $y \in B_y$ then $r := x - y$ belongs to

$$B_r := \{x - y : x \in B_x,\ y \in B_y\}.$$

Lemma 3.2 [21, Proposition 4.1.2] *Any polynomial $P(x, y)$ can be represented in the form*:

$$P(x, y) = \sum_{\nu=0}^{k_1} p_\nu(x) y^\nu \quad or \quad P(x, y) = \sum_{\mu=0}^{k_2} x^\mu q_\mu(y),$$

where k_1 (k_2) is the polynomial degree in x (y) and p_ν and q_μ are polynomials in one variable.

If the function $f(\cdot)$ is approximated in B_r by a polynomial $P(r)$ (Taylor series, Lagrange polynomial, etc.), i.e. $f(r) \approx P(r)$ then the variables x and y have the same degree $k = k_1 = k_2$ in $P(r)$. Applying the previous lemma, we obtain a separable k-term approximation of $f(x - y)$.

In [21, Sect. E.2] the author explains how to transfer the asymptotical smoothness of the function $f(t)$ to the asymptotical smoothness of the function $F(x, y) := f(|x - y|)$. Let f be defined on $\mathcal{G}_0 \subset \mathbb{R}$ and $\mathcal{G}_0 \supset (-d_f, d_f)$, $d_f > 0$.

Definition 3.4 [21, Sect. E.2] f is asymptotically smooth if

$$\left|\left(\frac{d}{dt}\right)^\nu f(t)\right| \leq C_0 |t|^{-\nu-s} \quad \text{for } t \in \mathcal{G}_0,\ \nu \in \mathbb{N},\ s \in \mathbb{R} \text{ and a constant } C_0 = C_0(\nu). \tag{15}$$

For $t := |x - y|$, $x, y \in \mathbb{R}^d$ we obtain the function

$$F(x, y) := f(|x - y|). \tag{16}$$

Let us denote the directional derivative by $D_\upsilon := \sum_{i=1}^d \upsilon_i \frac{\partial}{\partial x_i}$, where $\upsilon \in \mathbb{R}^d$.

Proposition 3.1 [21, Sect. E.2] *If the function f is asymptotically smooth in sense of* (15)*, then $F(x, y)$ from* (16) *is also asymptotically smooth, i.e. for all directional derivatives D we have*

$$|D^k F(x, y)| \le k! C_0 |x - y|^{-k-s} \quad (0 \ne |x - y| < d_f),\ C_0 > 0.$$

Lemma 3.3 *The function $F(x, y) = F(r) = e^{-|r|}$ is asymptotically smooth.*

Proof Apply Proposition (3.1) to the asymptotical smooth function $f(t) := e^{-t}$. □

Remark 3.1 For most of the covariance functions considered in our applications the asymptotic smoothness can be verified.

3.2 Rank-k adaptive cross approximation

Let $\boldsymbol{R} \in \mathbb{R}^{p\times q}$ and

$$\boldsymbol{R} = \boldsymbol{A}\boldsymbol{B}^T, \quad \text{where } \boldsymbol{A} \in \mathbb{R}^{p\times k}, \quad \boldsymbol{B} \in \mathbb{R}^{q\times k}, \quad k \in \mathbb{N}. \tag{17}$$

Note that any matrix of rank k can be represented in the form (17).

Suppose that b is a block of the matrix $\boldsymbol{W}$ and $\boldsymbol{R} := \boldsymbol{W}|_b$. Suppose it is known that $\boldsymbol{R}$ may be approximated by a rank-k matrix. We explain below how to compute $\boldsymbol{R}$ in the form (17). One possibility is the Adaptive Cross Approximation (ACA) algorithm [5,7–9,15]. ACA is especially effective for assembling low-rank matrices. It requires only k columns and k rows of the matrix under consideration and, thus, has the computational cost $k(p + q)$. In [15], it is proved that if there exists a sufficiently good low-rank approximation, then there also exists a cross approximation with almost the same accuracy in the sense of the 2-norm.

The ACA algorithm computes vectors a_ℓ and b_ℓ which form $\tilde{\boldsymbol{R}} = \sum_{\ell=1}^{k} a_\ell b_\ell^T$ such that $\|\boldsymbol{R} - \tilde{\boldsymbol{R}}\| \le \varepsilon$, where ε is the desired accuracy [7,9]. In [8], the reader can also find different counterexamples when the standard ACA algorithm does not work. Here we present the standard version of the ACA algorithm.

Algorithm 3.1 *ACA algorithm*
begin
/ input is a required accuracy ε and a function to compute $\boldsymbol{R}_{ij}$ */;*
/ output is matrix $\tilde{\boldsymbol{R}}$ */;*
$k = 0$; $\tilde{\boldsymbol{R}} = 0$;
$S = \emptyset$; $T = \emptyset$; / sets of row and column indices */*
do
Take a row $i^* \notin S$;
Subtract $\boldsymbol{R}_{i^*j} := \boldsymbol{R}_{i^*j} - \tilde{\boldsymbol{R}}_{i^*j}$, $j = 1..q$;
Find $\max_j |a_{i^*j}| \ne 0$, $j < q$. Suppose it lies in column j^*;
Compute all elements b_{ij^*} in column j^*, $i < p$;
Subtract $\boldsymbol{R}_{ij^*} := \boldsymbol{R}_{ij^*} - \tilde{\boldsymbol{R}}_{ij^*}$, $i = 1..p$;

$k := k + 1$; $S := S \cup \{i^*\}$; $T := T \cup \{j^*\}$;
Compute $\tilde{\boldsymbol{R}} = \tilde{\boldsymbol{R}} + a_{i^*} \cdot b_{j^*}^T$; /* *it is rank k approximation*/
if$(\|a_{i^*} \cdot b_{j^*}^T\|_2 \le \varepsilon \cdot \|a_1 \cdot b_1^T\|_2)$ *return* $\tilde{\boldsymbol{R}}$;
Find $\max_i |b_{ij^*}|$, $i < p$, $i \neq i^*$. *The row where it lies is a new row* i^*;
until$(k < k_{\max})$
return $\tilde{\boldsymbol{R}}$;
end;

Note that the algorithm does not compute the whole matrix $\boldsymbol{R}$. The subtraction is done only from the elements under consideration, i.e. row a_ℓ and column b_ℓ, $\ell = 1, \ldots, k$.

Remark 3.2 Further optimisation of the ACA algorithm can be done by the truncated SVD. Suppose that a factorisation of matrix $\boldsymbol{R} = \boldsymbol{A}\boldsymbol{B}^T$, $\boldsymbol{A} \in \mathbb{R}^{p \times K}$, $\boldsymbol{B} \in \mathbb{R}^{q \times K}$, is found by ACA. Suppose also that the rank of $\boldsymbol{R}$ is k, $k < K$. Then one can apply the truncated SVD algorithm to compute $\boldsymbol{R} = \boldsymbol{U}\Sigma\boldsymbol{V}^T$ requiring $\mathcal{O}((p+q)K^2 + K^3)$ FLOPS.

3.3 $\mathcal{H}$-Matrices

Definition 3.5 [20] Let I be an index set and $T_{I\times I}$ be a hierarchical division of the index set product $I \times I$ into subblocks (so-called block cluster tree). The set of $\mathcal{H}$-matrices is defined as

$$\mathcal{H}(T_{I\times I}, k) := \{\boldsymbol{W} \in \mathbb{R}^{I\times I} \mid \operatorname{rank}(\boldsymbol{W}|_b) \le k \text{ for all admissible blocks } b \text{ of } T_{I\times I}\}.$$

Here, $\boldsymbol{W}|_b = (w_{ij})_{(i,j)\in b}$ denotes the matrix block of $\boldsymbol{W} = (w_{ij})_{i,j\in I}$ corresponding to $b \in T_{I\times I}$.

We denote an $\mathcal{H}$-matrix approximation of $\boldsymbol{W}$ by $\tilde{\boldsymbol{W}}$.

Finally, we list computational complexities of basic algebraic operations with $\mathcal{H}$-matrices.

Theorem 3.1 [17,20] *Let I be an index set, $n := |I|$, $T_{I\times I}$ a tree which defines the block structure, depth$(T_{I\times I}) = \mathcal{O}(\log n)$, $\boldsymbol{W} \in \mathcal{H}(T_{I\times I}, k)$. Then the storage requirement of $\boldsymbol{W}$ and matrix vector multiplication cost $\mathcal{O}(kn \log n)$, matrix–matrix addition costs $\mathcal{O}(k^2 n \log n)$ and matrix–matrix product as well as matrix inverse cost $\mathcal{O}(k^2 n \log^2 n)$.*

Proof See [9,17,20]. □

Note that the result of addition of two hierarchical matrices $\boldsymbol{M}_1$ and $\boldsymbol{M}_2 \in \mathcal{H}(T_{I\times I}, k)$ is a matrix from $\mathcal{H}(T_{I\times I}, 2k)$. To have the sum $\boldsymbol{M}_1 + \boldsymbol{M}_2$ in the class $\mathcal{H}(T_{I\times I}, k)$ also, one should truncate the rank $2k$ to k.

4 $\mathcal{H}$-matrix approximation of covariance matrix C

Examples of the computational domain $\mathcal{G}$ are shown in Fig. 1.

Let $x = (x_1, \ldots, x_d)$ and $y = (y_1, \ldots, y_d) \in \mathcal{G}$. Define the (anisotropic) distance by

$$\rho = \sqrt{\sum_{i=1}^{d} |x_i - y_i|^2 / l_i^2}, \quad \text{where } l_i \text{ are correlation length scales, } d = 2, 3. \tag{18}$$

Typical examples of covariance functions are:

$$\text{(a)} \quad \operatorname{cov}(\rho) = e^{-\rho^2} \quad \text{(Gaussian)}, \tag{19}$$

$$\text{(b)} \quad \operatorname{cov}(\rho) = e^{-\rho} \quad \text{(exponential) and} \tag{20}$$

$$\text{(c)} \quad \operatorname{cov}(\rho) = \begin{cases} \left(1 - \frac{3}{2}\rho + \frac{1}{2}\rho^3\right) & \text{for } 0 \le \rho \le 1 \\ 0 & \text{for } \rho \ge 1 \end{cases} \quad \text{(spherical)}. \tag{21}$$

To demonstrate the accuracy of the $\mathcal{H}$-matrix approximation, we compute the following errors:

$$\varepsilon_2 := \frac{|\|C\|_2 - \|\tilde{C}\|_2|}{\|C\|_2}, \quad \varepsilon := \frac{\|(C - \tilde{C})z\|_2}{\|C\|_2 \|z\|_2}, \quad \text{where } z \text{ is a random vector.}$$

All the following numerical experiments are done on a computer with a 2GHz processor and with 3GB of memory. Table 1 shows the computing time and storage requirement for the $\mathcal{H}$-matrix approximation $\tilde{C}$ of C. One can see that $\tilde{C}$ needs much less memory and computing time than C. Table 2 demonstrates the dependence of computational resources on $\mathcal{H}$-matrix rank k for the standard (left) and weak (right) admissibility conditions. The matrix, obtained with the weak admissibility condition (see an example in Fig. 2, right), is simpler, but has a higher rank to achieve the same accuracy than the matrix obtained with the standard admissibility. For the cases $k = 6$ and $k = 20$ there are not enough memory (abbreviated as "nem").

Figure 2 shows two different examples of $\mathcal{H}$-matrix approximations to the discretised covariance function (20) with $l_1 = 0.15$ and $l_2 = 0.2$. For the matrix on the left

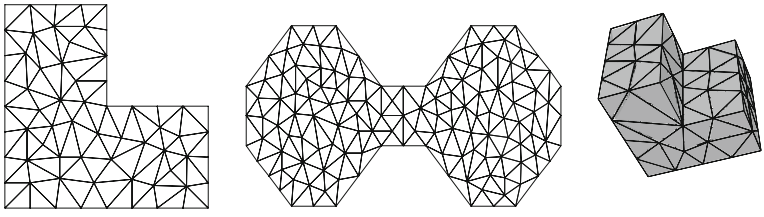

Fig. 1 Examples of computational domains $\mathcal{G}$ with a non-rectangular grid

Table 1 The accuracy of the $\mathcal{H}$-matrix approximation (weak admissibility) of the covariance function (20), $l_1 = l_3 = 0.1, l_2 = 0.5$

n	Rank k for $\tilde{C}$	Size (MB)		t (s)		ε	$\max_{i=1\ldots10} \lvert\lambda_i - \tilde{\lambda}_i\rvert, i$	ε_2
		C	$\tilde{C}$	C	$\tilde{C}$			
4.0×10^3	10	48	3	0.8	0.08	7×10^{-3}	$7.0 \times 10^{-2}, 9$	2.0×10^{-4}
1.05×10^4	18	439	19	7.0	0.4	7×10^{-4}	$5.5 \times 10^{-2}, 2$	1.0×10^{-4}
2.1×10^4	25	2054	64	45.0	1.4	1×10^{-5}	$5.0 \times 10^{-2}, 9$	4.4×10^{-6}

The geometry is shown in Fig. 1 (right)

Table 2 Dependence of the computing time and storage requirement on the $\mathcal{H}$-matrix rank k for the covariance function (20)

k	Size (MB)	t (s)	k	Size (MB)	t (s)
1	1548	33	4	463	11
2	1865	42	8	850	22
3	2181	50	12	1236	32
4	2497	59	16	1623	43
6	nem	–	20	nem	–

Left standard admissibility condition (12), geometry shown in Fig. 1 (middle), $l_1 = 0.1, l_2 = 0.5, n = 2.3 \times 10^5$. *Right* weak admissibility condition (14), geometry shown in Fig. 1 (right), $l_1 = 0.1, l_2 = 0.5, l_3 = 0.1, n = 4.61 \times 10^5$

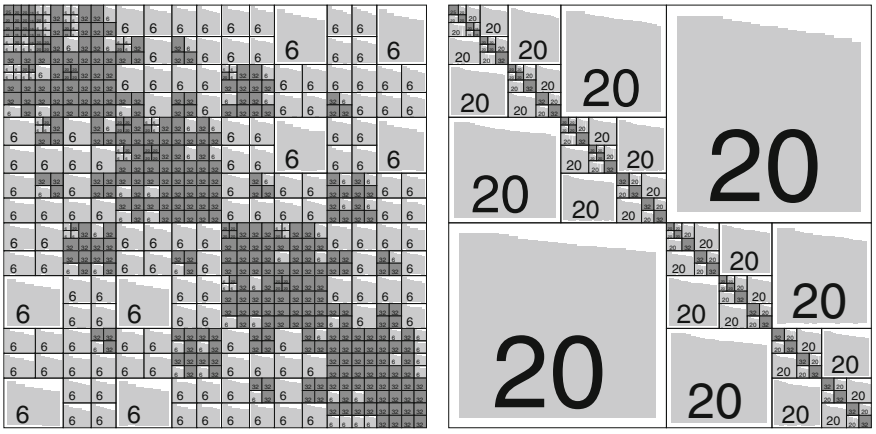

Fig. 2 Two examples of $\mathcal{H}$-matrix approximations $\in \mathbb{R}^{n\times n}, n = 32^2$, of the discretised covariance function $\mathrm{cov}(x, y) = e^{-\rho}, l_1 = 0.15, l_2 = 0.2, x, y \in [0, 1]^2$. The biggest dense (*dark*) blocks $\in \mathbb{R}^{32\times32}$, max. rank $k = 6$ on the *left* and $k = 20$ on the *right*. The *right* block structure is simpler, but the *left* structure is more accurate

the standard admissibility condition (12) was used and for the matrix on the right the weak admissibility condition (14). The dark blocks indicate the dense matrices and the gray blocks rank-k matrices. The steps inside blocks present the decay of singular

Table 3 Dependence of the computational time and storage cost on the problem size n, rank $k = 5$, $\mathrm{cov}(x, y) = e^{-\rho}$, $l_1 = l_2 = 1$, domain $\mathcal{G} = [0, 1]^2$

n	Time (s)		Memory (MB)		ε
	$\boldsymbol{C}$	$\tilde{\boldsymbol{C}}$	$\boldsymbol{C}$	$\tilde{\boldsymbol{C}}$	
33^2	0.14	0.01	9.5	0.7	4.3×10^{-3}
65^2	2.6	0.05	1.4×10^2	3.5	3.7×10^{-3}
129^2	–	0.24	nem	16	–
257^2	–	1	nem	64	–

Table 4 Dependence of the $\mathcal{H}$-matrix accuracy on the covariance lengths l_1 and l_2 for covariance function (20), $\mathcal{G} = [0, 1]^2$, $n = 129^2$

l_1	l_2	ε
0.01	0.02	3×10^{-2}
0.1	0.2	8×10^{-3}
0.5	1	2.8×10^{-5}

values in log scale. The approximation on the left has a more complex block structure, but has a smaller maximal rank k (=6). The approximation on the right has a less complex block structure, but the maximal rank k is larger (=20).

Table 3 demonstrates the computational resources needed for the $\mathcal{H}$-matrix approximation of the covariance function

$$\mathrm{cov}(x, y) = e^{-\rho}, \quad l_1 = l_2 = 1 \quad \text{(see (20))}.$$

For small problem sizes such as 33^2, 65^2 (in 2D) it is possible to compute the exact covariance matrix $\boldsymbol{C}$ and check the accuracy of the $\mathcal{H}$-matrix approximation. But for large problem sizes there is not enough memory (nem) to store the matrix $\boldsymbol{C}$. The last column presents the accuracy of the $\mathcal{H}$-matrix approximation.

One can see that $\mathcal{H}$-matrix approximations can be computed very fast even for 129^2 and 257^2 degrees of freedom, whereas for the dense matrices there is not enough memory.

Table 4 demonstrates the accuracy of the $\mathcal{H}$-matrix approximation of the covariance function (20) for different covariance lengths l_1 and l_2 .

5 Numerical computation of KLE

An analytical solution of the eigenvalue problem (2) is known very seldomly (usually only in 1D and for a small class of covariance functions). For instance, the solution of the eigenvalue problem (2) with $\mathrm{cov}(x, y) = e^{-\beta|x-y|}$, $x, y \in (-a, a) \subset \mathbb{R}$ is available in [13,14]. However already in 2D the analytical solutions are either more complex or almost impossible to deduce. In this section we solve the symmetric eigenvalue problem (7). We tested the ARPACK [27] and TRLAN [42] packages for computing m largest eigenvalues and corresponding eigenfunctions of (7). ARPACK is based upon an algorithmic variant of the Arnoldi process called the Implicitly

Table 5 t_1-computing times (in s) required for an $\mathcal{H}$-matrix and dense matrix vector multiplication, t_2-times to set up $\tilde{\boldsymbol{C}} \in \mathbb{R}^{n\times n}$

$k \setminus n$	1.05×10^4		2.4×10^4		3.5×10^4		6.8×10^4		2.3×10^5	
	t_1	t_2	t_1	t_2	t_1	t_2	t_1	t_2	t_1	t_2
3	8×10^{-4}	0.1	3×10^{-3}	0.2	6.0×10^{-3}	0.4	1×10^{-2}	1	5.0×10^{-2}	4
6	2×10^{-3}	0.15	6×10^{-3}	0.4	1.1×10^{-2}	0.7	2×10^{-2}	2	9.0×10^{-2}	7
9	3×10^{-3}	0.2	8×10^{-3}	0.5	1.5×10^{-2}	1.0	3×10^{-2}	3	1.3×10^{-1}	11
Full rank	0.13		0.62		2.48		10		140	

Restarted Arnoldi Method (IRAM). For symmetric matrices it reduces to a variant of the Lanczos process called the Implicitly Restarted Lanczos Method (IRLM) [27].

The TRLAN package targets the case where one wants both eigenvalues and eigenvectors of a large real symmetric eigenvalue problems that cannot use the shift-and-invert scheme. In this case the standard non-restarted Lanczos algorithm requires the storage of a large number of Lanczos vectors which can cause storage problem and make each iteration of the method very expensive. The algorithm used in TRLAN is a dynamic thick-restart Lanczos algorithm. The convergence test used in the TRLAN is the residual $r <$ tolerance $\cdot \|\tilde{\boldsymbol{C}}\|$ [42].

The three most time-consuming procedures in the Lanczos method are the matrix-vector multiplication, re-orthogonalisation and computation of the Ritz vectors. All matrix-vector products are approximated in the $\mathcal{H}$-matrix format with the cost $\mathcal{O}(n \log n)$. We also investigate how the $\mathcal{H}$-matrix technique reduces the memory requirements and the computing times of the eigenvalue solver.

Remark 5.1 Note that an $\mathcal{H}$-matrix approximation $\tilde{\boldsymbol{C}}$ of the symmetric matrix $\boldsymbol{C}$ is not always symmetric [6] (the possible reason is the rounding error). Therefore we take the symmetric part of $\tilde{\boldsymbol{C}}$. Note also that in HLIB [16] there is a possibility to set up only the upper (lower) half of the matrix.

In Table 5, one can see the computing times required for an $\mathcal{H}$-matrix vector multiplication. The table was made using the weak admissibility criteria (14). It approximates the covariance function (20) with $l_1 = l_3 = 0.1$ and $l_2 = 0.5$. The geometry is shown in Fig. 1 (right). The times needed to set up the $\mathcal{H}$-matrices are shown in parentheses. Numerical experiments confirm the theoretical estimation $\mathcal{O}(kn \log n)$ (from Theorem 3.1) for an $\mathcal{H}$-matrix vector multiplication. One can see a linear dependence on the rank k and an almost linear dependence on the problem size n. If the matrix $\boldsymbol{C}$ is stored in a dense matrix format then the complexity should be $\mathcal{O}(n^2)$ (the last row). Note that for $n = 3.5 \cdot 10^4$ and higher there is not enough memory to store $\boldsymbol{C}$. The corresponding computing times for $n \geq 3.5 \cdot 10^4$ are extrapolated from the previous values.

Tables 6 and 7 show the computing times which required TRLAN [42] to compute m eigenpairs. Computing times in Table 7 are larger than the times in Table 6. The reason is that TRLAN performs more iteration steps.

Table 6 Time required for computing m eigenpairs of the covariance function (20) with $l_1 = l_2 = l_3 = 1$

Matrix info (MB, s)				m					
n	k	Size of $\tilde{C}$	Time to set up $\tilde{C}$	2	5	10	20	40	80
2.4×10^4	4	12	0.2	0.2	0.2	0.4	0.7	1.8	5
6.8×10^4	8	95	0.7	0.7	0.8	1.6	3.4	7.0	19
2.3×10^5	12	570	6.8	3.6	4.0	7.2	15.0	31.0	75

The geometry is shown in Fig. 1 (right)

Table 7 Time required for computing m eigenpairs of the covariance function (20) with $l_1 = l_3 = 0.1$, $l_3 = 0.5$

Matrix info (MB, s)				m					
n	k	Size of $\tilde{C}$	Time to set up $\tilde{C}$	2	5	10	20	40	80
2.4×10^4	4	12	0.2	0.6	0.9	1.3	2.3	4.2	8
6.8×10^4	8	95	2	2.4	3.8	5.6	8.4	18.0	28
2.3×10^5	12	570	11	10.0	17.0	24.0	39.0	70.0	150

The geometry is shown in Fig. 1 (right)

6 Conclusion

We have successfully applied the $\mathcal{H}$-matrix technique for the approximation of covariance matrices (19–21) in 2D and 3D cases. The use of the $\mathcal{H}$-matrix technique reduces computational resources (Tables 1, 2, 3, 5), required by eigensolvers (e.g. ARPACK [27] and TRLAN [42]) for solving the eigenvalue problem (7), dramatically. The combination of the $\mathcal{H}$-matrix technique and iterative eigenvalue solvers are seen to be a very efficient way to compute the KLE (Tables 6, 7).

Acknowledgments The authors are appreciative to Eveline Rosseel (K.U.Leuven, Department of Computer Science, Belgium) for valuable comments. We would like also to thank our student Jeremy Rodriguez for the help in providing Tables 1, 2, 5, 6 and 7. It is also acknowledged that this research has been conducted within the project MUNA under the framework of the German Luftfahrtforschungsprogramm funded by the Ministry of Economics (BMWA).

References

1. Atkinson KE (1997) The numerical solution of integral equations of the second kind. Cambridge monographs on applied and computational mathematics, vol 4. Cambridge University Press, Cambridge
2. Babuška I, Tempone R, Zouraris GE (2004) Galerkin finite element approximations of stochastic elliptic partial differential equations. SIAM J Numer Anal 42(2):800–825 (electronic)
3. Babuška I, Nobile F, Tempone R (2007) A stochastic collocation method for elliptic partial differential equations with random input data. SIAM J Numer Anal 45(3):1005–1034 (electronic)

4. Bai Z, Demmel J, Dongarra J, Ruhe A, van der Vorst H (eds) (2000) Templates for the solution of algebraic eigenvalue problems— practical guide, vol 11 of Software, Environments, and Tools. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA
5. Bebendorf M (2000) Approximation of boundary element matrices. Numer Math 86(4):565–589
6. Bebendorf M, Hackbusch W (2007) Stabilized rounded addition of hierarchical matrices. Numer Linear Algebra Appl 14(5):407–423
7. Bebendorf M, Rjasanow S (2003) Adaptive low-rank approximation of collocation matrices. Computing 70(1):1–24
8. Börm S, Grasedyck L (2005) Hybrid cross approximation of integral operators. Numer Math 101(2):221–249
9. Börm S, Grasedyck L, Hackbusch W (2003) Hierarchical Matrices, Lecture Note. Max-Planck Institute for Mathematics, Leipzig
10. Eiermann M, Ernst OG, Ullmann E (2007) Computational aspects of the stochastic finite element method. Comput Vis Sci 10(1):3–15
11. Elman HC, Ernst OG, O'Leary DP, Stewart M (2005) Efficient iterative algorithms for the stochastic finite element method with application to acoustic scattering. Comput Methods Appl Mech Eng 194 (9–11):1037–1055
12. Frigo M, Johnson S (1998) FFTW: An adaptive software architecture for the FFT. In: Proc. ICASSP, IEEE, Seattle, WA, (3):1381–1384. http://www.fftw.org
13. Ghanem R, Spanos D (1991) Spectral stochastic finite-element formulation for reliability analysis. J Eng Mech 117(10):2351–2370
14. Ghanem R, Spanos P (1991) Stochastic finite elements: a spectral approach. Springer, New York
15. Goreinov SA, Tyrtyshnikov EE, Zamarashkin NL (1997) A theory of pseudoskeleton approximations. Linear Algebra Appl 261:1–21
16. Grasedyck L, Börm S: $\mathcal{H}$-matrix library. http://www.hlib.org
17. Grasedyck L, Hackbusch W (2003) Construction arithmetics of $\mathcal{H}$-matrices. Computing 70(4):295–334
18. Hackbusch W, Khoromskij BN (2000) A sparse $\mathcal{H}$-matrix arithmetic. II. Application to multi-dimensional problems. Computing 64(1):21–47
19. Hackbusch W (1995) Integral equations, vol 120 of International Series of Numerical Mathematics. Birkhäuser Verlag, Basel, 1995. Theory and numerical treatment, Translated and revised by the author from the 1989 German original
20. Hackbusch W (1999) A sparse matrix arithmetic based on $\mathcal{H}$-matrices. I. Introduction to $\mathcal{H}$-matrices. Computing 62(2):89–108
21. Hackbusch W (2004) Hierarchische Matrizen—Algorithmen und Analysis. Max-Planck-Institut für Mathematik, Leipzig
22. Hackbusch W, Khoromskij BN, Kriemann R (2004) Hierarchical matrices based on a weak admissibility criterion. Computing 73(3):207–243
23. Hackbusch W, Khoromskij BN, Tyrtyshnikov EE (2005) Hierarchical Kronecker tensor-product approximations. J Numer Math 13(2):119–156
24. Hida T, Kuo H-H, Potthoff J, Streit L (1993) White noise—an infinite-dimensional calculus, vol 253 of Mathematics and its Applications. Kluwer Academic Publishers Group, Dordrecht (1993)
25. Keese A (2004) Numerical solution of systems with stochastic uncertainties. A general purpose framework for stochastic finite elements. Ph.D. Thesis, TU Braunschweig, Germany
26. Lanczos C (1995) An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. J Res Nat Bur Standards 45:255–282
27. Lehoucq RB, Sorensen DC, Yang C (1998) ARPACK users' guide. Solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods, vol 6 of Software, Environments, and Tools. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA
28. Loève, M (1977) Probability theory I. Graduate Texts in Mathematics, vol 45, 6, 4th edn. Springer, New York
29. Matthies HG (2007) Uncertainty quantification with stochastic finite elements. Part 1. Fundamentals. Encyclopedia of Computational Mechanics. Wiley, New York
30. Matthies HG, Keese A (2005) Galerkin methods for linear and nonlinear elliptic stochastic partial differential equations. Comput Methods Appl Mech Eng 194(12–16):1295–1331
31. Pellissetti MF, Ghanem R (2000) Iterative solution of systems of linear equations arising in the context of stochastic finite elements. Adv Eng Softw 31(8–9):607–616

32. Press WH, Teukolsky SA, Vetterling WT, Flannery BP (1992) Numerical recipes in C—the art of scientific computing, 2nd edn. Cambridge University Press, Cambridge
33. Riesz F, Sz.-Nagy B (1990) Functional analysis. Dover Books on Advanced Mathematics. Dover Publications Inc., New York, 1990. Translated from the second French edition by Leo F. Boron, Reprint of the 1955 original
34. Saad Y (1992) Numerical methods for large eigenvalue problems. Manchester University Press, Manchester
35. Schwab C, Todor RA (2003) Sparse finite elements for elliptic problems with stochastic loading. Numer Math 95(4):707–734
36. Schwab C, Todor RA (2003) Sparse finite elements for stochastic elliptic problems—higher order moments. Computing 71:43–63
37. Schwab C, Todor RA (2006) Karhunen–Loève approximation of random fields by generalized fast multipole methods. J Comput Phys 217(1):100–122
38. Seynaeve B, Rosseel E, Nicolaï B, Vandewalle S (2007) Fourier mode analysis of multigrid methods for partial differential equations with random coefficients. J Comput Phys 224(1):132–149
39. Todor RA, Schwab C (2007) Convergence rates for sparse chaos approximations of elliptic problems with stochastic coefficients. IMA J Numer Anal 27(2):232–261
40. Werner D (2000) Funktionalanalysis, extended edn. Springer, Berlin
41. Wiener N (1938) The homogeneous chaos. Am J Math 60:897–936
42. Wu K, Simon H (2000) Thick-restart Lanczos method for large symmetric eigenvalue problems. SIAM J Matrix Anal Appl 22(2):602–616 (electronic)