MAX-PLANCK-GESELLSCHAFT

Max Planck Institute Magdeburg
Preprints

Peter Benner Akwum Onwunta Martin Stoll

Low Rank Solution of Unsteady Diffusion Equations with Stochastic Coefficients

MAX–PLANCK–INSTITUT
FÜR DYNAMIK KOMPLEXER
TECHNISCHER SYSTEME
MAGDEBURG

MPIMD/13-13 August 12, 2013

Abstract

We study the solution of linear systems resulting from the discreitization of unsteady diffusion equations with stochastic coefficients. In particular, we focus on those linear systems that are obtained using the so-called stochastic Galerkin finite element method (SGFEM). These linear systems are usually very large with Kronecker product structure and, thus, solving them can be both time- and computer memory-consuming. Under certain assumptions, we show that the solution of such linear systems can be approximated with a vector of low tensor rank. We then solve the linear systems using low rank preconditioned iterative solvers. Numerical experiments demonstrate that these low rank preconditioned solvers are effective.

Impressum:

Max Planck Institute for Dynamics of Complex Technical Systems, Magdeburg

Publisher:
Max Planck Institute for
Dynamics of Complex Technical Systems

Address:
Max Planck Institute for
Dynamics of Complex Technical Systems
Sandtorstr. 1
39106 Magdeburg

www.mpi-magdeburg.mpg.de/preprints

1 Introduction

Many problems in science and engineering are modelled using partial differential equations (PDEs). One of such important models is the diffusion equation which arises in, for instance, fluid flow and transport of chemicals in heterogeneous porous media (see e.g. [6], [21]), as well as in temperature prediction of biological bodies, [25], etc. More often than not, the diffusion equation is modelled deterministically. However, in the transport models for groundwater flows, for example, it is only possible to measure the hydraulic conductivity at a limited number of spatial locations; this leads to uncertainty in the groundwater flow simulations, [6]. Hence, it is reasonable to model the hydraulic conductivity as a random field. This, in turn, implies that the solution to the resulting stochastic model is necessarily also a random field. There is, therefore, the need to quantify the uncertainty in the solution of the model.

Generally, in order to solve PDEs with stochastic inputs, three competing methods are standard in the literature: the Monte Carlo method (MCM), the stochastic collocation method (SCM) and the stochastic Galerkin finite element method (SGFEM), see e.g. [6], [1], [3], [14], [12]. In contrast to MCM and SCM (both of which are based on stochastic sampling), SGFEM is a non-sampling approach which transforms a PDE with uncertain inputs into a large system of coupled deterministic PDEs. Despite this curse of dimensionality problem associated with the SGFEM, the beauty of the approach lies, among others, in the ease with which it lends itself to the computation of such quantities of interest as the moments and the density of the solution. In this paper, our key objective is to study the solution of systems resulting from the discretization of unsteady diffusion equations with stochastic coefficients, using the SGFEM.

In the past two decades, research on the solution of diffusion equations with random inputs using the SGFEM has been focused mainly on developing solvers for steady-state problems, see e.g. [1], [9], [23], [11], [21], etc. Time-dependent problems have not yet received adequate attention. A few attempts in this direction include [25], [20], [22] and [26]. Unlike the steady-state problem, the time-dependent model problem presents the additional challenge of solving a large coupled linear system for each time step. As opposed to the literature above on unsteady diffusion problems, the main aim of this paper is to tackle this dimensionality problem using low rank iterative solvers studied in [19] in the framework of parametrized linear systems. The rest of the paper is organised as follows. In Section 2, we give some basic notions on which we shall rely in the rest of the paper. Next, we present our model problem and provide an overview of its discretization in Section 3. Since our approach is based on low rank approximation, we first show the existence of a low rank approximation of the solution to the stochastic Galerkin system in Section 4 before proceeding to discuss our preconditioned low rank iterative solvers in Section 5 and numerical results in Section 6. Finally, we draw some conclusions in Section 7 based on our findings in the paper.

2 Basic notions and definitions

Let the triplet $(\Omega, \mathcal{F}, \mathcal{P})$ be a complete probability space, where Ω is a sample space of events. Here, $\mathcal{F}$ denotes a σ-algebra on Ω and is endowed with an appropriate probability measure $\mathcal{P}$. Moreover, let $\mathcal{D} \subset \mathbb{R}^d$ with $d \in \{1, 2, 3\}$, be a bounded open set with Lipschitz boundary $\partial\mathcal{D}$.

Definition. *A mapping $\kappa : \mathcal{D} \times \Omega \to \mathbb{R}$ is called a random field if for each fixed $\mathbf{x} \in \mathcal{D}$, $\kappa(\mathbf{x}, \cdot)$ is a random variable with respect to $(\Omega, \mathcal{F}, \mathcal{P})$.*

We denote the mean of κ at a point $\mathbf{x} \in \mathcal{D}$ by $\bar{\kappa}(\mathbf{x}) := \langle \kappa(\mathbf{x}, \cdot) \rangle$. The covariance of κ at $\mathbf{x}, \mathbf{y} \in \mathcal{D}$ is given by

$$\mathrm{Cov}_\kappa(\mathbf{x}, \mathbf{y}) := \langle (\kappa(\mathbf{x}, \cdot) - \bar{\kappa}(\mathbf{x}))(\kappa(\mathbf{y}, \cdot) - \bar{\kappa}(\mathbf{y})) \rangle . \tag{1}$$

Note that the variance $\mathrm{Var}(\kappa) = \sigma_\kappa^2$ of κ at $\mathbf{x} \in \mathcal{D}$ is obtained if we set $\mathbf{x} = \mathbf{y}$ in (1) and the standard deviation of κ is $\sqrt{\mathrm{Var}(\kappa)}$. Let $L^2(\Omega, \mathcal{F}, \mathcal{P})$ denote the space of square-integrable random fields defined on $(\Omega, \mathcal{F}, \mathcal{P})$.

We shall also need the concepts of Kronecker product and $\mathrm{vec}(\cdot)$ operators.

Definition. *Let $X = [x_1, \ldots, x_m] \in \mathbb{R}^{n \times m}$ and $Y \in \mathbb{R}^{p \times q}$. Then*

$$X \otimes Y = \begin{bmatrix} x_{11}Y \ldots x_{1m}Y \\ \vdots \qquad\quad \vdots \\ x_{1n}Y \ldots x_{nm}Y \end{bmatrix} \in \mathbb{R}^{np \times mq}, \quad \mathrm{vec}(X) = \begin{bmatrix} x_1 \\ \vdots \\ x_{nm} \end{bmatrix} \in \mathbb{R}^{nm \times 1}. \tag{2}$$

It follows from (2) that the $\mathrm{vec}(\cdot)$ operator essentially reshapes a matrix into a column vector. In MATLAB notation, for example, we have

```
vec(X)=reshape(X,n*m,1).
```

More precisely, we consider the $\mathrm{vec}(\cdot)$ operator as a vector space isomorphism $\mathrm{vec} : \mathbb{R}^{n \times m} \to \mathbb{R}^{nm}$ and denote its inverse by $\mathrm{vec}^{-1} : \mathbb{R}^{nm} \to \mathbb{R}^{n \times m}$. Kronecker product and $\mathrm{vec}(\cdot)$ operators exhibit the following properties, see e.g. [8].

$$\mathrm{vec}(AXB) = (B^T \otimes A)\mathrm{vec}(X), \tag{3}$$

$$(A \otimes B)(C \otimes D) = AC \otimes BD. \tag{4}$$

Finally, we introduce the tensor rank of a vectorized matrix, see e.g., [16].

Definition. *Let $X \in \mathbb{R}^{n \times n}$ and $\mathbf{x} = \mathrm{vec}(X) \in \mathbb{R}^{n^2}$. Then, the tensor rank of $\mathbf{x}$ is the smallest $k \in \mathbb{Z}_+$ such that*

$$\mathbf{x} = \sum_{i=1}^{k} u_i \otimes v_i,$$

where $u_i, v_i \in \mathbb{R}^n$. In particular, the tensor rank of the vector $\mathbf{x}$ coincides with the rank of the matrix X.

3 A model problem with stochastic inputs

In this section, we introduce and discretize our model problem. More precisely, we study the solution of the stochastic initial-boundary value problem of the form

$$\left.\begin{aligned} \frac{\partial u(\mathbf{x},\omega,t)}{\partial t} &= \nabla\cdot(\kappa(\mathbf{x},\omega)\nabla u(\mathbf{x},\omega,t)) + f(\mathbf{x}), \quad \text{in } \mathcal{D}\times\Omega\times(0,T], \\ u(\mathbf{x},\omega,t) &= 0, \;\; \mathbf{x}\in\partial\mathcal{D},\; \omega\in\Omega,\; t\in T, \\ u(\mathbf{x},\omega,0) &= 0, \;\; \mathbf{x}\in\mathcal{D},\; \omega\in\Omega, \end{aligned}\right\} \tag{5}$$

where, for ease of exposition, we restrict our discussion to a sufficiently smooth, time-independent deterministic source term, as well as Dirichlet boundary conditions. However, our discussion naturally generalizes to other stochastic boundary conditions and stochastic time-dependent source terms. In the model (5), we note here that $\kappa(\mathbf{x},\omega)$, and hence the solution $u(\mathbf{x},\omega,t)$ are random fields. We assume that the random input κ is $\mathcal{P}$-almost surely uniformly positive; that is,

$$\exists\, \alpha,\beta \;\text{ such that }\; 0<\alpha\leq\beta<+\infty,$$

with

$$\alpha\leq\kappa(\mathbf{x},\omega)\leq\beta, \quad \text{a.e. in } \mathcal{D}\times\Omega. \tag{6}$$

The well-posedness of the model (5) then follows from classical the Lax-Milgram Lemma (see e.g. [21]).

Next, following [22], we review the discretization of (5) using the stochastic Galerkin finite element method (SGFEM).

3.1 Overview of stochastic Galerkin finite element method

As we noted in Section 1, the discretization of partial differential equations with random coefficients using classical SGFEM is standard in the literature. Indeed, one usually follows a four-step procedure in this method, see e.g., [25], [22], [21]. The randomness in the model is, first of all, represented with a finite number of random variables. Then, we use the Karhunen-Lòeve expansion (KLE) to decouple the random and spatial dependencies in the random field, κ. Next, we approximate the solution as a finite-term expansion using basis orthogonal polynomials – the so-called generalized polynomial chaos expansion (PCE). The fourth and final stage entails performing a Galerkin projection on the set of polynomial basis functions. The above procedure transforms the stochastic problem (5) to a system of (usually) large coupled system of deterministic diffusion equations, which can then be solved with the appropriate methods for deterministic PDEs. In what follows, we briefly review the four steps.

3.2 Karhunen-Lòeve representation of stochastic inputs

Let $\kappa : \mathcal{D} \times \Omega \to \mathbb{R}$ be a random field with continuous covariance function $C_\kappa(\mathbf{x}, \mathbf{y})$. Then κ admits a proper orthogonal decomposition (or KLE)

$$\kappa(\mathbf{x}, \omega) = \bar{\kappa}(\mathbf{x}) + \sigma_\kappa \sum_{i=1}^{\infty} \sqrt{\lambda_i} \varphi_i(\mathbf{x}) \xi_i(\omega), \tag{7}$$

where σ_κ is the standard deviation of κ. The random variables $\xi := \{\xi_1, \xi_2, \ldots\}$ are centred, normalized and uncorrelated (but not necessarily independent) with

$$\xi_i(\omega) = \frac{1}{\sigma_\kappa \sqrt{\lambda_i}} \int_{\mathcal{D}} (\kappa(\mathbf{x}, \omega) - \bar{\kappa}(\mathbf{x})) \varphi_i(\mathbf{x}) \, d\mathbf{x},$$

and $\{\lambda_i, \varphi_i\}$ is the set of eigenvalues and eigenfunctions corresponding to $C_\kappa(\mathbf{x}, \mathbf{y})$. In other words, the eigenpairs $\{\lambda_i, \varphi_i\}$ solve the integral equations

$$\int_{\mathcal{D}} C_\kappa(\mathbf{x}, \mathbf{y}) \varphi_i(\mathbf{y}) \, d\mathbf{y} = \lambda_i \varphi_i(\mathbf{x}).$$

The eigenfunctions $\{\varphi_i\}$ form a complete orthogonal basis in $L^2(\mathcal{D})$. The eigenvalues $\{\lambda_i\}$ form a sequence of non-negative real numbers decreasing to zero. In practice, the series (7) is truncated after, say, N terms based on the speed of decay of the eigenvalues since the series converges in $L^2(\mathcal{D} \times \Omega)$ due to

$$\sum_{i=1}^{\infty} \lambda_i = \int_{\Omega} \int_{\mathcal{D}} (\kappa(\mathbf{x}, \omega) - \bar{\kappa}(\mathbf{x}))^2 \, d\mathbf{x} d\mathcal{P}(\omega).$$

However, one has to ensure that the truncated random field

$$\kappa_N(\mathbf{x}, \omega) = \bar{\kappa}(\mathbf{x}) + \sigma_\kappa \sum_{i=1}^{N} \sqrt{\lambda_i} \varphi_i(\mathbf{x}) \xi_i(\omega), \tag{8}$$

satisfies the positivity condition (6) so that the model (5) is still well-posed. It should be noted, though, that the truncated KLE (8) is a finite representation of $\kappa(\mathbf{x}, \omega)$ with the minimal mean-square error over all such finite representations.

For some random inputs, the covariance functions and eigenpairs can be computed explicitly. If they are not known a priori, then they can be approximated numerically, see e.g. [12] for details regarding the computation and convergence of KLE.

3.3 Generalized polynomial chaos expansion

Generalized polynomial chaos expansion is a means of representing a random field $u \in L^2(\Omega, \mathcal{F}, \mathcal{P})$ parametrically through a set of random variables. More precisely, we have

$$u(\mathbf{x}, \omega, t) = \sum_{j=0}^{\infty} u_j(\mathbf{x}, t) \psi_j(\xi(\omega)), \tag{9}$$

where u_j, the deterministic modes of the expansion, are given by

$$u_j(\mathbf{x}, t) = \frac{\langle u(\mathbf{x}, \omega, t)\psi_j(\xi)\rangle}{\langle \psi_j^2(\xi)\rangle},$$

ξ is a finite-dimensional random vector as in (8) and ψ_j are multivariate orthogonal polynomials satisfying

$$\langle \psi_0(\xi)\rangle = 1, \;\; \langle \psi_j(\xi)\rangle = 0, \; j > 0, \;\; \langle \psi_j(\xi)\psi_k(\xi)\rangle = \delta_{jk},$$

with

$$\begin{aligned} \langle \psi_j(\xi)\psi_k(\xi)\rangle &= \int_{\omega\in\Omega} \psi_j(\xi(\omega))\psi_k(\xi(\omega))\, d\mathcal{P}(\omega) && (10) \\ &= \int_{\xi\in\Pi} \psi_j(\xi)\psi_k(\xi)\rho(\xi)\, d\xi, && (11) \end{aligned}$$

where Π and ρ are, respectively, the support and probability density of ξ. The random variables are chosen such that their probability density coincides with the weight function of the orthogonal polynomials used in the expansion, e.g., Hermite polynomials and Gaussian random variables, Legendre polynomials and uniform random variables, Jacobi polynomials and beta random variables, etc. Note that n-dimensional orthogonal polynomials are constructed by taking n products of 1-dimensional orthogonal polynomials.

By the Cameron-Martin Theorem, the series (9) converges in the Hilbert space $L^2(\Omega, \mathcal{F}, \mathcal{P})$, see e.g. [10]. Thus, as in the case of KLE, we truncate (9) after, say, P terms to obtain

$$u(\mathbf{x}, \omega, t) = \sum_{j=0}^{P} u_j(\mathbf{x}, t)\psi_j(\xi(\omega)), \tag{12}$$

where P is determined by the expression

$$P = \frac{(N+Q)!}{N!Q!}. \tag{13}$$

In (13) above, Q is the highest degree of the orthogonal polynomial used to represent u. A detailed discussion on how to choose Q (and hence P) can be found in, for instance, [21].

Observe from (7) and (9) that the expansions decouple the random fields into stochastic and deterministic dependencies. Besides, the KLE in (7) is a special case of the PCE in (9) with $Q = 1$.

3.4 Stochastic Galerkin approach

If we substitute the expressions (8) and (12) into the model (5), we get

$$\begin{aligned} \sum_{i=0}^{P} \frac{\partial u_i(\mathbf{x}, t)}{\partial t}\psi_i &= \sum_{i=0}^{P} \nabla \cdot \left(\left(\bar{\kappa}(\mathbf{x}) + \sigma_\kappa \sum_{k=1}^{N} \sqrt{\lambda_k}\varphi_k(\mathbf{x})\xi_k \right) \nabla u_i(\mathbf{x}, t)\psi_i \right) \\ &+ \; f(\mathbf{x}). && (14) \end{aligned}$$

Next, we project (14) onto the space spanned by the $P+1$ polynomial chaos basis functions to obtain, for $j = 0, 1, \ldots, P$,

$$\left\langle \psi_j^2 \right\rangle \frac{\partial u_j(\mathbf{x},t)}{\partial t} = \sum_{i=0}^{P} \nabla \cdot (a_{ij}(\mathbf{x}) \nabla u_i(\mathbf{x},t)) + \langle \psi_j \rangle f(\mathbf{x}), \tag{15}$$

where

$$\begin{aligned} a_{ij}(\mathbf{x}) &= \bar{\kappa}(\mathbf{x}) \langle \psi_i \psi_j \rangle + \sigma_\kappa \textstyle\sum_{k=1}^{N} \sqrt{\lambda_k} \varphi_k(\mathbf{x}) \langle \xi_k \psi_i \psi_j \rangle \\ &= \bar{\kappa}(\mathbf{x}) \delta_{ij} + \sigma_\kappa \textstyle\sum_{k=1}^{N} \sqrt{\lambda_k} \varphi_k(\mathbf{x}) \langle \xi_k \psi_i \psi_j \rangle . \end{aligned} \tag{16}$$

It should be noted that the system of $P+1$ deterministic diffusion equations in (15) are coupled. Designing a fast solver for such a large coupled system can be quite a challenge. This is the main purpose of the remainder of this paper. However, if doubly orthogonal polynomials (see e.g. [2]) are used, then one obtains a decoupled system which can be solved relatively easily. We do not consider the latter case in this paper.

In practice, the quantity of interest is not the solution u of the model (5) itself; rather, one is usually interested in some functional of u. Once the modes $u_i,\ i = 0, 1, \ldots, P$, have been computed, the intended quantities of interest, such as the moments and probability density of the solution can easily be deduced. For instance, the mean and the variance of the solution are, respectively, given explicitly by

$$\langle u(\mathbf{x}, \xi, t) \rangle = u_0(\mathbf{x}, t), \;\; \mathrm{Var}(u(\mathbf{x}, \xi, t)) = \sum_{i=1}^{P} u_i^2(\mathbf{x}, t) \left\langle \psi_i^2(\xi) \right\rangle .$$

3.5 Spatial and time discretizations

In the spirit of [21] and [22], we use classical finite elements to discretize the spatial domain. Furthermore, we assume that each of the deterministic coefficients $u_i,\ i = 0, 1, \ldots, P$, in (15) is discretized on the same mesh and with equal number of elements. More precisely, with J basis functions $s_j(\mathbf{x})$, each mode u_i is approximated as a linear combination of the form

$$u_i(\mathbf{x}, t) \approx \sum_{j=1}^{J} u_{ij}(t) s_j(\mathbf{x}), \;\; i = 0, \ldots, P.$$

After spatial discretization and some algebraic manipulations (see e.g [21]), one gets the system of ordinary differential equations:

$$(G_0 \otimes M) \frac{d\mathbf{u}(t)}{dt} + \left(\sum_{i=0}^{N} G_i \otimes K_i \right) \mathbf{u}(t) = \mathbf{g}_0 \otimes \mathbf{f}_0, \tag{17}$$

where

$$\mathbf{u}(t) = \begin{bmatrix} u_0(t) \\ \vdots \\ u_P(t) \end{bmatrix}, \;\; \text{with } \; u_i(t) \in \mathbb{R}^J, \; i = 0, 1, \ldots, P. \tag{18}$$

The stochastic matrices $G_i \in \mathbb{R}^{(P+1)\times(P+1)}$ are given by

$$G_0(j,k) = \langle \psi_j(\xi)\psi_k(\xi) \rangle \,, \quad G_i(j,k) = \langle \xi_i \psi_j(\xi)\psi_k(\xi) \rangle \,, \quad i = 1,\dots,N, \tag{19}$$

and the vectors $\mathbf{g}_0$ and $\mathbf{f}_0$ are defined via

$$\mathbf{g}_0(j) = \langle \psi_j(\xi) \rangle \,, \quad \mathbf{f}_0(j) = \int_{\mathcal{D}} f(\mathbf{x}) s_j(\mathbf{x}) \, d\mathbf{x}. \tag{20}$$

The mass matrix $M \in \mathbb{R}^{J\times J}$ and the stiffness matrices $K_i \in \mathbb{R}^{J\times J}$, $i = 0,1,\dots,N$, are given, respectively, by

$$M(j,k) = \int_{\mathcal{D}} s_j(\mathbf{x}) s_k(\mathbf{x}) \, d\mathbf{x}, \tag{21}$$

$$K_0(j,k) = \int_{\mathcal{D}} \bar{\kappa}(\mathbf{x}) \nabla s_j(\mathbf{x}) \nabla s_k(\mathbf{x}) \, d\mathbf{x}, \tag{22}$$

$$K_i(j,k) = \sigma_\kappa \sqrt{\lambda_i} \int_{\mathcal{D}} \varphi_i(\mathbf{x}) \nabla s_j(\mathbf{x}) \nabla s_k(\mathbf{x}) \, d\mathbf{x}. \tag{23}$$

Observe, in particular, from (22) and (23) that the matrix K_0 contains the mean information of the random field κ, whereas the matrices K_i, $i > 0$, capture the fluctuations therein.

For time discretization, we use implicit Euler to avoid stability issues. To this end, we set $t_n = n\tau$, $n = 0,1,\dots,T_{\max}$, with $\tau = T/T_{\max}$. Moreover, we define the computed numerical approximation $\mathbf{u}(t_n) := \mathbf{u}^n$, so that (17) yields

$$G_0 \otimes M \left(\frac{\mathbf{u}^n - \mathbf{u}^{n-1}}{\tau} \right) + \left(\sum_{i=0}^{N} G_i \otimes K_i \right) \mathbf{u}^n = \left(\mathbf{g}_0 \otimes \mathbf{f}_0 \right)^n, \tag{24}$$

or, equivalently,

$$\mathcal{A}\mathbf{u}^n = \mathbf{b}^n, \tag{25}$$

where

$$\mathbf{b}^n = \left(G_0 \otimes M \right) \mathbf{u}^{n-1} + \tau \left(\mathbf{g}_0 \otimes \mathbf{f}_0 \right)^n, \tag{26}$$

and

$$\begin{aligned} \mathcal{A} &= G_0 \otimes M + \tau \sum_{i=0}^{N} G_i \otimes K_i \\ &= G_0 \otimes (M + \tau K_0) + \tau \sum_{i=1}^{N} G_i \otimes K_i \\ &= G_0 \otimes \tilde{K}_0 + \sum_{i=1}^{N} G_i \otimes \tilde{K}_i, \end{aligned} \tag{27}$$

with $\tilde{K}_0 := M + \tau K_0,\ \ \tilde{K}_i = \tau K_i,\ i = 1, \ldots, N.$

We note that the stochastic Galerkin matrix $\mathcal{A}$ as defined in (27) is sparse in the block sense, symmetric and positive definite. Indeed, in such practical applications as flow problems, the length N of the random vector ξ is usually large due to the presence of small correlation length in the covariance function of κ. This, in turn, increases the value of P in (13) (and hence the dimension of $\mathcal{A}$) quite fast, see e.g., [12]. This is a major drawback of the SGFEM. To tackle this problem, we consider low rank approximation to the solution of the linear system (25).

4 Existence of low rank solution of the Galerkin system

In what follows, we focus our attention on the solution of the system (25) using two iterative solvers. First, however, following [4], we show, under certain conditions, that the solution of (27) can be approximated with a vector of low Kronecker rank. To this end, for arbitrary $\mathcal{A} \in \mathbb{R}^{m\times m}$ and $\mathbf{b} \in \mathbb{R}^m$, consider the following linear system

$$\mathcal{A}\mathbf{x} = \mathbf{b}. \tag{28}$$

Define, for $k \in \mathbb{N}$, the following quadrature points and weights

$$\begin{aligned} h_{st} &= \pi^2/\sqrt{k}, & (29)\\ t_j &= \log\left(\exp(jh_{st}) + \sqrt{1+\exp(2jh_{st})}\right), & (30)\\ w_j &= h_{st}/\sqrt{1+\exp(-2jh_{st})}. & (31) \end{aligned}$$

Our point of departure is the following lemma from [16].

Lemma 1. *Let the matrix $\mathcal{A} \in \mathbb{R}^{m\times m}$ in (28) be symmetric and positive definite. Suppose that the spectrum of $\mathcal{A}$ is contained in the strip $\Lambda := [\lambda_{\min}, \lambda_{\max}] \subset \mathbb{R}_+$ and let Γ be the boundary of $[1, 2\lambda_{\min}/\lambda_{\max} + 1]$. Let $k \in \mathbb{N}$ and $j = -k, \ldots, k$. Then the solution $\mathbf{x} = \mathcal{A}^{-1}\mathbf{b}$ to the system (28) can be approximated by*

$$\tilde{\mathbf{x}} := -\sum_{j=-k}^{k} \frac{2w_j}{\lambda_{\min}} \exp\left(-\frac{2t_j}{\lambda_{\min}}\mathcal{A}\right)\mathbf{b}, \tag{32}$$

with the approximation error

$$||\mathbf{x} - \tilde{\mathbf{x}}||_2 \leq \frac{C_{st}}{\pi\lambda_{\min}} \exp\left(\frac{1}{\pi} - \pi\sqrt{k}\right)|\Gamma|||\mathbf{b}||_2, \tag{33}$$

where $|\Gamma|$ is the length of Γ and the quadrature weights t_j, w_j are given by (30) and (31).

A sharper bound can, in fact, be obtained in (33) if $\mathcal{A}$ possesses some special Kronecker product structure, see e.g. [18]. Next, we recall the so-called the Sherman-Morrison-Woodbury formula (see e.g. [15]), on which, together with Lemma 1, we shall rely to prove our main result.

Lemma 2. *Let $X \in \mathbb{R}^{n\times n}$ be nonsingular and let $Y, Z \in \mathbb{R}^{n\times m}$, with $m \leq n$. Then $X + YZ^T$ is invertible if and only if $I_m + Z^T X^{-1} Y$ is invertible, with*

$$(X + YZ^T)^{-1} = X^{-1} - X^{-1}Y(I_m + Z^T X^{-1}Y)^{-1}Z^T X^{-1}. \tag{34}$$

We can now state our main result, which shows that the solution of the system (25) can indeed be approximated with a vector of low tensor rank. For this purpose, we split the matrix (27) as follows:

$$\mathcal{A} = \underbrace{G_0 \otimes \tilde{K}_0}_{=\mathcal{L}} + \sum_{i=1}^{N} G_i \otimes \tilde{K}_i. \tag{35}$$

Observe then from (19), (21), (22) and (27) that $\mathcal{L}$ in (35) is symmetric and positive definite. Furthermore, let the stochastic matrices $G_i,\ i = 1, \ldots, N$, be decomposed in low rank format:

$$G_i := U_i V_i^T,\ \ U_i, V_i \in \mathbb{R}^{(P+1)\times r_i},\ i = 1, \ldots, N. \tag{36}$$

Since also the stiffness matrices $\tilde{K}_i,\ i = 1, \ldots, N$, are symmetric, then each of them admits the factorization:

$$\tilde{K}_i := L_i D_i L_i^T = \tilde{L}_i L_i^T,\ \ \tilde{L}_i, L_i \in \mathbb{R}^{J\times J},\ i = 1, \ldots, N, \tag{37}$$

where $\tilde{L}_i := L_i D_i,\ i = 1, \ldots, N$, with D_i and L_i (and hence $\tilde{L}_i$) being, respectively, diagonal and lower triangular matrices. The following result holds.

Theorem 1. *Let $\mathcal{A}$ denote a matrix of Kronecker product structure as in (27). Assume that the spectrum of $\mathcal{L}$ in (35) is contained in the strip $\Lambda := [\lambda_{\min}, \lambda_{\max}] \subset \mathbb{R}_+$ and let Γ be the boundary of $[1, 2\lambda_{\max}/\lambda_{\min} + 1]$. Let $G_i,\ i = 1, \ldots, N$, have the low rank representation (36) with $r = \sum_{j=1}^{N} r_j$, and let $\tilde{K}_i,\ i = 1, \ldots, N$, be given by the decomposition (37). Suppose further that $U = [U_1 \otimes \tilde{L}_1, \ldots, U_N \otimes \tilde{L}_N]$ and $V = [V_1 \otimes L_1, \ldots, V_N \otimes L_N]$. For all time steps $n \geq 2$, let the tensor rank of $\mathbf{b}^n \leq \ell$, where $\ell \ll J(P+1)$. Then, for $k \in \mathbb{N}$, the solution $\mathbf{u}^n$ of (25) can be approximated by a vector $\tilde{\mathbf{u}}^n$ of the form*

$$\tilde{\mathbf{u}}^n = -\sum_{j=-k}^{k} \frac{2w_j}{\lambda_{\min}} \left(\exp(G_0) \otimes \exp\left(-\frac{2t_j}{\lambda_{\min}}\tilde{K}_0\right)\right)[\mathbf{b}^n - U\mathcal{Y}], \tag{38}$$

where the vector $\mathcal{Y} \in \mathbb{R}^{J\cdot r}$ is the solution of

$$(I_{J\cdot r} + V^T \mathcal{L}^{-1} U)\mathcal{Y} = V^T \mathcal{L}^{-1} \mathbf{b}^n, \tag{39}$$

and t_j, w_j are the quadrature weights and points as given by (30) and (31). The corresponding approximation error is given by

$$||\mathbf{u}^n - \tilde{\mathbf{u}}^n||_2 \ \ \leq\ \ \frac{C_{st}}{\pi\lambda_{\min}} \exp\left(\frac{1}{\pi} - \pi\sqrt{k}\right) |\Gamma| ||\mathbf{b}^n - U\mathcal{Y}||_2. \tag{40}$$

Moreover, the tensor rank of $\tilde{\mathbf{u}}^n$ in (38) is at most *(i)* $(2k+1)\cdot(r+1)$, *if* $n=1$, *and*

(ii) $(2k+1)\cdot(r+\ell)$, *if* $n\geq 2$.

Proof. Observe first from (4), (36) and (37) that we have the low rank representation

$$\sum_{i=1}^{N} G_i\otimes\tilde{K}_i=\sum_{i=1}^{N}(U_iV_i^T)\otimes(\tilde{L}_iL_i^T)=\sum_{i=1}^{N}(U_i\otimes\tilde{L}_i)(V_i^T\otimes L_i^T)=UV^T. \tag{41}$$

Hence, from Lemma 2, (35) and (41), we note that

$$\mathcal{A}^{-1}=(\mathcal{L}+UV^T)^{-1}=\mathcal{L}^{-1}-\mathcal{L}^{-1}U(I_{J\cdot r}+V^T\mathcal{L}^{-1}U)^{-1}V^T\mathcal{L}^{-1},$$

so that

$$\mathbf{u}^n=\mathcal{A}^{-1}\mathbf{b}^n\Leftrightarrow\mathbf{u}^n=\mathcal{L}^{-1}\left[\mathbf{b}^n-U\underbrace{(I_{J\cdot r}+V^T\mathcal{L}^{-1}U)^{-1}V^T\mathcal{L}^{-1}\mathbf{b}^n}_{=\mathcal{Y}}\right]. \tag{42}$$

Now, by definition, the matrix $\mathcal{L}=G_0\otimes\tilde{K}_0$ is symmetric and positive definite. Thus, using the fact that

$$\begin{aligned}\exp(-\beta\mathcal{L}) &= \exp(-\beta(G_0\otimes\tilde{K}_0))\\ &= \exp(G_0\otimes(-\beta\tilde{K}_0))\\ &= \exp(G_0)\otimes\exp(-\beta\tilde{K}_0),\end{aligned}$$

where $\beta:=2t_j/\lambda_{\min}$, together with (42) and Lemma 1, immediately yields (38) and (40).

To show (i), it suffices to show that the tensor rank of $\mathbf{b}^1-U\mathcal{Y}$ is at most $r+1$. Now, note that

$$\operatorname{rank}(\operatorname{vec}^{-1}(\mathbf{b}^1-U\mathcal{Y}))\leq\operatorname{rank}(\operatorname{vec}^{-1}(\mathbf{b}^1))+\operatorname{rank}(\operatorname{vec}^{-1}(-U\mathcal{Y})). \tag{43}$$

From (26), we see that $\mathbf{b}^1=\tau\left(\mathbf{g}_0\otimes\mathbf{f}_0\right)$, since $\tilde{\mathbf{u}}^0=0$ and the source term f is time-independent. But then, since the orthogonal polynomials $\{\psi_j\}$ satisfy

$$\mathbf{g}_0(j)=\langle\psi_j\rangle=\begin{cases}1, & j=0,\\ 0, & \text{otherwise},\end{cases}$$

it follows from (20) that $\operatorname{vec}^{-1}(\mathbf{g}_0\otimes\mathbf{f}_0)\in\mathbb{R}^{J\times(P+1)}$ is a matrix of rank 1. Hence, $\mathbf{b}^1$ is a vector of tensor rank 1. Next, following similar arguments as in the proof of Theorem 1 in [4], we show that the tensor rank of $U\mathcal{Y}$ is r, which, together with (43), completes the proof of (i). Now, let $\mathcal{Y}_{r_i}$ denote $J\cdot r_i$ elements of $\mathcal{Y}$, and observe from

(3) that

$$\begin{aligned} U\mathcal{Y} &= [U_1 \otimes \tilde{L}_1, \ldots, U_N \otimes \tilde{L}_N]\mathcal{Y} \\ &= [U_1 \otimes \tilde{L}_1, \ldots, U_N \otimes \tilde{L}_N]\text{vec}\left(\text{vec}^{-1}(\mathcal{Y})\right) \\ &= \sum_{i=1}^{N} \text{vec}\left(\tilde{L}_i \text{vec}^{-1}(\mathcal{Y}_{r_i}) U_i^T\right) \\ &= \sum_{i=1}^{N} \sum_{j=1}^{r_i} \text{vec}\left(\tilde{L}_{i,j} Y_{i,j}^T\right), \end{aligned} \tag{44}$$

where $Y_i^T := \text{vec}^{-1}(\mathcal{Y}_{r_i})U_i^T$. Applying (3) again to (44), we obtain

$$U\mathcal{Y} = \sum_{i=1}^{N} \sum_{j=1}^{r_i} (Y_{i,j} \otimes \tilde{L}_{i,j})\text{vec}(1) = \sum_{i=1}^{N} \sum_{j=1}^{r_i} (Y_{i,j} \otimes \tilde{L}_{i,j}). \tag{45}$$

But then, by assumption the r_i sum up to r. Hence, the tensor rank of $U\mathcal{Y}$ is r.

Finally, to prove the assertion (ii), suppose that, for $n \geq 2$, the tensor rank of $\mathbf{b}^n$ is at most $\ell \ll J(P+1)$. Since the tensor rank of $U\mathcal{Y}$ is r, it trivially follows from the previous argument and the definition of $\mathbf{b}^n$ in (26) that (ii) holds with $\ell \geq 1$. □

Remark 1. *Note that, G_0 is just a $(P+1) \times (P+1)$ identity matrix if we work with orthonomal basis polynomials $\{\psi_i\}$. Hence, in this special case, (38) reduces to*

$$\tilde{\mathbf{u}}^n = -\sum_{j=-k}^{k} \frac{2w_j}{\lambda_{\min}} \left(I_{(P+1)} \otimes \exp\left(-\frac{2t_j}{\lambda_{\min}} \tilde{K}_0 \right) \right) [\mathbf{b}^n - U\mathcal{Y}].$$

Remark 2. *The assumption in Theorem 1 that, $\forall n \geq 2$, the tensor rank of the right hand side $\mathbf{b}^n$ is at most ℓ, where $1 \leq \ell \ll J(P+1)$, is justified by the fact that the tensor rank tends to grow as the time step n increases. In practical computations, the tensor rank of $\mathbf{u}^{n-1}$ is truncated with respect to its singular value decay to ensure that the tensor rank of $\mathbf{b}^n$ is kept under control.*

Remark 3. *We note here that Theorem 1 merely provides a theoretical evidence for the existence of low rank approximation to the solution of (25) as $J, P \to \infty$. Thus, we will not rely on these estimates above especially in the iterative solvers that will be discussed in the rest of this paper.*

5 Computing low rank approximations

Although the stochastic Galerkin matrix $\mathcal{A}$ in (27) is block sparse, symmetric and positive definite, it is generally ill-conditioned with respect to stochastic and spatial discretization parameters, e.g. the finite element mesh size, the length N of the random vector ξ, or the total degree of the multivarite stochastic basis polynomials

$\{\psi_i\}$, [21]. Hence, a natural iterative solver for the system is a preconditioned Congugate Gradient (CG) method, [23], [21]. Another iterative solver is a preconditioned Richardson method, [19]. Nevertheless, the choice of an 'appropriate' preconditioner is of utmost concern in this regard. In dealing with steady problems with relatively small σ_κ, many authors use the so-called mean-based preconditioner proposed originally by [13]. Ullmann in [23] points out that the mean-based preconditioner does not take into account all the information contained in $\mathcal{A}$ and thus proposes and analyses an optimal preconditioner based on an approach discussed in [24]. In what follows, we call this the Ullmann preconditioner.

The relative efficiency and optimality of the two preconditioners above notwithstanding, a major issue in solving (25) is evident. More precisely, for each timestep n, one has to solve an enormous elliptic system. Due to the coupled nature of the systems, this exercise can be both computer memory- and time-consuming. To mitigate this problem, we propose to solve (25) with the preconditioners above, together with low rank CG and Richardson methods proposed in [19] in the framework of parameterized steady problems. First, however, we introduce the preconditioners.

5.1 Preconditioning

5.1.1 Mean-based preconditioner

The mean-based preconditioner[1] is given by

$$\mathcal{M}_0 := G_0 \otimes \tilde{K}_0. \tag{46}$$

Now, observe that G_0 is a diagonal matrix due to the orthogonality of the stochastic basis functions $\{\psi_i\}$. Hence, $\mathcal{M}_0$ is a block diagonal matrix. Moreover, by definition, $\tilde{K}_0 = M + \tau K_0$, so that $\tilde{K}_0$ is symmetric and positive definite since M and K_0 are both symmetric and positive definite from (21) and (22). So, $\mathcal{M}_0$ is positive definite and $\mathcal{M}_0^{-1} = G_0^{-1} \otimes \tilde{K}_0^{-1}$, where $G_0^{-1}(j,j) = 1/G_0(j,j) > 0$.

5.1.2 Ullmann preconditioner

This preconditioner is of the form

$$\mathcal{M}_1 := \underbrace{\sum_{i=0}^{N} \frac{\operatorname{trace}(\tilde{K}_i^T \tilde{K}_0)}{\operatorname{trace}(\tilde{K}_0^T \tilde{K}_0)} G_i}_{:=G} \otimes \tilde{K}_0. \tag{47}$$

The Ullmann preconditioner (47) can be thought of as a 'perturbed' version of $\mathcal{M}_0$ since

$$\mathcal{M}_1 = \underbrace{G_0 \otimes \tilde{K}_0}_{:=\mathcal{M}_0} + \sum_{i=1}^{N} \frac{\operatorname{trace}(\tilde{K}_i^T \tilde{K}_0)}{\operatorname{trace}(\tilde{K}_0^T \tilde{K}_0)} G_i \otimes \tilde{K}_0. \tag{48}$$

[1] This is just the matrix $\mathcal{L}$ in Theorem 1.

It is inspired by the first part of the following result obtained by Van Loan and Pitsianis.

Lemma 3. *([24]) Suppose $m = m_1 m_2$, $n = n_1 n_2$, and $X \in \mathbb{R}^{m \times n}$. If $R \in \mathbb{R}^{m_2 \times n_2}$ is fixed, then the matrix $L \in \mathbb{R}^{m_1 \times n_1}$ defined by*

$$L_{i,j} := \frac{\operatorname{trace}(X_{i,j}^T R)}{\operatorname{trace}(R^T R)}, \ i = 1, \dots, m_1, \ j = 1, \dots, n_1, \tag{49}$$

minimizes $||X - L \otimes R||_F$ where $X_{i,j}^T = X((i-1)m_2 + 1 : im_2, (j-1)n_2 + 1 : jn_2)$. Likewise, if $L \in \mathbb{R}^{m_1 \times n_1}$ is fixed, then the matrix $R \in \mathbb{R}^{m_2 \times n_2}$ defined by

$$R_{i,j} := \frac{\operatorname{trace}(\tilde{X}_{i,j}^T L)}{\operatorname{trace}(L^T L)}, \ i = 1, \dots, m_2, \ j = 1, \dots, n_2, \tag{50}$$

minimizes $||X - L \otimes R||_F$ where $\tilde{X}_{i,j}^T = X(i : m_2 : m, j : n_2 : n)$.

Van Loan and Pitsianis further show that the matrices L defined in (49) and R defined in (50) are symmetric and positive definite provided X and R or L, respectively, are symmetric and positive definite.

Now if we set $X = \mathcal{A}$ and $R = \tilde{K}_0$ in (27), it follows from (49) that the matrix G in (47) minimizes $||\mathcal{A} - G \otimes \tilde{K}_0||_F$. More interestingly, $\mathcal{M}_1$ inherits the sparsity pattern, symmetry and positive definiteness of the Galerkin matrix $\mathcal{A}$. Besides, unlike $\mathcal{M}_0$, it makes use of all the information in $\mathcal{A}$. Unfortunately, by reason of its construction, $\mathcal{M}_1$ loses the block diagonal structure enjoyed by $\mathcal{M}_0$ which makes it more expensive to invert than the latter.

Next, we consider a similar preconditioner which combines the advantages of both $\mathcal{M}_0$ and $\mathcal{M}_1$, and but is less expensive to invert than $\mathcal{M}_1$.

5.1.3 A variant of Ullmann preconditioner

Based on (50) in Lemma 3, we set $X = \mathcal{A}$ and fix $L = G_0$ in (27). It turns out that the matrix $\tilde{K}$, defined by

$$\tilde{K} := \sum_{i=0}^{N} \frac{\operatorname{trace}(G_i^T G_0)}{\operatorname{trace}(G_0^T G_0)} \tilde{K}_i, \tag{51}$$

minimizes $||\mathcal{A} - G_0 \otimes \tilde{K}||_F$. Hence, we define the next preconditioner $\mathcal{M}_2$ by

$$\mathcal{M}_2 := G_0 \otimes \tilde{K}. \tag{52}$$

Now, observe that, just as $\mathcal{M}_1$, the new preconditioner $\mathcal{M}_2$ is also a perturbation of $\mathcal{M}_0$ since

$$\begin{aligned} \mathcal{M}_2 &= G_0 \otimes \sum_{i=0}^{N} \frac{\text{trace}(G_i^T G_0)}{\text{trace}(G_0^T G_0)} \tilde{K}_i \\ &= G_0 \otimes \tilde{K}_0 + G_0 \otimes \sum_{i=1}^{N} \frac{\text{trace}(G_i^T G_0)}{\text{trace}(G_0^T G_0)} \tilde{K}_i \\ &= \mathcal{M}_0 + G_0 \otimes \sum_{i=1}^{N} \frac{\text{trace}(G_i^T G_0)}{\text{trace}(G_0^T G_0)} \tilde{K}_i. \qquad (53) \end{aligned}$$

Moreover, from (52) and (51) we see that $\mathcal{M}_2$ is block diagonal. Since $\mathcal{A}$ and G_0 are both symmetric and positive definite, we also know from the work of Van Loan and Pitsianis above that $\mathcal{M}_2$ is symmetric and positive definite. In terms of implementation, it turns out that $\mathcal{M}_2$ is fairly easy to implement because of its block diagonal structure since

$$\mathcal{M}_2^{-1}\mathbf{x} = (G_0 \otimes \tilde{K})^{-1}\mathbf{x} = (G_0^{-1} \otimes \tilde{K}^{-1})\mathbf{x},$$

for any vector $\mathbf{x}$ of appropriate dimension. But then, G_0^{-1} is just a diagonal matrix from (19), and

$$\begin{aligned} \text{trace}(G_0^T G_0) &= \sum_{j=1}^{P+1} G_0(j,j)^2, \\ \text{trace}(G_i^T G_0) &= \sum_{j=1}^{P+1} G_i(j,j) G_0(j,j), \ i = 1, \ldots, N, \end{aligned}$$

from (51). So, the major task here is to invert $\tilde{K}$ just as it is to invert $\tilde{K}_0$ in $\mathcal{M}_0$. We approximate the inverses $\tilde{K}^{-1}$ and $\tilde{K}_0^{-1}$ using an algebraic multigrid V-cycle solver in our computations.

Remark 4. *As pointed out in [23], if the probability density ρ of the random vector ξ is even, that is, $\rho(\xi) = \rho(-\xi)$, as in the case of Gaussian and uniform densities, then the eigenvalues of the stochastic matrices $G_i, i = 1, \ldots, N$, are symmetric about the origin. Thus,* $\text{trace}(G_i^T G_0) = 0$, *and $\mathcal{M}_2$ reduces to $\mathcal{M}_0$.*

5.2 Preconditioned iterative solvers

Having presented the preconditioners, we proceed in this section to discuss the low rank preconditioned Congugate Gradient (LRPCG) method and the low rank preconditioned Richardson (LRPR) method, [19]. The basic idea behind LRPCG and LRPR is that the iterates in the algorithms are truncated based on the decay of their singular values. Thus, at each iteration, the iterates are put in low rank format (cf. (25)). The truncation, no doubt, introduces further error in the solution. However, the truncation tolerance can be so tightened that the error becomes negligible. More importantly, the computer memory required to store the matrices is reduced and the computational time is thus enhanced.

First, we present LRPCG in Algorithm 1. We point out a few things regarding the implementation of LRPCG with respect to the solution of (25). Note that, in Algorithm 1, all vectors in $\mathbb{R}^{J\cdot(P+1)}$ (cf. (18)) are reshaped into $\mathbb{R}^{J\times(P+1)}$ matrices by the vec^{-1} operator. Now, recall that for each fixed time step $n = 1, 2, \ldots, T_{\max}$, we need to solve an elliptic system using the LRPCG algorithm. In particular, for each solve, we need to evaluate $\mathcal{A}(X)$, where $X := \mathbf{u}_k^n$ or P_k. For this purpose, we set

Algorithm 1 Low Rank Preconditioned Congugate Gradient Method

```
Input: Matrix functions $\mathcal{A}, \mathcal{M} : \mathbb{R}^{J\times(P+1)} \to \mathbb{R}^{J\times(P+1)}$, right hand side $B^n \in \mathbb{R}^{J\times(P+1)}$ in low rank format. Truncation operator $\mathcal{T}$ w.r.t relative accuracy $\varepsilon_{rel}$.
Output: Matrix $\mathbf{u}^n \in \mathbb{R}^{J\times(P+1)}$ fulfulling $||\mathcal{A}(\mathbf{u}^n) - B^n||_F \leq \text{tol}$.
$\mathbf{u}_0^n = 0,\ R_0 = B^n,\ Z_0 = \mathcal{M}^{-1}(R_0),\ P_0 = Z_0, Q_0 = \mathcal{A}(P_0),$
$\vartheta_0 = \langle P_0, Q_0 \rangle,\ k = 0.$
  while $||R_k||_F > \text{tol}$ do
      $\omega_k = \langle R_k, P_k \rangle / \vartheta_k$
      $\mathbf{u}_{k+1}^n = \mathbf{u}_k^n + \omega_k P_k,$          $\mathbf{u}_{k+1}^n \leftarrow \mathcal{T}(\mathbf{u}_{k+1}^n)$
      $R_{k+1} = B^n - \mathcal{A}(\mathbf{u}_{k+1}^n),$      $Optionally : R_{k+1} \leftarrow \mathcal{T}(R_{k+1})$
      $Z_{k+1} = \mathcal{M}^{-1}(R_{k+1})$
      $\beta_{k+1} = -\langle Z_{k+1}, Q_k \rangle / \vartheta_k$
      $P_{k+1} = Z_{k+1} + \beta_k P_k,$          $P_{k+1} \leftarrow \mathcal{T}(P_{k+1})$
      $Q_{k+1} = \mathcal{A}(P_{k+1}),$           $Optionally : Q_{k+1} \leftarrow \mathcal{T}(Q_{k+1})$
      $\vartheta_{k+1} = \langle P_k, Q_k \rangle$
      $k = k + 1$
  end while
  $\mathbf{u}^n = \mathbf{u}_k^n$
```

$$\mathcal{A}\text{vec}(X) = \left(\sum_{i=0}^{N} G_i \otimes \tilde{K}_i \right) \text{vec}(X), \tag{54}$$

where $X \in \mathbb{R}^{J\times(P+1)}$ is of low rank, say, k :

$$\begin{array}{rcl} X & = & UV^T,\ U \in \mathbb{R}^{J\times k},\ V \in \mathbb{R}^{(P+1)\times k},\ k \ll J, P, \\ U & = & [u_1, \ldots, u_k],\ V \in [v_1, \ldots, v_k], \end{array}$$

so that, using (3), one gets

$$\begin{aligned}\mathrm{vec}(X) &= \mathrm{vec}\left(\sum_{i=1}^{k} u_j v_j^T\right) \\ &= \sum_{j=1}^{k} \mathrm{vec}(u_j v_j^T) \\ &= \sum_{j=1}^{k} v_j \otimes u_j. \end{aligned} \tag{55}$$

Hence, we have

$$\begin{aligned}\mathcal{A}\mathrm{vec}(X) &= \left(\sum_{i=0}^{N} G_i \otimes \tilde{K}_i\right) \mathrm{vec}(X) \\ &= \left(\sum_{i=0}^{N} G_i \otimes \tilde{K}_i\right)\left(\sum_{j=1}^{k} v_j \otimes u_j\right) \\ &= \sum_{i=0}^{N}\sum_{j=1}^{k} (G_i v_j) \otimes (\tilde{K}_i u_j) \in \mathbb{R}^{J\cdot(P+1)\times 1}, \end{aligned} \tag{56}$$

and we then have to reshape (56) to have

$$\mathcal{A}(X) := \mathrm{vec}^{-1}(\mathcal{A}\mathrm{vec}(X)) \in \mathbb{R}^{J\times(P+1)}. \tag{57}$$

Moreover, in order to apply any of the three preconditioners to the residual matrices R_k, that is, $\mathcal{M}^{-1}(R_k)$, we have to ensure that R_k are in low rank format as in (55), so we can obtain similar expressions as in (56) and (57), since $\mathcal{M}^{-1} := \mathcal{M}_i^{-1}$, $i = 0, 1, 2$, have the same size and Kronecker product structure as $\mathcal{A}$. The right hand side of (25), that is, $\mathbf{b}^n = (G_0 \otimes M)\,\mathbf{u}^{n-1} + \tau\,(\mathbf{g}_0 \otimes \mathbf{f}_0)$ is also reshaped such that $B^n := \mathrm{vec}^{-1}(\mathbf{b}^n) \in \mathbb{R}^{J\times(P+1)}$. Finally, the iterates $\mathbf{u}_k^n$ are truncated in every iteration by the trucation operator $\mathcal{T}$ based on the decay of their singular values.

Next, we present the LRPR in Algorithm 2. The implementation issues in Algorithm 2 are handled as discussed above in the case of LRPCG. One point is noteworthy here, though. Since $\mathcal{A}$ and $\mathcal{M}$ are symmetric and positive definite, the parameter α is chosen as

$$\alpha = \frac{2}{\lambda_{\min}(\mathcal{M}^{-1}\mathcal{A}) + \lambda_{\max}(\mathcal{M}^{-1}\mathcal{A})} \tag{58}$$

to ensure the best convergence rate, see e.g., [17]. For large linear systems, the eigenvalues in (58) can be quite expensive to compute.

Having discussed the two low rank solvers, we proceed to the next section to investigate the performance of these solvers in conjuction with the preconditioners.

Algorithm 2 Low Rank Preconditioned Richardson Method

```
Input: Matrix functions $\mathcal{A}, \mathcal{M} : \mathbb{R}^{J\times(P+1)} \to \mathbb{R}^{J\times(P+1)}$, right hand side $B^n \in \mathbb{R}^{J\times(P+1)}$ in low rank format. Parameter $\alpha > 0$, truncation operator $\mathcal{T}$ w.r.t relative accuracy $\varepsilon_{rel}$.
Output: Matrix $\mathbf{u}^n \in \mathbb{R}^{L\times(P+1)}$ fulfulling $||\mathcal{A}(\mathbf{u}^n) - B^n||_F \leq$ tol.
$\mathbf{u}_0^n = 0,\ R_0 = B^n,\ k = 0.$
  while $||R_k||_F >$ tol do
    $\mathbf{u}_{k+1}^n = \mathbf{u}_k^n + \alpha\mathcal{M}^{-1}(R_{k+1}),$        $\mathbf{u}_{k+1}^n \leftarrow \mathcal{T}(\mathbf{u}_{k+1}^n)$
    $R_{k+1} = B^n - \mathcal{A}(\mathbf{u}_{k+1}^n)$
    $k = k + 1$
  end while
  $\mathbf{u}^n = \mathbf{u}_k^n$
```

6 Numerical experiments

To demonstrate the performance of the approach presented in this paper, we consider the 1D version of our model problem (5) which was studied in [7]. More precisely, we choose $f = 1$ and $\mathcal{D} = (-a, a)$, where $a = 1$. The random input κ is characterized by

$$\bar{\kappa} = 10, \quad C_\kappa(x, y) = \sigma_\kappa \exp\left(-\frac{|x-y|}{\ell}\right), \quad \forall x, y \in \mathcal{D}.$$

The eigenpairs (λ_j, φ_j) of the KL expansion of κ are given explicitly in [14]:

$$\begin{aligned} \varphi_{2j}(x) &= \frac{\cos(\omega_{2j}x)}{\sqrt{a + \frac{\sin(2a\omega_{2j})}{2\omega_{2j}}}}, & \varphi_{2j-1}(x) &= \frac{\sin(\omega_{2j-1}x)}{\sqrt{a + \frac{\sin(2a\omega_{2j-1})}{2\omega_{2j-1}}}}, \quad j \in \mathbb{N}, \\ \lambda_{2j} &= \frac{2\ell}{1 + \ell^2\omega_{2j}^2}, & \lambda_{2j-1} &= \frac{2\ell}{1 + \ell^2\omega_{2j-1}^2}, \quad j \in \mathbb{N}. \end{aligned}$$

Here, ω_{2j} and ω_{2j-1}, respectively solve

$$\begin{aligned} \frac{1}{\ell} - \omega_{2j}\tan(\omega_{2j}) &= 0, \quad j \in \mathbb{N}, \\ \omega_{2j-1} + \frac{1}{\ell}\tan(\omega_{2j-1}) &= 0, \quad j \in \mathbb{N}. \end{aligned}$$

In the simulations, we set $\ell = 1$ and investigate the behavior of the solvers for different values of the discretization parameters N, Q, σ_κ. Moreover, we choose $\xi = \{\xi_1, \ldots, \xi_N\}$ such that

(i) $\xi_j \sim \mathcal{U}[-1, 1]$, and N-dimensional Legendre polynomials with support in $[-1, 1]^N$. Note then that this choice yields $\mathcal{M}_2 = \mathcal{M}_0$. Hence, we will compare the iterative solvers with respect to only the mean-based preconditioner $\mathcal{M}_0$ and the Ullmann preconditioner $\mathcal{M}_1$ in this particular example.

(ii) $\xi_j \sim \mathrm{Beta}(\alpha, \beta)$, with $1 < \alpha < \beta$, so that density of ξ is not symmetric but positively skewed and we can use Jacobi polynomials with support in $[-1, 1]^N$. In this case, we can conveniently appeal to the three preconditioners $\mathcal{M}_0$, $\mathcal{M}_1$ and $\mathcal{M}_2$ in our comparative analyses.

The numerical experiments were performed on a Linux machine with 80 GB RAM using MATLAB® 7.14 together with a MATLAB® version of the AMG code HSL MI20, [5]. In both examples (i) and (ii), the resulting linear systems were solved for time $T = 1$; the stopping criterion for both CG and Richardson methods was 10^{-5} and relative tolerance for the truncation operator in both cases was 10^{-8}.

Timesteps=143	LR$\mathcal{M}_0$	LR$\mathcal{M}_1$	$\mathcal{M}_0$	$\mathcal{M}_1$
	LRPCG(LRPR)	LRPCG(LRPR)	PCG	PCG
Par=(5,3,1)				
Total iterations	702 (1110)	695 (1076)	345	572
Total CPU time	207.6 (180.2)	205.7 (181.4)	1155	1904
Par=(5,3,0.1)				
Total iterations	553 (709)	559 (708)	286	286
Total CPU time	232.3 (110.9)	237.4 (112.9)	949.7	956.9
Par=(6,4,0.1)				
Total iterations	553 (710)	559 (709)	286	429
Total CPU time	1193.9 (674.8)	1222.4 (685.2)	1762.6	2593.9
Par=(6,4,1)				
Total iterations	702 (1162)	694 (1110)	349	572
Total CPU time	1612.3 (1131.8)	1553.2 (1052.4)	15043	24208

Table 1: Outputs of simulations showing total CPU times and total iterations from preconditioned low rank solvers (second and third columns) compared with those from plain preconditioned CG (last two columns) for selected parameter values using model (i).

Tables 1 and 2 show the simulation results for examples (i) and (ii) repectively. Here, the 1D unsteady diffusion equation is solved using low rank preconditioned CG and Richardson algorithms, as well as using the plain preconditioned CG method. In all the simulations, the total number of iterations and total CPU times[2] as reported in the table are used as benchmarks to compare the performance of the solution approaches. We used the tuple of parameters (N, Q, σ_κ). Thus, in $(5, 3, 1)$ and $(5, 3, 0.1)$ we have $P = 56$ (cf. (13)), whereas $(6, 4, 0.1)$ and $(6, 4, 1)$ give $P = 210$.

In the second and third columns of Table 1 are the outputs from the low rank approach while the last two are without the low rank truncations (using just the MATLAB command 'pcg'). Also in the second and third columns, the outputs in parentheses are from the Richardson method, whereas adjacent to them are those from the CG method.

[2] In our numerical experiments, we noticed that the solvers were rubost with respect to the step size τ.

Observe that the low rank approach generally takes less CPU time than the plain approach, although the plain approach does a little better on average in terms of the iteration counts. The performance of the low rank approach is particularly pronounced as P increases to 210 (that is, with $(6, 4, 1)$) in which case the CPU time is reduced by more than 9 times using $\mathcal{M}_0$ and 15 times using $\mathcal{M}_1$.

Timesteps=143	LR$\mathcal{M}_0$	LR$\mathcal{M}_1$	LR$\mathcal{M}_2$
	LRPCG(LRPR)	LRPCG(LRPR)	LRPCG(LRPR)
Par=(4,3,1)			
Total iterations	709 (1006)	704 (977)	705 (963)
Total CPU time	133.1 (77.1)	129.0 (78.6)	134.2 (75.6)
Par=(4,3,0.05)			
Total iterations	557 (703)	556 (702)	555 (702)
Total CPU time	105.7 (54.9)	103.5 (55.4)	102.3 (52.7)
Par=(4,3,0.5)			
Total iterations	645 (848)	605 (839)	571(832)
Total CPU time	194.4 (93.3)	178.8 (90.7)	176.4 (77.6)

Timesteps=143	$\mathcal{M}_0$	$\mathcal{M}_1$	$\mathcal{M}_2$
	PCG	PCG	PCG
Par=(4,3,1)			
Total iterations	321	572	320
Total CPU time	412.0	723	400.7
Par=(4,3,0.05)			
Total iterations	286	286	286
Total CPU time	354.2	333.8	366.4
Par=(4,3,0.5)			
Total iterations	294	429	294
Total CPU time	315.8	465.4	362.5

Table 2: Outputs of simulations showing total CPU times and total iterations from preconditioned low rank solvers (first four rows) compared with those from plain preconditioned CG (last four rows) for selected parameter values using model (ii).

In Table 2, we include the preconditioner $\mathcal{M}_2$ in the comparison. As in Table 1, we also observe here that the low rank approach is less time-consuming than the plain approach regardless of the preconditioner used. Particularly noteworthy is that, with the low-rank solvers, on average, the new preconditioner $\mathcal{M}_2$ competes favourably in terms of CPU time and iterations relative to both $\mathcal{M}_0$ and $\mathcal{M}_1$; however, with the plain preconditioned CG, it performs relatively better than $\mathcal{M}_1$ especially when $\sigma_\kappa \geq 0.5$.

Finally, in both considered examples, we note that low rank preconditioned Richardson outperforms low rank preconditioned CG in terms of CPU time, but takes more iterations. This is not suprising because the latter does at least two low rank truncations unlike the former which does only one.

7 Conclusions and outlook

The use of SGFEM in discretizing linear PDEs with uncertain inputs is standard in the literature. For it to compete favourably with other approaches like Monte Carlo and stochastic collocation methods in solving time-dependent problems, efficient solvers with appropriate preconditioners have to be developed to solve the resulting large dimensional coupled linear system. In this paper, we have solved the linear systems (25) using low rank iterative solvers, together with three different preconditioners. In general, the combination of each of the preconditioners and the iterative solvers seems quite promising as it reduces the CPU time and computer memory required to solve the linear system. Although the low rank approach introduces further error in the simulation due to the low rank truncations, the relative tolerance of the truncation operator can be so tightened that the error will become negligible while computational time is greatly reduced. In the future, we plan to apply the low rank approach to unsteady (Navier)-Stokes problems.

Acknowledgement

The authors would like to thank Tobias Breiten and Piotr Skrzypacz for fruitful discussions in the course of writing this paper.

References

[1] I. Babuška and P. Chatzipantelidis, *On solving linear elliptic stochastic partial differential equations*, Computer Methods in Applied Mechanics and Engineering, 191 (2002), pp. 4093–4122.

[2] I. Babuška, R. Tempone, and G. Zouraris, *Galerkin finite element approximations of stochastic elliptic partial differential equations*, SIAM Journal of Numerical Analysis, 42 (2004), pp. 800–825.

[3] N. F. Babuška, I. and R. Tempone, *A stochastic collocation method for elliptic partial differential equations with random input data*, SIAM Journal of Numerical Analysis, 45 (2007), pp. 1005–1034.

[4] P. Benner and T. Breiten, *Low rank methods for a class of generalized Lyapunov equations and related issues*, Numerische Mathematik, 124 (2013), pp. 441–470.

[5] J. Boyle, M. D. Mihajlovic, and J. A. Scott, *HSL MI20: an efficient AMG preconditioner*, Tech. Rep. RAL-TR-2007-021, Rutherford Appleton Laboratory (CCLRC), 2007.

[6] K. A. Cliffe, M. B. Giles, R. Scheichl, and A. L. Teckentrup, *Multilevel Monte Carlo methods and applications to elliptic PDEs with random coefficients*, Computing and Visualization in Science, 14 (2011), pp. 3–15.

[7] P. Constantine, *A primer on stochastic Galerkin methods*, Working Paper, (2007).

[8] T. Damm, *Direct methods and ADI-preconditioned Krylov subspace methods for generalized Lyapunov equations*, Numerical Linear Algebra and Applications, 15 (2008), pp. 853–871.

[9] H. Elman and D. Furnival, *Solving steady-state diffusion problem using multigrid*, IMA Journal of Numerical Analysis, 27 (2007), pp. 675–688.

[10] O. G. Ernst, A. Mugler, H.-J. Starkloff, and E. Ullmann, *On the convergence of generalized polynomial chaos expansions*, ESAIM: Mathematical Modelling and Numerical Analysis, 46 (2012), pp. 317 – 339.

[11] O. G. Ernst and E. Ullmann, *Stochastic Galerkin matrices*, SIAM Journal on Matix Analysis and Applications, 31 (2010), pp. 1848–1872.

[12] P. Frauenfelder, C. Schwab, and R. A. Todor, *Finite elements for elliptic problems with stochastic coefficients*, Computer Methods in Applied Mechanics and Engineering, 194 (2005), pp. 205–228.

[13] R. G. Ghanem and R. M. Kruger, *Numerical solution of spectral stochastic finite element systems*, Computer Methods in Applied Mechanics and Engineering, 129 (2005), pp. 289–303.

[14] R. G. Ghanem and P. Spanos, *Stochastic finite elements: A spectral approach*, Springer-Verlag, New York, 1996.

[15] G. H. Golub and C. H. Van Loan, *Matrix computations*, Johns Hopkins University Press, 1996.

[16] L. Grasedyck, *Existence and computation of low Kronecker-rank approximations for large linear systems of tensor product structure*, Computing, 72 (2004), pp. 247–265.

[17] B. N. Khoromskij and C. Schwab, *Tensor-structured Galerkin approximation of parametric and stochastic elliptic PDEs*, SIAM Journal of Scientific Computing,, 33 (2011), pp. 364–385.

[18] D. Kressner and C. Tobler, *Krylov subspace methods for linear systems with tensor product structure*, SIAM Journal on Matix Analysis and Applications, 31 (2010), pp. 1688–1714.

[19] ——, *Low-rank tensor Krylov subspace methods for parametrized linear systems*, SIAM Journal on Matix Analysis and Applications, 32 (2011), pp. 1288–1316.

[20] O. P. Le Maitre, O. M. Knio, B. J. Debusschere, H. N. Najm, and R. G. Ghanem, *A multigrid solver for two-dimensional stochastic diffusion equations*, Computer Methods in Applied Mechanics and Engineering, 192 (2003), pp. 4723 – 4744.

[21] C. E. Powell and H. C. Elman, *Block-diagonal preconditioning for spectral stochastic finite-element systems*, IMA Journal of Numerical Analysis, 29 (2009), pp. 350–375.

[22] E. Rosseel, T. Boonen, and S. Vandewalle, *Algebraic multigrid for stationary and time-dependent partial differential equations with stochastic coefficients*, Numerical Linear Algebra and Applications, 15 (2008), pp. 141–163.

[23] E. Ullmann, *A Kronecker product preconditioner for stochastic Galerkin finite element discretizations*, SIAM Journal on Scientific Computing, 32 (2010), pp. 923–946.

[24] C. F. Van Loan and N. Pitsianis, *Approximation with Kronecker products*, Kluwer Publications, Dordrecht, 1992, ch. 4, pp. 293–314.

[25] D. Xiu and G. E. Karniadakis, *A new stochastic approach to transient heat conduction modeling with uncertainty*, International Journal of Heat & Mass Transfer, 46 (2003), pp. 4681–4693.

[26] D. Xiu and J. Shen, *Efficient stochastic Galerkin methods for random diffusion*, Journal of Computational Physics, 228 (2009), pp. 266–281.