SIAM/ASA J. UNCERTAINTY QUANTIFICATION
Vol. 7, No. 4, pp. 1275–1300

A Low-Rank Solver for the Navier–Stokes Equations with Uncertain Viscosity[*]

Kookjin Lee[†], Howard C. Elman[‡], and Bedřich Sousedík[§]

Abstract. We study an iterative low-rank approximation method for the solution of the steady-state stochastic Navier–Stokes equations with uncertain viscosity. The method is based on linearization schemes using Picard and Newton iterations and stochastic finite element discretizations of the linearized problems. For computing the low-rank approximate solution, we adapt the nonlinear iterations to an inexact and low-rank variant, where the solution of the linear system at each nonlinear step is approximated by a quantity of low rank. This is achieved by using a tensor variant of the GMRES method as a solver for the linear systems. We explore the inexact low-rank nonlinear iteration with a set of benchmark problems, using a model of flow over an obstacle, under various configurations characterizing the statistical features of the uncertain viscosity, and we demonstrate its effectiveness by extensive numerical experiments.

Key words. stochastic Galerkin method, Navier–Stokes equations, low-rank approximation

AMS subject classifications. 35R60, 60H15, 65F10

DOI. 10.1137/17M1151912

1. Introduction.

We are interested in the efficient computation of solutions of the steady-state Navier–Stokes equations with uncertain viscosity. Such uncertainty may arise from measurement error or uncertain ratios of multiple phases in porous media. The uncertain viscosity can be modeled as a positive random field parameterized by a set of random variables [23, 28, 31], and, consequently, the solution of the stochastic Navier–Stokes equations also can be modeled as a random vector field depending on the parameters associated with the viscosity (i.e., a function of the same set of random variables). As a solution method, we consider the stochastic Galerkin method [1, 12] combined with the generalized polynomial chaos (gPC) expansion [34], which provides a spectral approximation of the solution function. The stochastic Galerkin method results in a coupled algebraic system of equations. There has been considerable progress in the development of solvers for these systems [7, 22, 24, 29, 32], although costs may be high when the global system becomes large.

[*]Received by the editors October 17, 2017; accepted for publication (in revised form) May 20, 2019; published electronically October 31, 2019.
https://doi.org/10.1137/17M1151912
Funding: The work of the authors was supported by the U.S. Department of Energy Office of Advanced Scientific Computing Research, Applied Mathematics program under award DEC-SC0009301 and by the National Science Foundation under grants DMS-1819115 and DMS-1521563.
[†]Department of Computer Science at the University of Maryland, College Park, MD 20742, and Extreme-scale Data Science and Analytics Department, Sandia National Laboratories, Livermore, CA 94550 (koolee@sandia.gov).
[‡]Department of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742 (elman@cs.umd.edu).
[§]Department of Mathematics and Statistics, University of Maryland, Baltimore County, MD 21250 (sousedik@umbc.edu).

One way to address this issue is through use of tensor *Krylov subspace* methods, which operate in tensor format and reduce the costs of matrix operations by exploiting a Kronecker-product structure of system matrices. Variants of this approach have been developed for the Richardson iteration [14, 18], the conjugate gradient and the BiCGstab methods [14], the minimum residual method [30], and the generalized minimum residual (GMRES) method [2]. Efficiencies are also obtained from the fact that solutions can often be well approximated by low-rank objects. These ideas have been shown to reduce costs for solving steady [6, 11, 16, 18] and unsteady stochastic diffusion equations [4].

In this study, we adapt the low-rank approximation scheme to a solver for the systems of nonlinear equations obtained from the stochastic Galerkin discretization of the stochastic Navier–Stokes equations. In particular, we consider a low-rank variant of linearization schemes based on Picard and Newton iteration, where the solution of the nonlinear system is computed by solving a sequence of linearized systems using a low-rank variant of the GMRES method (lrGMRES) [2] in combination with inexact nonlinear iteration [5].

We base our development of the stochastic Galerkin formulation of the stochastic Navier–Stokes equations on ideas from [23, 28]. In particular, we consider a random viscosity affinely dependent on a set of random variables as suggested in [23] (and in [28], which considers a gPC approximation of the lognormally distributed viscosity). The stochastic Galerkin formulation of the stochastic Navier–Stokes equations is also considered in [3], which studies an optimal control problem constrained by the stochastic Navier–Stokes problem and computes an approximate solution using a low-rank tensor-train decomposition [21]. Related work [31] extends a proper generalized decomposition method [20] for the stochastic Navier–Stokes equations, where a low-rank approximate solution is built from successively computing rank-one approximations. See the book [15] for an overview and other spectral approximation approaches for models of computational fluid dynamics.

An outline of the paper is as follows. In section 2, we review the stochastic Navier–Stokes equations and their discrete Galerkin formulations. In section 3, we present an iterative low-rank approximation method for solutions of the discretized stochastic Navier–Stokes problems. In section 4, we introduce an efficient variant of the inexact Newton method, which solves linear systems arising in nonlinear iteration using low-rank format. We follow a hybrid approach, which employs several steps of Picard iteration followed by Newton iteration. In section 5, we examine the performance of the proposed method on a set of benchmark problems that model the flow over an obstacle. Finally, in section 6, we draw some conclusions.

2. Stochastic Navier–Stokes equations.

Consider the stochastic Navier–Stokes equations: Find velocity $\vec{u}(x,\xi)$ and pressure $p(x,\xi)$ such that

$$\begin{aligned} -\nabla\cdot\nu(x,\xi)\nabla\vec{u}(x,\xi) + (\vec{u}(x,\xi)\cdot\nabla)\vec{u}(x,\xi) + \nabla p(x,\xi) &= \vec{f}(x,\xi), \\ \nabla\cdot\vec{u}(x,\xi) &= 0 \end{aligned} \tag{1}$$

in $D\times\Gamma$, with boundary conditions

$$\begin{aligned} \vec{u}(x,\xi) &= \vec{g}(x,\xi) && \text{on } \partial D_{\text{Dir}}\times\Gamma, \\ \nu(x,\xi)\nabla\vec{u}(x,\xi)\cdot\vec{n} - p(x,\xi)\vec{n} &= \vec{0} && \text{on } \partial D_{\text{Neu}}\times\Gamma, \end{aligned}$$

where $\partial D = \partial D_{\text{Dir}} \cup \partial D_{\text{Neu}}$. The stochasticity of (1) stems from the random viscosity

$\nu(x,\xi)$, which is modeled as a positive random field parameterized by a set of independent and identically distributed random variables $\xi = \{\xi_1, \ldots, \xi_{n_\nu}\}$. The random variables comprising ξ are defined on a probability space $(\Omega, \mathcal{F}, P)$ such that $\xi : \Omega \to \Gamma \subset \mathbb{R}^{n_\nu}$, where Ω is a sample space, $\mathcal{F}$ is a σ-algebra on Ω, P is a probability measure on Ω, and $\Gamma = \Gamma_1 \times \cdots \times \Gamma_{n_\nu}$ is the joint image of ξ. For each i, Γ_i is a finite interval symmetric about the origin. The joint probability density function (PDF) of ξ is denoted by $\rho(\xi) = \prod_{i=1}^{n_\nu} \rho_i(\xi_i)$, where $\rho_i(\xi_i)$ is assumed to be symmetric about 0, i.e., $\rho_i(-\delta) = \rho_i(\delta)$ for all $\delta \in \Gamma_i$. The expected value of a random function $v(\xi)$ on Γ is then $\langle v \rangle_\rho = \mathbb{E}[v] \equiv \int_\Gamma v(\xi)\rho(\xi)d\xi$.

For the random viscosity, we consider a random field that has affine dependence on the random variables ξ,

$$\nu(x,\xi) \equiv \nu_0 + \sigma_\nu \sum_{k=1}^{n_\nu} \nu_k(x)\xi_k, \tag{2}$$

where $\{\nu_0, \sigma_\nu^2\}$ are parameters characterizing the statistical properties of the random field $\nu(x,\xi)$. The random viscosity leads to the random Reynolds number

$$\mathrm{Re}(\xi) \equiv \frac{UL}{\nu(\xi)}, \tag{3}$$

where U is the characteristic velocity and L is the characteristic length. We denote the Reynolds number associated with ν_0 by $\mathrm{Re}_0 = \frac{UL}{\nu_0}$. In this study, we ensure that the viscosity (2) has positive values by controlling $\{\nu_0, \sigma_\nu^2\}$ and only consider small enough Re_0 so that the flow problem has a unique solution.

2.1. Stochastic Galerkin method. In the stochastic Galerkin method, a mixed variational formulation of (1) can be obtained by employing Galerkin orthogonality: Find $(\vec{u}, p) \in (V_E, Q_D) \otimes L^2(\Gamma)$ such that

$$\left\langle \int_D \nu \nabla\vec{u} : \nabla\vec{v} + [(\vec{u}\cdot\nabla)\,\vec{u}]\cdot\vec{v} - p(\nabla\cdot\vec{v}) \right\rangle_\rho = \left\langle \int_D \vec{f}\cdot\vec{v} \right\rangle_\rho \qquad \forall \vec{v} \in V_D \otimes L^2(\Gamma), \tag{4}$$

$$\left\langle \int_D q(\nabla\cdot\vec{u}) \right\rangle_\rho = 0 \qquad \forall q \in Q_D \otimes L^2(\Gamma). \tag{5}$$

The velocity solution and test spaces are $V_E = \{\vec{u} \in \mathcal{H}^1(D)^2 | \vec{u} = \vec{g} \text{ on } \partial D_{\mathrm{Dir}}\}$ and $V_D = \{\vec{v} \in \mathcal{H}^1(D)^2 | \vec{v} = \vec{0} \text{ on } \partial D_{\mathrm{Dir}}\}$, where $\mathcal{H}^1(D)$ refers to the Sobolev space of functions with derivatives in $L^2(D)$, for the pressure solution, $Q_D = L^2(D)$, and $L^2(\Gamma)$ is a Hilbert space equipped with the inner product

$$\langle u, v \rangle_\rho \equiv \int_\Gamma u(\xi)v(\xi)\rho(\xi)d\xi.$$

The solution of the variational formulation (4)–(5) satisfies

$$\mathcal{R}(\vec{u}, p; \vec{v}, q) = 0 \qquad \forall \vec{v} \in V_D \otimes L^2(\Gamma),\ \forall q \in Q_D \otimes L^2(\Gamma), \tag{6}$$

where $\mathcal{R}(\vec{u}, p; \vec{v}, q)$ is a nonlinear residual

$$\mathcal{R}(\vec{u}, p; \vec{v}, q) \equiv \begin{bmatrix} \langle \int_D \vec{f}\cdot\vec{v} - \nu\nabla\vec{u} : \nabla\vec{v} - [(\vec{u}\cdot\nabla)\,\vec{u}]\cdot\vec{v} + \int_D p(\nabla\cdot\vec{v}) \rangle_\rho \\ \langle -\int_D q(\nabla\cdot\vec{u}) \rangle_\rho \end{bmatrix}. \tag{7}$$

To compute the solution of the nonlinear equation (6), we employ linearization techniques based on either Picard iteration or Newton iteration [9]. Replacing $(\vec{u}, p)$ of (4)–(5) with $(\vec{u}+\delta\vec{u}, p+\delta p)$ and neglecting the quadratic term $c(\delta\vec{u}; \delta\vec{u}, \vec{v})$, where $c(\vec{z}; \vec{u}, \vec{v}) \equiv \int_D [(\vec{z}\cdot\nabla)\vec{u}]\cdot\vec{v}$, gives

$$\begin{bmatrix} \langle \int_D \nu\nabla\delta\vec{u} : \nabla\vec{v} + c(\delta\vec{u}; \vec{u}, \vec{v}) + c(\vec{u}; \delta\vec{u}, \vec{v}) - \int_D \delta p(\nabla\cdot\vec{v})\rangle_\rho \\ \langle \int_D q(\nabla\cdot\delta\vec{u})\rangle_\rho \end{bmatrix} = \mathcal{R}(\vec{u}, p; \vec{v}, q). \tag{8}$$

In the Newton iteration, the $(n+1)$st iterate $(\vec{u}^{n+1}, p^{n+1})$ is computed by taking $\vec{u} = \vec{u}^n$, $p = p^n$ in (8), solving (8) for $(\delta\vec{u}^n, \delta p^n)$, and updating

$$\vec{u}^{n+1} := \vec{u}^n + \delta\vec{u}^n, \quad p^{n+1} := p^n + \delta p^n.$$

In the Picard iteration, the term $c(\delta\vec{u}; \vec{u}, \vec{v})$ is omitted from the linearized form (8).

2.2. Discrete stochastic Galerkin system.

To obtain a discrete system, the velocity $\vec{u}(x, \xi)$ and the pressure $p(x, \xi)$ are approximated by a generalized polynomial chaos expansion [34]:

$$\vec{u}(x,\xi) \approx \sum_{i=1}^{n_\xi} \vec{u}_i(x)\psi_i(\xi), \quad p(x,\xi) \approx \sum_{i=1}^{n_\xi} p_i(x)\psi_i(\xi), \tag{9}$$

where $\{\psi_i(\xi)\}_{i=1}^{n_\xi}$ is a set of n_ν-variate orthogonal polynomials (i.e., $\langle\psi_i\psi_j\rangle_\rho = 0$ if $i \neq j$) consisting of products of univariate orthogonal polynomials $\psi_i(\xi) = \prod_{j=1}^{n_\nu} \ell_{d_j(i)}(\xi_j)$, where $d(i) = (d_1(i), \ldots, d_{n_\nu}(i))$ is a multi-index consisting of nonnegative integers and $\ell_{d_j(i)}$ is the $d_j(i)$th-order polynomial of ξ_j. In this study, we set the total degree space, $\Lambda_{n_\nu, d_{\text{tot}}} = \{d(i) \in \mathbb{N}_0^{n_\nu} : \|d(i)\|_1 \leq d_{\text{tot}}\}$, where $\mathbb{N}_0$ is the set of nonnegative integers, $\|d(i)\|_1 = \sum_{k=1}^{n_\nu} d_k(i)$, and d_{tot} defines the maximal degree of $\{\psi_i(\xi)\}_{i=1}^{n_\xi}$. For ordering the index set $\{d(i)\}_{i=1}^{n_\xi}$, we consider the graded lexicographical ordering [33, section 5.2], where $i > j$ if and only if $\|d(i)\|_1 > \|d(j)\|_1$ or $\|d(i)\|_1 = \|d(j)\|_1$ and the rightmost nonzero entry of $d(i) - d(j)$ is positive (e.g., $\Lambda_{3,2} = \{(0,0,0), (1,0,0), (0,1,0), (0,0,1), (2,0,0), (1,1,0), (0,2,0), (0,1,1), (0,0,2)\}$). This set of orthogonal polynomials gives rise to a finite-dimensional approximation space $S = \text{span}(\{\psi_i(\xi)\}_{i=1}^{n_\xi}) \subset L^2(\Gamma)$. For spatial discretization, a div-stable mixed finite element method [9] is considered, the Taylor–Hood element consisting of biquadratic velocities and bilinear pressure. Basis sets for the velocity space V_E^h and the pressure space Q_D^h are denoted by

$$\left\{ \begin{bmatrix} \phi_i(x) \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ \phi_i(x) \end{bmatrix} \right\}_{i=1}^{n_u}$$

and $\{\varphi_i(x)\}_{i=1}^{n_p}$, respectively. Then the fully discrete version of (9) can be written as

$$\vec{u}(x,\xi) = \begin{bmatrix} u^x(x,\xi) \\ u^y(x,\xi) \end{bmatrix} \approx \begin{bmatrix} \sum_{i=1}^{n_\xi}\sum_{j=1}^{n_u} u_{ij}^x \phi_j(x)\psi_i(\xi) \\ \sum_{i=1}^{n_\xi}\sum_{j=1}^{n_u} u_{ij}^y \phi_j(x)\psi_i(\xi) \end{bmatrix}, \quad p(x,\xi) \approx \sum_{i=1}^{n_\xi}\sum_{j=1}^{n_p} p_{ij}\varphi_j(x)\psi_i(\xi). \tag{10}$$

Let us introduce a vector notation for the coefficients, $\bar{u}_i^x \equiv [u_{i1}^x, \ldots, u_{in_u}^x]^T \in \mathbb{R}^{n_u}$, $\bar{u}_i^y \equiv [u_{i1}^y, \ldots, u_{in_u}^y]^T \in \mathbb{R}^{n_u}$, and $\bar{p}_i \equiv [p_{i1}, \ldots, p_{in_p}]^T \in \mathbb{R}^{n_p}$ for $i = 1, \ldots, n_\xi$, which, for each gPC index i, groups the horizontal velocity coefficients together followed by the vertical velocity coefficients, and then by the pressure coefficients, giving a vector

$$\bar{u}_i = [(\bar{u}_i^x)^T, (\bar{u}_i^y)^T, p_i^T]^T. \tag{11}$$

Taking $\nu(x,\xi)$ from (2) and replacing $\vec{u}(x,\xi)$, $p(x,\xi)$ in (8) with their discrete approximations (10) yields a system of linear equations of order $(2n_u + n_p)n_\xi$. The coefficient matrix has a Kronecker-product structure,

$$J \equiv \sum_{l=1}^{n_\xi} G_l \otimes \mathcal{F}_l, \tag{12}$$

where G_l refers to the "stochastic matrix"

$$[G_l]_{ij} = \langle \psi_l \psi_i \psi_j \rangle_\rho, \quad l = 1, \dots, n_\xi,$$

with $\psi_1 = 1$, $\psi_l(\xi) = \gamma \xi_{l-1}$, $l = 2, \dots, n_\nu + 1$, where the scalar γ is chosen so that $\langle \psi_l^2 \rangle_\rho = 1$, and

$$\mathcal{F}_1 = \begin{bmatrix} F_1^{xx} & F_1^{xy} & B^{xT} \\ F_1^{yx} & F_1^{yy} & B^{yT} \\ B^x & B^y & 0 \end{bmatrix}, \qquad \mathcal{F}_l = \begin{bmatrix} F_l^{xx} & F_l^{xy} & 0 \\ F_l^{yx} & F_l^{yy} & 0 \\ 0 & 0 & 0 \end{bmatrix}, \; l = 2, \dots, n_\xi, \tag{13}$$

where some of the block entries in (13) depend on whether Newton or Picard iteration is used, with $F_l^{xx} \equiv A_l + N_l + W_l^{xx}$, $F_l^{yy} \equiv A_l + N_l + W_l^{yy}$, $F_l^{xy} \equiv W_l^{xy}$, $F_l^{yx} \equiv W_l^{yx}$ for the Newton iteration and $F_l^{xx} \equiv A_l + N_l$, $F_l^{yy} \equiv A_l + N_l$, $F_l^{xy} \equiv 0$, $F_l^{yx} \equiv 0$ for Picard iteration. We denote the matrices of (12) and (13) derived from the Newton iteration and the Picard iteration by J_N, $\mathcal{F}_l^N$ and J_P, $\mathcal{F}_l^P$, respectively. Here, A_l is a symmetric matrix defined as

$$[A_1]_{ij} \equiv \int_D \nu_0 (\nabla \phi_i \cdot \nabla \phi_j), \quad [A_l]_{ij} \equiv \int_D \frac{\sigma_\nu \nu_{l-1}}{\gamma} (\nabla \phi_i \cdot \nabla \phi_j), \quad l = 2, \dots, n_\nu + 1, \tag{14}$$

N_l is a discrete convection operator, which depends on the coefficient $\vec{u}_l$ of the current velocity estimate and is derived from the convection term, $c(\vec{u}; \delta\vec{u}, \vec{v})$,

$$[N_l]_{ij} \equiv \int_D (\vec{u}_l \cdot \nabla \phi_j) \phi_i,$$

$W_l^{xx}, W_l^{xy}, W_l^{yx}$, and W_l^{yy} are matrices representing weak derivatives of the current velocity estimate in the x and y directions, which are derived from $c(\delta\vec{u}; \vec{u}, \vec{v})$, for example,

$$[W_l^{xy}]_{ij} \equiv \int_D \frac{\partial u_l^x}{\partial y} \phi_i \phi_j,$$

and B^x and B^y make up a discrete divergence operator,

$$[B^x]_{ij} \equiv -\int_D \varphi_i \frac{\partial \phi_j}{\partial x}, \quad [B^y]_{ij} \equiv -\int_D \varphi_i \frac{\partial \phi_j}{\partial y}. \tag{15}$$

If the number of gPC polynomial terms in (10) is larger than the number of terms in (2) (i.e., $n_\xi > n_\nu + 1$), we simply set $\{A_l\}_{l=n_\nu+2}^{n_\xi}$ as matrices containing only zeros so that for the

Newton iteration $F_l^{xx} = N_l + W_l^{xx}$, $F_l^{yy} = N_l + W_l^{yy}$ and for the Picard iteration $F_l = N_l$ for $l = n_\nu + 2, \dots, n_\xi$.

The discrete system of equations to be solved at each iteration then has the form

$$J^n \delta u^n = -\bar{r}^n, \tag{16}$$

where $J = J_N$ or J_P as appropriate, evaluated at the nth discrete iterate $\bar{u}^n := [(\bar{u}_1^n)^T \dots (\bar{u}_{n_\xi}^n)^T]^T \in \mathbb{R}^{(2n_u+n_p)n_\xi}$ with $\bar{u}_i^n$ as in (11), and $\bar{r}^n$ is the discrete version of the residual (7) evaluated at $\bar{u}^n$.

2.3. Nonlinear iteration.

For the nonlinear iteration, we consider a hybrid Picard–Newton strategy. An initial approximation for the nonlinear solution is computed by solving the parameterized Stokes equations,

$$\begin{aligned} -\nabla \cdot \nu(x,\xi)\nabla \vec{u}(x,\xi) + \nabla p(x,\xi) &= \vec{f}(x,\xi), \\ \nabla \cdot \vec{u}(x,\xi) &= 0. \end{aligned}$$

The discrete Stokes operator, which is obtained from the stochastic Galerkin discretization as shown in section 2.2, is

$$\left(\sum_{l=1}^{n_\nu+1} G_l \otimes \mathcal{S}_l \right) \bar{u}_{\text{st}} = b_{\text{st}}, \tag{17}$$

where

$$\mathcal{S}_1 = \begin{bmatrix} A_1 & B^T \\ B & 0 \end{bmatrix}, \qquad \mathcal{S}_l = \begin{bmatrix} A_l & 0 \\ 0 & 0 \end{bmatrix}, \quad l = 2, \dots, n_\nu + 1,$$

with $\{A_l\}_{l=1}^{n_\nu+1}$ defined in (14) and B defined in (15). After this initial computation, updates to the solution are computed by first solving m_p Picard systems with coefficient matrix J_P and then using Newton's method with coefficient matrix J_N to compute the solution. The Newton iteration is performed until a certain stopping criteria is satisfied, i.e., either the nonlinear residual norm meets a relative error tolerance ϵ_{nl} or a maximal number m_n of Newton steps is performed. Algorithm 1 summarizes the hybrid nonlinear iteration method.

At each nonlinear iteration step, the update $\delta\bar{u}^n$ is computed by solving (16). The order of the system $(2n_u + n_p)n_\xi$ grows fast as the number of random variables used to parameterize the random viscosity increases. Even for a moderate-dimensional stochastic Navier–Stokes problem, solving a sequence of linear systems of order $(2n_u + n_p)n_\xi$ can be computationally prohibitive. To address this issue, we present an efficient variant of nonlinear iteration using the Kronecker-product structure in the following sections. See [26] for adaptive strategies for switching from Picard iteration to Newton's method, and [17] for an alternative approach of hybridizing Picard iteration and Newton's method.

3. Low-rank Newton–Krylov method.

In this section, we outline the formalism in which the solutions to (16) can be efficiently approximated by low-rank objects while not losing much accuracy and we show how solvers are adjusted within this formalism.

3.1. Approximation in low rank.

In order to develop a low-rank variant of Algorithm 1, we begin by introducing some concepts to define the rank of computed quantities. Let

Algorithm 1 Nonlinear iteration—The hybrid approach

```
1: compute an approximate solution of $A_{\mathrm{st}}\bar{u}_{\mathrm{st}} = b_{\mathrm{st}}$ in (17)
2: set an initial approximate for the Navier–Stokes problem $\bar{u}^0 := \bar{u}_{\mathrm{st}}$
3: for $k = 0, \dots, m_p - 1$ do {Picard iteration}
4:     solve $J_{\mathrm{P}}^k \, \delta\bar{u}^k = -\bar{r}^k$
5:     update $\bar{u}^{k+1} := \bar{u}^k + \delta\bar{u}^k$
6: end for
7: while $k < m_n$ and $\|\bar{r}^k\|_2 > \epsilon_{\mathrm{nl}}\|\bar{r}^0\|_2$ do {Newton iteration}
8:     solve $J_{\mathrm{N}}^k \, \delta\bar{u}^k = -\bar{r}^k$
9:     update $\bar{u}^{k+1} := \bar{u}^k + \delta\bar{u}^k$
10: end while
```

$X = [\bar{x}_1, \dots, \bar{x}_{n_2}] \in \mathbb{R}^{n_1 \times n_2}$ and $\bar{x} = [\bar{x}_1^T, \dots, \bar{x}_{n_2}^T]^T \in \mathbb{R}^{n_1 n_2}$, where $\bar{x}_i \in \mathbb{R}^{n_1}$ for $i = 1, \dots, n_2$. That is, $\bar{x}$ can be constructed by rearranging the elements of X, and vice versa. Suppose X has rank α_x. Then two mathematically equivalent expressions for X and $\bar{x}$ are given by

$$X = YZ^T = \sum_{i=1}^{\alpha_{\bar{x}}} \bar{y}_i \bar{z}_i^T \quad \Leftrightarrow \quad \bar{x} = \sum_{i=1}^{\alpha_{\bar{x}}} \bar{z}_i \otimes \bar{y}_i, \tag{18}$$

where $Y \equiv [\bar{y}_1, \dots, \bar{y}_{\alpha_{\bar{x}}}] \in \mathbb{R}^{n_1 \times \alpha_{\bar{x}}}$, $Z \equiv [\bar{z}_1, \dots, \bar{z}_{\alpha_{\bar{x}}}] \in \mathbb{R}^{n_2 \times \alpha_{\bar{x}}}$ with $\bar{y}_i \in \mathbb{R}^{n_1}$, and $\bar{z}_i \in \mathbb{R}^{n_2}$ for $i = 1, \dots, \alpha_{\bar{x}}$. The representation of X and its rank is standard matrix notation; we also use α_x to refer to the rank of the corresponding vector $\bar{x}$.

With this definition of rank, our goal is to inexpensively find a low-rank approximate solution $\bar{u}^k$ satisfying $\|\bar{r}^k\|_2 \le \epsilon_{\mathrm{nl}}\|\bar{r}^0\|_2$ for small enough ϵ_{nl}. To achieve this goal, we approximate updates $\delta\bar{u}^k$ in low-rank form using a low-rank variant of the GMRES method, which exploits the Kronecker product structure in the system matrix as in (12) and (17). In the following section, we present the solutions $\bar{u}$ (and $\delta\bar{u}$) in the formats of (18) together with matrix and vector operations that are essential for developing the low-rank GMRES method.

3.2. Solution coefficients in Kronecker-product form.

We seek separate low-rank approximations of the horizontal and vertical velocity solutions and the pressure solution. With the representation shown in (18), the solution coefficient vector $\bar{u} \in \mathbb{R}^{(2n_u+n_p)n_\xi}$, which consists of the coefficients of the velocity and pressure solutions (10), has an equivalent matricized representation $U = [U^{xT}, U^{yT}, P^T]^T \in \mathbb{R}^{(2n_u+n_p)\times n_\xi}$, where $U^x = [\bar{u}_1^x, \dots, \bar{u}_{n_\xi}^x], U^y = [\bar{u}_1^y, \dots, \bar{u}_{n_\xi}^y] \in \mathbb{R}^{n_u \times n_\xi}$, and $P = [\bar{p}_1, \dots, \bar{p}_{n_\xi}] \in \mathbb{R}^{n_p \times n_\xi}$. The components admit the following representations:

$$U^x = \sum_{i=1}^{\alpha_{\bar{u}^x}} (\bar{v}_i^x)(\bar{w}_i^x)^T = (V^x)(W^x)^T \quad \Leftrightarrow \quad \bar{u}^x = \sum_{i=1}^{\alpha_{\bar{u}^x}} \bar{w}_i^x \otimes \bar{v}_i^x, \tag{19}$$

$$U^y = \sum_{i=1}^{\alpha_{\bar{u}^y}} (\bar{v}_i^y)(\bar{w}_i^y)^T = (V^y)(W^y)^T \quad \Leftrightarrow \quad \bar{u}^y = \sum_{i=1}^{\alpha_{\bar{u}^y}} \bar{w}_i^y \otimes \bar{v}_i^y, \tag{20}$$

$$P = \sum_{i=1}^{\alpha_{\bar{p}}} (\bar{v}_i^p)(\bar{w}_i^p)^T = (V^p)(W^p)^T \quad \Leftrightarrow \quad \bar{p} = \sum_{i=1}^{\alpha_{\bar{p}}} \bar{w}_i^p \otimes \bar{v}_i^p, \tag{21}$$

where $V^x = [\bar{v}_1^x \dots \bar{v}_{\alpha_{\bar{u}^x}}^x]$, $W^x = [\bar{w}_1^x \dots \bar{w}_{\alpha_{\bar{u}^x}}^x]$, $\alpha_{\bar{u}^x}$ is the rank of $\bar{u}^x$ and U^x, and the same interpretation can be applied to $\bar{u}^y$ and $\bar{p}$.

3.2.1. Matrix operations. In this section, we introduce essential matrix operations used by the low-rank GMRES methods, using the representations shown in (19)–(21). First, consider the matrix-vector product with the system matrix (12) and vectors (19)–(21),

$$J^n \bar{u}^n = \left(\sum_{l=1}^{n_\xi} G_l \otimes \mathcal{F}_l^n \right) \bar{u}^n. \tag{22}$$

The expression (22) has the equivalent matricized form $\sum_{l=1}^{n_\xi} \mathcal{F}_l^n U^n G_l^T$, which can be evaluated using the componentwise representation of $\mathcal{F}_l$ as in (13), for example,

$$\mathcal{F}_1^n U^n G_1^T = \begin{bmatrix} F_1^{xx,n} V^{x,n} (G_1 W^{x,n})^T + F_1^{xy,n} V^{y,n} (G_1 W^{y,n})^T + B^{x^T} V^{p,n} (G_1 W^{p,n})^T \\ F_1^{yx,n} V^{x,n} (G_1 W^{x,n})^T ! F_1^{yy,n} V^{y,n} (G_1 W^{y,n})^T + B^{y^T} V^{p,n} (G_1 W^{p,n})^T \\ B^x V^{x,n} (G_1 W^{x,n})^T + B^y V^{y,n} (G_1 W^{y,n})^T \end{bmatrix}. \tag{23}$$

Equivalently, in the Kronecker-product structure, the matrix-vector product (23) updates each set of solution coefficients as follows:

$$\sum_{l=1}^{n_\xi} \left((G_l \otimes F_l^{xx,n}) \bar{u}^{x,n} + (G_l \otimes F_l^{xy,n}) \bar{u}^{y,n} \right) + (G_1 \otimes B^{x^T}) \bar{p}^n \quad (x\text{-velocity}), \tag{24}$$

$$\sum_{l=1}^{n_\xi} \left((G_l \otimes F_l^{yx,n}) \bar{u}^{x,n} + (G_l \otimes F_l^{yy,n}) \bar{u}^{y,n} \right) + (G_1 \otimes B^{y^T}) \bar{p}^n \quad (y\text{-velocity}), \tag{25}$$

$$(G_1 \otimes B^x) \bar{u}^{x,n} + (G_1 \otimes B^y) \bar{u}^{y,n} \quad (\text{pressure}), \tag{26}$$

where each matrix-vector product can be performed by exploiting the Kronecker-product structure, for example,

$$\sum_{l=1}^{n_\xi} (G_l \otimes F_l^{xx,n}) \bar{u}^{x,n} = \sum_{l=1}^{n_\xi} G_l \otimes F_l^{xx,n} \sum_{i=1}^{\alpha_{\bar{u}^x}} w_i^x \otimes v_i^x = \sum_{l=1}^{n_\xi} \sum_{i=1}^{\alpha_{\bar{u}^x}} G_l w_i^x \otimes F_l^{xx,n} v_i^x. \tag{27}$$

The matrix-vector product shown in (24)–(26) requires $O(2n_u + n_p + n_\xi)$ flops, whereas (22) requires $O((2n_u + n_p) n_\xi)$ flops. Thus, as the problem size grows, the additive form of the former count grows much less rapidly than the multiplicative form of (22).

The addition of two vectors $\bar{u}^x$ and $\bar{u}^y$ can also be efficiently performed in the Kronecker-product structure,

$$\bar{u}^x + \bar{u}^y = \sum_{i=1}^{\alpha_{\bar{u}^x}} w_i^x \otimes v_i^x + \sum_{i=1}^{\alpha_{\bar{u}^y}} w_i^y \otimes v_i^y = \sum_{i=1}^{\alpha_{\bar{u}^x} + \alpha_{\bar{u}^y}} \hat{w}_i \otimes \hat{v}_i, \tag{28}$$

where $\hat{v}_i = v_i^x$, $\hat{w}_i = w_i^x$ for $i = 1, \dots, \alpha_{\bar{u}^x}$, and $\hat{v}_i = v_i^y$, $\hat{w}_i = w_i^y$ for $i = \alpha_{\bar{u}^x} + 1, \dots, \alpha_{\bar{u}^x} + \alpha_{\bar{u}^y}$.

Inner products can be performed with similar efficiencies. Consider two vectors $\bar{x}_1$ and $\bar{x}_2$, whose matricized representations are

$$X_1 = \begin{bmatrix} Y_{11}Z_{11}^T \\ Y_{12}Z_{12}^T \\ Y_{13}Z_{13}^T \end{bmatrix}, \qquad X_2 = \begin{bmatrix} Y_{21}Z_{21}^T \\ Y_{22}Z_{22}^T \\ Y_{23}Z_{23}^T \end{bmatrix}. \tag{29}$$

We consider the Frobenius inner product [19] between X_1 and X_2, $\langle X_1, X_2\rangle_{\mathrm{F}}$, as

$$\bar{x}_1^T\bar{x}_2 = \mathrm{trace}((Y_{11}Z_{11}^T)^TY_{21}Z_{21}^T) + \mathrm{trace}((Y_{12}Z_{12}^T)^TY_{22}Z_{22}^T) + \mathrm{trace}((Y_{13}Z_{13}^T)^TY_{23}Z_{23}^T),$$

where trace(X) is defined as a sum of the diagonal entries of the matrix X. An efficient implementation of this inner product does not use the trace formulas, but instead constructs $\widehat{Y}_j = Y_{1j}Y_{2j}^T$, $\widehat{Z}_j = Z_{1j}Z_{2j}^T$, for $j = 1, 2, 3$, and

$$\langle X_1, X_2\rangle_{\mathrm{F}} = \mathrm{sum}\left(\widehat{Y}_1 \circ \widehat{Z}_1\right) + \mathrm{sum}\left(\widehat{Y}_2 \circ \widehat{Z}_2\right) + \mathrm{sum}\left(\widehat{Y}_3 \circ \widehat{Z}_3\right),$$

where $Y \circ Z$ is the elementwise (Hadamard) matrix product and the sum is over all matrix elements. We used this construction in our implementation.

Although the matrix-vector product and the sum, as described in (27) and (28), can be performed efficiently, the results of (27) and (28) are represented by $n_\xi\alpha_{\bar{u}^x}$ terms and $\alpha_{\bar{u}^x}+\alpha_{\bar{u}^y}$ terms, respectively, which typically causes the ranks of the computed quantities to be higher than the inputs for the computations and potentially undermines the efficiency of the solution method. To resolve this issue, a truncation operator will be used to modify the result of matrix-vector products and sums and to force the ranks of quantities used to be small.

3.2.2. Truncation of $U^{x,n}$, $U^{y,n}$, and P^n.

Consider the velocity and the pressure represented in a matrix form as in (19)–(21). The best α-rank approximation of a matrix can be found by using the singular value decomposition (SVD) [14, 18]. We define a truncation operator for a given matrix $U = VW^T$ whose rank is α_U,

$$\mathcal{T}_{\epsilon_{\mathrm{trunc}}} : U \to \tilde{U},$$

where the rank of U is larger than the rank of $\tilde{U}$ (i.e., $\alpha_U \gg \alpha_{\tilde{U}}$). The truncation operator $\mathcal{T}_{\epsilon_{\mathrm{trunc}}}$ compresses U to $\tilde{U}$ such that $\|\tilde{U} - U\|_F \le \epsilon_{\mathrm{trunc}}\|U\|_F$, where $\|\cdot\|_F$ is the Frobenius norm. To achieve this goal, the SVD of U can be computed (i.e., $U = \hat{V}D\hat{W}^T$, where $D = \mathrm{diag}(d_1, \ldots, d_n)$ is the diagonal matrix of singular values).[1] Letting $\{\hat{v}_i\}$ and $\{\hat{w}_i\}$ denote the singular vectors, the approximation is $\tilde{U} = \sum_{i=1}^{\alpha_{\tilde{U}}} \tilde{v}_i\tilde{w}_i^T$ with $\tilde{v}_i = d_i\hat{v}_i$ and the

[1] In computing the SVD, we follow the approach used in [10], where, based on complexity analysis of costs, one can adaptively choose a method to compute the SVD of U either by computing QR decompositions of the factors $V = Q_VR_V$ and $W = Q_WR_W$, and computing the SVD of $R_VR_W^T$ as in [14], or by computing the SVD of U directly when that is less expensive.

truncation rank $\alpha_{\tilde{U}}$ is determined by the condition

$$\sqrt{d^2_{\alpha_{\tilde{U}}+1} + \cdots + d^2_n} \leq \epsilon_{\text{trunc}} \sqrt{d^2_1 + \cdots + d^2_n}. \tag{30}$$

3.3. Low-rank GMRES method. We describe the lrGMRES method with a generic linear system $Ax = b$. The method follows the standard Arnoldi iteration used by GMRES [25]: construct a set of basis vectors $\{v_i\}_{i=1}^{m_{\text{gm}}}$ by applying the linear operator A to basis vectors, i.e., $w_j = Av_j$ for $j = 1, \ldots, m_{\text{gm}}$, and orthogonalizing the resulting vector w_j with respect to previously generated basis vectors $\{v_i\}_{i=1}^{j-1}$. In the low-rank GMRES method [2], iterates, basis vectors v_i, and intermediate quantities w_i are represented in terms of the factors of their matricized representations (so that X in (18) would be represented using Y and Z without explicit construction of X), and matrix operations such as matrix-vector products are performed as described in section 3.2.1. As pointed out in section 3.2.1, these matrix operations typically tend to increase the rank of the resulting quantity, and this is resolved by interleaving the truncation operator $\mathcal{T}$ with the matrix operations. The low-rank GMRES method computes a new iterate by solving

$$\min_{\bar{\beta} \in \mathbb{R}^{m_{\text{gm}}}} \|b - A(x_0 + V_{m_{\text{gm}}} \bar{\beta})\|_2 \tag{31}$$

and constructing a new iterate $x_1 = x_0 + V_{m_{\text{gm}}} \bar{\beta}$, where x_0 is an initial guess. Due to truncation, the basis vectors $\{v_i\}$ are not orthogonal and $\text{span}(V_{m_{\text{gm}}})$, where $V_{m_{\text{gm}}} = [v_1 \ldots v_{m_{\text{gm}}}]$, is not a Krylov subspace, so that (31) must be solved explicitly rather than exploiting Hessenberg structure as in standard GMRES. Algorithm 2 summarizes the lrGMRES method. We will use this method to solve the linear system of (16).

Algorithm 2 lrGMRES(m_{gm}) (restarted low-rank GMRES method)

```
1: set the initial solution $x_0$
2: for $k = 0, 1, \ldots$ do
3:   $r^k_{\text{gm}} := b - Ax_k$
4:   if $\|r^k_{\text{gm}}\|_2 / \|b\|_2 < \epsilon_{\text{gmres}}$ or $\|r^k_{\text{gm}}\|_2 \geq \|r^{k-1}_{\text{gm}}\|_2$ then
5:     return $x_k$
6:   end if
7:   $\bar{v}_1 := \mathcal{T}_{\epsilon_{\text{trunc}}}(r^k_{\text{gm}})$
8:   $v_1 := \bar{v}_1 / \|\bar{v}_1\|_2$
9:   for $j = 1, \ldots, m_{\text{gm}}$ do
10:    $w_j := Av_j$
11:    solve $(V_j^T V_j)\bar{\alpha} = V_j^T w_j$ where $V_j = [v_1, \ldots, v_j]$
12:    $\bar{v}_{j+1} := \mathcal{T}_{\epsilon_{\text{trunc}}}\left(w_j - \sum_{i=1}^{j} \alpha_i v_i\right)$
13:    $v_{j+1} := \bar{v}_{j+1} / \|\bar{v}_{j+1}\|_2$
14:  end for
15:  solve $(W^T_{m_{\text{gm}}} A V_{m_{\text{gm}}})\bar{\beta} = W^T_{m_{\text{gm}}} r^k_{\text{gm}}$ where $W_j = [w_1, \ldots, w_j]$
16:  $x_{k+1} := \mathcal{T}_{\epsilon_{\text{trunc}}}(x_k + V_{m_{\text{gm}}}\bar{\beta})$
17: end for
```

3.4. Preconditioning. We also use preconditioning to speed up convergence of the lrGMRES method. For this, we consider a right-preconditioned system

$$J^n (M^n)^{-1} \delta\tilde{u}^n = -\bar{r}^n,$$

where M^n is the preconditioner and $M^n \delta\bar{u}^n = \delta\tilde{u}^n$ such that $J^n \delta\bar{u}^n = -\bar{r}^n$. We consider an approximate mean-based preconditioner [22], which is derived from the matrix $G_1 \otimes \mathcal{F}_1$ associated with the mean ν_0 of the random viscosity (2),

$$M^n = G_1 \otimes \begin{bmatrix} M_A^n & B^T \\ 0 & -M_s^n \end{bmatrix}, \tag{32}$$

where

$$M_A^n = \begin{bmatrix} A_1^{xx} + N_1^n & 0 \\ 0 & A_1^{yy} + N_1^n \end{bmatrix} \qquad \text{(Picard iteration)},$$

$$M_A^n = \begin{bmatrix} A_1^{xx} + N_1^n + W_1^{xx,n} & 0 \\ 0 & A_1^{yy} + N_1^n + W_1^{yy,n} \end{bmatrix} \qquad \text{(Newton iteration)}.$$

For approximating the action of the inverse, $(M_s^n)^{-1}$, we choose the boundary-adjusted least-squares commutator (LSC) preconditioning scheme [9],

$$M_s^n = B F_1^{-1} B^T \approx (BH^{-1}B^T)(BM_*^{-1}F_1 H^{-1} B^T)^{-1}(BM_*^{-1}B^T),$$

where M_* is the diagonal of the velocity mass matrix and $H = D^{-1/2} M_* D^{-1/2}$, where D is a diagonal scaling matrix deemphasizing contributions near the boundary. During the iteration, the action of the inverse of the preconditioner (32) can be applied to a vector in a manner analogous to (24)–(26).

4. Inexact nonlinear iteration.

As outlined in Algorithm 1, we use the hybrid approach, employing a few steps of Picard iteration followed by Newton iteration, and the linear systems (lines 4 and 8 in Algorithm 1) are solved using lrGMRES (Algorithm 2). We extend the hybrid approach to an inexact variant based on an inexact Newton algorithm, in which the accuracy of the approximate linear system solution is tied to the accuracy of the nonlinear iterate (see, e.g., [13] and references therein). That is, when the nonlinear iterate is far from the solution, the linear systems may not have to be solved accurately. Thus, a sequence of iterates $\bar{u}^{n+1} := \bar{u}^n + \delta\bar{u}^n$ is computed where $\delta\bar{u}^n$ satisfies

$$\|J_N^n \delta\bar{u}^n + \bar{r}^n\|_2 \le \epsilon_{\text{gmres}}^n \|\bar{r}^n\|_2 \quad (J_P \text{ for Picard iteration}),$$

where the lrGMRES stopping tolerance ($\epsilon_{\text{gmres}}^n$ of Algorithm 2) is given by

$$\epsilon_{\text{gmres}}^n := \rho_{\text{gmres}} \|\bar{r}^n\|_2, \tag{33}$$

where $0 < \rho_{\text{gmres}} \le 1$. With this strategy, the system (22) is solved with increased accuracy as the error becomes smaller, leading to savings in the average cost per step and, as we will show, with no degradation in the asymptotic convergence rate of the nonlinear iteration.

In addition, in Algorithms 1 and 2, the truncation operator $\mathcal{T}_{\epsilon_{\text{trunc}}}$ is used for the low-rank approximation of the nonlinear iterate (i.e., truncating $\bar{u}^x$, $\bar{u}^y$, and $\bar{p}$ at lines 5 and 9 in Algorithm 1) and updates (i.e., truncating $\delta\bar{u}^x$, $\delta\bar{u}^y$, and $\delta\bar{p}$ at lines 7, 12, and 16 in Algorithm 2). As the lrGMRES stopping tolerance is adaptively determined by the criterion (33), we also choose the value of the truncation tolerances $\epsilon_{\text{trunc,sol}}$ and $\epsilon^n_{\text{trunc,corr}}$, adaptively. For truncating the nonlinear iterate, the truncation tolerance for the iterate $\epsilon^n_{\text{trunc,sol}}$ is chosen based on the nonlinear iteration stopping tolerance,

$$\epsilon_{\text{trunc,sol}} := \rho_{\text{nl}}\epsilon_{\text{nl}},$$

where $0 < \rho_{\text{nl}} \leq 1$. For truncating the updates (or corrections), the truncation tolerance for the correction $\epsilon^n_{\text{trunc,corr}}$ is adaptively chosen based on the stopping tolerance of the linear solver,

$$\epsilon^n_{\text{trunc,corr}} := \rho_{\text{trunc,P}}\epsilon^n_{\text{gmres}} \quad \text{(for the } n\text{th Picard step)},$$
$$\epsilon^n_{\text{trunc,corr}} := \rho_{\text{trunc,N}}\epsilon^n_{\text{gmres}} \quad \text{(for the } n\text{th Newton step)},$$

where $0 < \rho_{\text{trunc,P}}, \rho_{\text{trunc,N}} \leq 1$. Thus, for computing nth update $\delta\bar{u}^n$, we set $\epsilon_{\text{trunc}} = \epsilon^n_{\text{trunc,corr}}$ in Algorithm 2. The complete computation is shown in Algorithm 3.

Algorithm 3 Inexact nonlinear iteration with adaptive tolerances

```
1: set $\epsilon_{\text{trunc,sol}} := \rho_{\text{nl}}\epsilon_{\text{nl}}$
2: compute an approximate solution of $A_{\text{st}}\bar{u}_{\text{st}} = b_{\text{st}}$ using Algorithm 2
3: set an initial guess for the Navier–Stokes problem $\bar{u}^0 := \bar{u}_{\text{st}}$
4: **for** $k = 0, \dots, m_p - 1$ **do** {Picard iteration}
5:    set $\epsilon^k_{\text{gmres}} = \rho_{\text{gmres}}\|\bar{r}^k\|_2$, and $\epsilon^k_{\text{trunc,corr}} = \rho_{\text{trunc,P}}\|\bar{r}^k\|_2$
6:    solve $J^k_{\text{P}}\,\delta\bar{u}^k = -\bar{r}^k$ using Algorithm 2
7:    update $\bar{u}^{k+1} := \mathcal{T}_{\epsilon_{\text{trunc,sol}}}(\bar{u}^k + \delta\bar{u}^k)$
8: **end for**
9: **while** $\|\bar{r}^k\|_2 > \epsilon_{\text{nl}}\|\bar{r}^0\|_2$ **do** {Newton iteration}
10:   set $\epsilon^k_{\text{gmres}} = \rho_{\text{gmres}}\|\bar{r}^k\|_2$, and $\epsilon^k_{\text{trunc,corr}} = \rho_{\text{trunc,N}}\|\bar{r}^k\|_2$
11:   solve $J^k_{\text{N}}\,\delta\bar{u}^k = -\bar{r}^k$ using Algorithm 2
12:   update $\bar{u}^{k+1} := \mathcal{T}_{\epsilon_{\text{trunc,sol}}}(\bar{u}^k + \delta\bar{u}^k)$
13: **end while**
```

5. Numerical results.

In this section, we present the results of numerical experiments on a model problem, flow around a square obstacle in a channel, for which the details are depicted in Figure 1. The domain has length 12 and height 2, and it contains a square obstacle centered at (2,0) with sides of length .25.

For the numerical experiments, we define the random viscosity (2) using a stochastic expansion with $\nu_i(x) = \sqrt{\lambda_i}\tilde{\nu}_i(x)$,

$$\nu(x,\xi) = \nu_0 + \sigma_\nu \sum_{i=1}^{\infty} \sqrt{\lambda_i}\tilde{\nu}_i(x)\xi_i, \tag{34}$$

where $\{\xi_i\}$ are uncorrelated random variables with zero mean and unit variance, ν_0 and σ^2_ν are the mean and the variance of $\nu(x,\xi)$, and $\{(\lambda_i, \tilde{\nu}_i(x))\}$ are eigenpairs of the eigenvalue problem associated with a covariance kernel $C(x,y)$. We consider two types of covariance

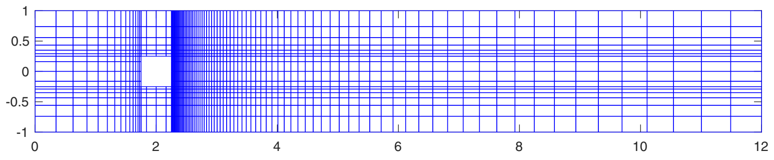


Figure 1. *Spatial domain and finite element discretization.*

kernel: absolute difference exponential (AE) and squared difference exponential (SE), which are defined via

$$(35)\qquad C^{\mathrm{AE}}(x,y)=\exp\left(-\sum_{i=1}^{2}\frac{|x_i-y_i|}{l}\right),\quad C^{\mathrm{SE}}(x,y)=\exp\left(-\sum_{i=1}^{2}\frac{(x_i-y_i)^2}{l}\right),$$

where $x=(x_1,x_2)$ and $y=(y_1,y_2)$ are points in the spatial domain, and l is a correlation length.[2] We assume that ξ_i (for $i=1,\ldots,n_\nu$) follows a uniform distribution over $[-\sqrt{3},\sqrt{3}]$. With this choice, $\gamma=1$ in (14). For the mean of the viscosity, we consider several choices, $\nu_0=\{\frac{1}{50},\frac{1}{100},\frac{1}{150}\}$, which correspond to $\mathrm{Re}_0=\{100,200,300\}$. We will also refer to the coefficient of variation (CoV), the relative size of the standard deviation with respect to the mean,

$$(36)\qquad CoV\equiv\frac{\sigma_\nu}{\nu_0}.$$

To ensure the positivity of the random field, we check $\nu_0(1-\frac{\sqrt{12}}{2}CoV)>0$, which holds for all benchmark problems tested. In most experiments, we use a truncated stochastic expansion of (34) with the largest five eigenvalues ($n_\nu=5$). For constructing the finite-dimensional approximation space $S=\mathrm{span}(\{\psi_i(\xi)\}_{i=1}^{n_\xi})$ in the parameter domain, we use orthogonal polynomials $\{\psi_i(\xi)\}_{i=1}^{n_\xi}$ of total degree $d_{\mathrm{tot}}=3$, which results in $n_\xi=56$. The orthogonal polynomials associated with uniform random variables are Legendre polynomials. For the spatial discretization, Taylor–Hood elements are used on a stretched grid, which results in $\{6320, 6320, 1640\}$ degrees of freedom in $\{u^x,u^y,p\}$, respectively (i.e., $n_u=6320$ and $n_p=1640$.) The implementation is based on the Incompressible Flow and Iterative Solver Software (IFISS) package [8, 27].

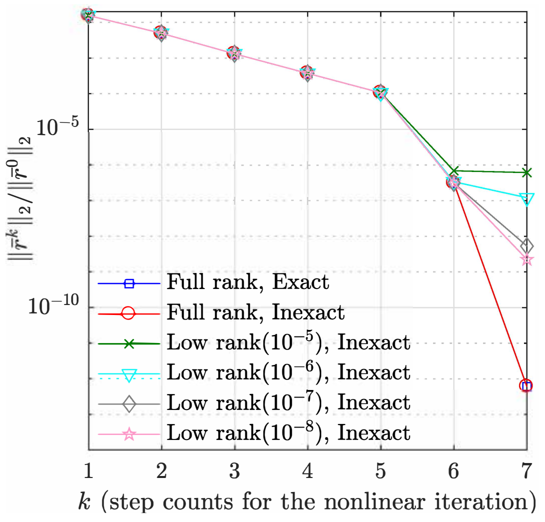
5.1. Low-rank inexact nonlinear iteration. In this section, we compare the results obtained from the low-rank inexact nonlinear iteration with those obtained from other methods, the exact and the inexact nonlinear iteration with full rank solutions, and the Monte Carlo method. Default parameter settings are listed in Table 1, where the truncation tolerances only apply to the low-rank method. Unless otherwise specified, the linear system is solved using a restarted version of low-rank GMRES, lrGMRES(20), which generates 20 basis vectors at each GMRES cycle.

We first examine the convergence behavior of the inexact nonlinear iteration for two model problems characterized by $\mathrm{Re}_0=100$, $CoV=1\%$, $l=32$ and $\mathrm{Re}_0=100$, $CoV=5\%$, $l=1$.

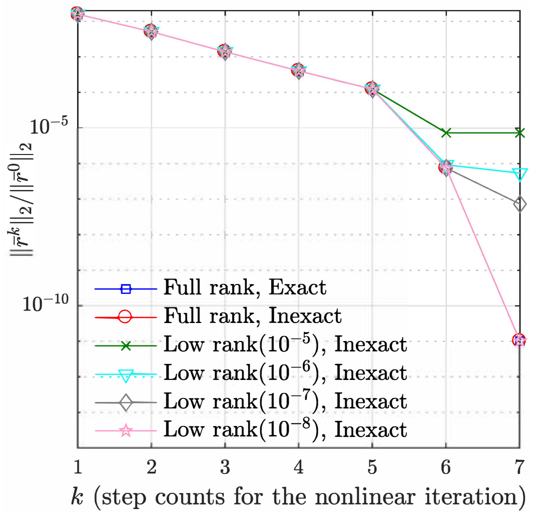
[2]Note that the expansion (34) has covariance given by $\sigma_\nu^2 C$, where C is one of the functions in (35).

Table 1
Tolerances and adaptive parameters.

Nonlinear iteration stopping tolerance	$\epsilon_{\text{nl}} = 10^{-5}$
GMRES tolerance (Stokes)	$\epsilon_{\text{gmres}} = 10^{-4}$
GMRES tolerances (Picard and Newton)	$\epsilon^n_{\text{gmres}} = \rho_{\text{gmres}} \lVert \bar{r}^n \rVert_2$ ($\rho_{\text{gmres}} = 10^{-.5}$)
Truncation tolerance for solutions	$\epsilon_{\text{trunc,sol}} = \rho_{\text{nl}} \epsilon_{\text{nl}}$ ($\rho_{\text{nl}} = 10^{-1}$)
Truncation tolerance for corrections	$\epsilon^n_{\text{trunc,corr}} = \rho_{\text{trunc}} \epsilon^n_{\text{gmres}}$ ($\rho_{\text{trunc}} = 10^{-1}$)



(a) $CoV = 1\%$ and $l = 32$



(b) $CoV = 5\%$ and $l = 1$

Figure 2. *Convergence of both exact and inexact nonlinear iterations (full-rank) and the low-rank inexact nonlinear iteration.*

Also, the SE covariance kernel in (35) is considered for both problems. We compute a full-rank solution using the exact nonlinear iteration ($\epsilon^n_{\text{gmres}} = 10^{-12}$ and no truncation) until the nonlinear iterate reaches the nonlinear stopping tolerance, $\epsilon_{\text{nl}} = 10^{-8}$. Then we compute another full-rank solution using the inexact nonlinear iteration (i.e., adaptive choice of $\epsilon^n_{\text{gmres}}$ as shown in Table 1 and no truncation). Last, we compute a low-rank approximate solution using the low-rank inexact nonlinear iteration (i.e., adaptive choices of $\epsilon^n_{\text{gmres}}$ and $\epsilon^n_{\text{trunc,corr}}$ as shown in Table 1 and for varying $\epsilon_{\text{trunc,sol}} = \{10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}\}$). Figure 2 shows the convergence behavior of the three methods. We use a hybrid approach, in which the first step corresponds to the Stokes problem (line 2 of Algorithm 3), the second–fifth steps correspond to the Picard iteration (lines 4–8 of Algorithm 3, and $m_p = 4$), and the sixth–seventh steps correspond to the Newton iteration (lines 9–13 of Algorithm 3). The choice of $m_p = 4$ (number of Picard steps) was derived from our empirical observation (also made in [28]) that with this choice the following Newton iteration always converged. We also allowed a maximal number of Newton steps $m_n = 4$, which was not reached in any of the experiments. Figure 2 confirms that the inexact nonlinear iteration is as effective as the exact nonlinear iteration. The low-rank inexact nonlinear iteration behaves similarly up to the sixth nonlinear

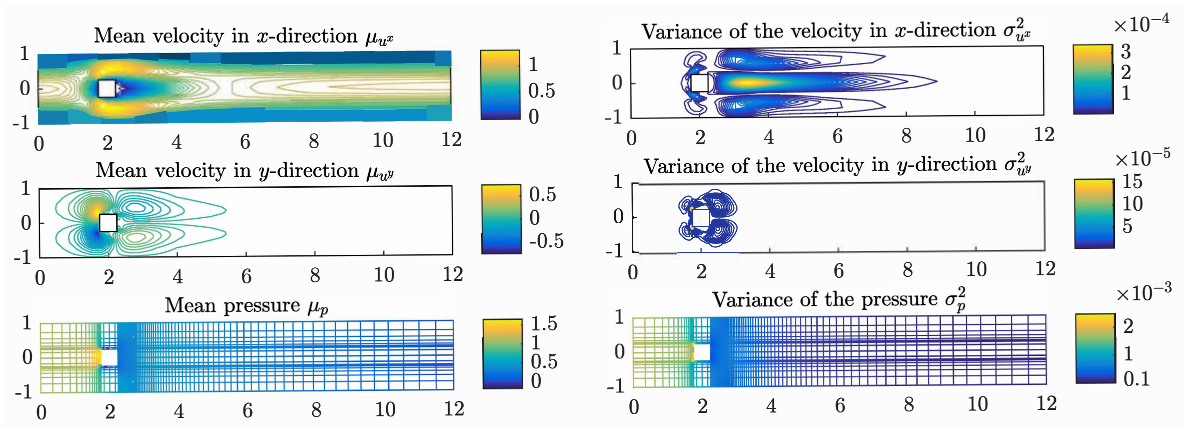


Figure 3. *Mean and variances of full-rank velocity solutions $u^x(x,\xi)$, $u^y(x,\xi)$, and pressure solution $p(x,\xi)$ for $\mathrm{Re}_0 = 100$, $CoV = 1\%$, and $l = 32$.*

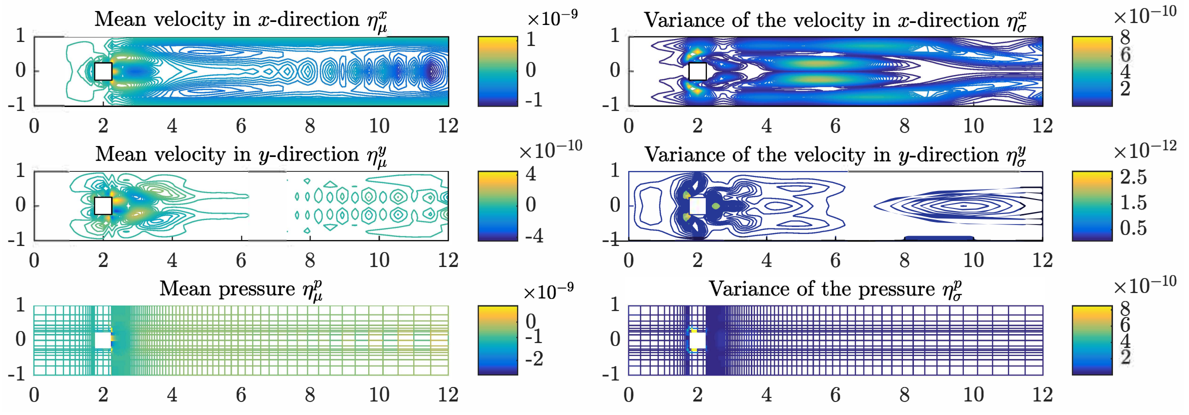


Figure 4. *Difference between the means and variances of the full-rank and the low-rank solutions for $\mathrm{Re}_0 = 100$, $CoV = 1\%$, and $l = 32$.*

step but when the truncation tolerances are large, $\epsilon_{\text{trunc,sol}} = \{10^{-5}, 10^{-6}\}$ in Figure 2(a) and $\epsilon_{\text{trunc,sol}} = \{10^{-5}, 10^{-6}, 10^{-7}\}$ in Figure 2(b), it fails to produce a nonlinear solution satisfying $\epsilon_{\text{nl}} = 10^{-8}$.

Figure 3 shows means and variances of the components of the full-rank solution of the problem with $\mathrm{Re}_0 = 100$, $CoV = 1\%$, and $l = 32$, given by

$$\mu_{u^x} = \mathbb{E}[u^x], \qquad \mu_{u^y} = \mathbb{E}[u^y], \qquad \mu_p = \mathbb{E}[p], \tag{37}$$

$$\sigma^2_{u^x} = \mathbb{E}[(u^x - \mu_{u^x})^2], \quad \sigma^2_{u^y} = \mathbb{E}[(u^y - \mu_{u^y})^2], \quad \sigma^2_p = \mathbb{E}[(p - \mu_p)^2]. \tag{38}$$

These quantities are easily computed by exploiting the orthogonality of basis functions in the gPC expansion. Figure 4 shows the differences in the means and variances of the solutions computed using the full-rank and the low-rank inexact nonlinear iteration. Let us denote the full-rank and low-rank horizontal velocity solutions by $u^{x,\text{full}}$ and $u^{x,\text{lr}}$, with analogous notation for the vertical velocity and the pressure. Thus, the differences in the means and the variances are

$$\eta^x_\mu = \mu_{u^{x,\text{full}}} - \mu_{u^{x,\text{lr}}}, \quad \eta^y_\mu = \mu_{u^{y,\text{full}}} - \mu_{u^{y,\text{lr}}}, \quad \eta^p_\mu = \mu_{p^{\text{full}}} - \mu_{p^{\text{lr}}},$$
$$\eta^x_\sigma = \sigma^2_{u^{x,\text{full}}} - \sigma^2_{u^{x,\text{lr}}}, \quad \eta^y_\sigma = \sigma^2_{u^{y,\text{full}}} - \sigma^2_{u^{y,\text{lr}}}, \quad \eta^p_\sigma = \sigma^2_{p^{\text{full}}} - \sigma^2_{p^{\text{lr}}}.$$

Figure 4 shows these differences, normalized by graph norms $\|\nabla\vec{\mu}_{u^{\text{full}}}\| + \|\mu_{p^{\text{full}}}\|$ for the means and $\|\nabla\vec{\sigma}^2_{u^{\text{full}}}\| + \|\sigma^2_{p^{\text{full}}}\|$ for the variances, where $\|\nabla\vec{u}\| = (\int_D \nabla\vec{u} : \nabla\vec{u}\, dx)^{\frac{1}{2}}$ and $\|p\| = (\int_D p^2 dx)^{\frac{1}{2}}$. Figure 4 shows that the normalized differences in the mean and the variance are of order 10^{-10} to 10^{-9} and 10^{-12} to 10^{-10}, respectively, i.e., the errors in low-rank solutions are considerably smaller than the magnitude of the truncation tolerances $\epsilon_{\text{trunc,sol}}, \epsilon_{\text{trunc,corr}}$ (see Table 1). This outcome is typical of our experience. For example, in another experiment with $CoV = 5\%$ and $l = 1$, the normalized differences in the mean and the variance are of order 10^{-9} to 10^{-8} and 10^{-11} to 10^{-9}.

5.2. Characteristics of the Galerkin solution.

In this section, we examine various properties of the Galerkin solutions, with emphasis on comparison of the low-rank and full-rank versions of these solutions and development of an enhanced understanding of the relation between the Galerkin solution and the polynomial chaos basis. We use the same experimental settings studied above and describe the result mainly with the model problem with $CoV = 1\%$ and $l = 32$.

We begin by comparing the Galerkin solution with the solution of the parameterized discrete system via Monte Carlo simulation. For the latter approach, spatial discretization of the Navier–Stokes equations (1), using the same Taylor–Hood element described in section 2.2, produces a deterministic parameterized nonlinear algebraic system whose finite element velocity $(u^x_h(x,\xi), u^y_h(x,\xi))^T$ and pressure $p_h(x,\xi)$ solutions for ξ comprise discrete approximations to solutions of the parameterized system (1). We can use these discrete solutions to estimate PDFs at a specific point in the spatial domain. For Monte Carlo simulation, we solve $n_{\text{MC}} = 25000$ such deterministic systems associated with n_{MC} realizations $\{\xi^{(k)}\}_{k=1}^{n_{\text{MC}}}$ in the parameter space. Using the MATLAB function `ksdensity`, the PDFs of $(u^x_h(x,\xi)$, $u^y_h(x,\xi)$, $p_h(x,\xi))$ are estimated at the spatial point with coordinates $(3.6436, 0)$, where the variance of $u^x_h(x,\xi)$ is large (see Figure 3). The results are shown in Figure 5, where the same sampling points $\xi^{(k)}$ are used with the Galerkin solutions and the Monte Carlo simulation of the parameterized discrete system. They indicate that the PDF estimate of the Galerkin solution is virtually identical to that obtained from Monte Carlo simulation of the parameterized discrete system, and there is essentially no difference between the low-rank and full-rank results.

Next, we explore some characteristics of the Galerkin solution, focusing on the horizontal velocity solution; the observations made here also hold for the other components of the solution. Given the coefficients of the discrete velocity solution in matricized form, U^x, the velocity solution is then approximated by

$$u^x(x,\xi) \approx \Phi^T(x) U^x \Psi(\xi),$$

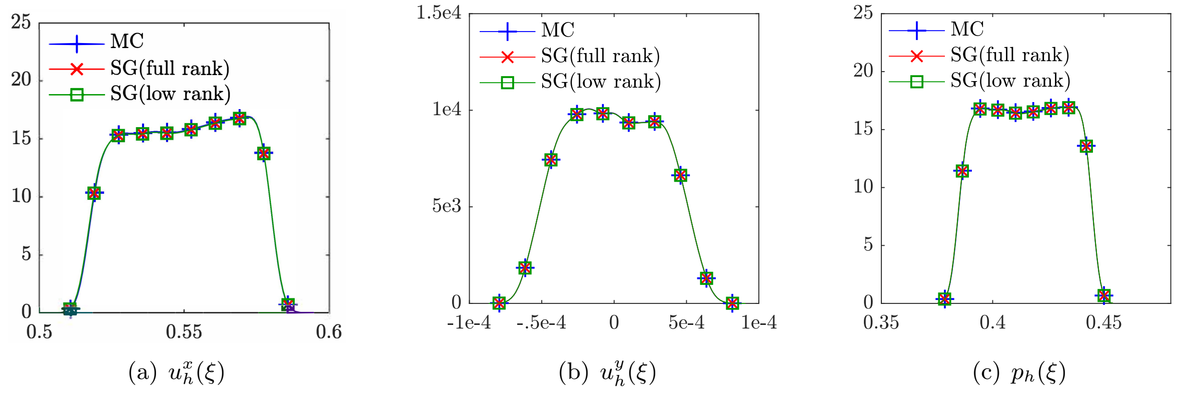


Figure 5. *Estimated PDFs of the velocities* $u_h^x(\xi)$, $u_h^y(\xi)$, *and the pressure* $p_h(\xi)$ *at the point* $(3.6436, 0)$.

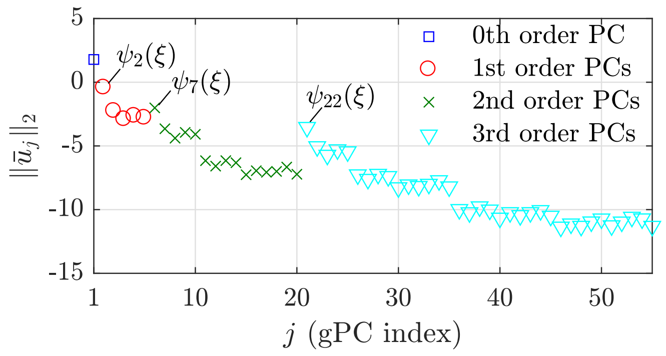


Figure 6. *Norms of the gPC coefficients* $\|\bar{u}_i^x\|_2$ *for* $\mathrm{Re}_0 = 100$, *CoV* $= 1\%$, *and* $l = 32$.

where $\Phi(x) = [\phi_1(x), \dots, \phi_{n_u}(x)]^T$ and $\Psi(\xi) = [\psi_1(\xi), \dots, \psi_{n_\xi}(\xi)]^T$. Consider in particular the component of this expression corresponding to the jth column of U^x,

$$\left(\sum_{i=1}^{n_u} u_{ij}^x \phi_i(x)\right) \psi_j(\xi)$$

so that this (jth) column $\bar{u}_j^x = [U^x]_j$ corresponds to the coefficient of the jth polynomial basis function ψ_j. Figure 6 plots the values of the coefficients $\|\bar{u}_j^x\|_2$. (This data is computed with $\mathrm{Re}_0 = 100$, $CoV = 1\%$, and SE covariance kernel with $l = 32$.) Note that the gPC indices $\{j\}$ are in one-to-one correspondence with multi-indices $d(j) = (d_1(j), \dots, d_{n_u}(j))$, where the element of the multi-index indicates the degree of univariate Legendre polynomial. The multi-indices $\{d(j)\}_{j=1}^{n_\xi}$ are ordered using *the graded lexicographic order* [33, Chapter 5.2]. In Figure 6, the blue square is associated with the zeroth-order gPC component ($d(1)$), the red circles are associated with the first-order gPC components ($\{d(j)\}_{j=2}^{6}$), and so on. Let us focus on three gPC components associated only with ξ_1, $\{\psi_2(\xi) = \ell_1(\xi_1), \psi_7(\xi) = \ell_2(\xi_1), \psi_{22}(\xi) = \ell_3(\xi_1)\}$, where, for $j = 2, 7, 22$, the multi-indices are $d(2) = (1, 0, 0, 0, 0)$, $d(7) = (2, 0, 0, 0, 0)$, and $d(22) = (3, 0, 0, 0, 0)$. Figure 6 confirms that the gPC components $\{\psi_2(\xi), \psi_7(\xi), \psi_{22}(\xi)\}$

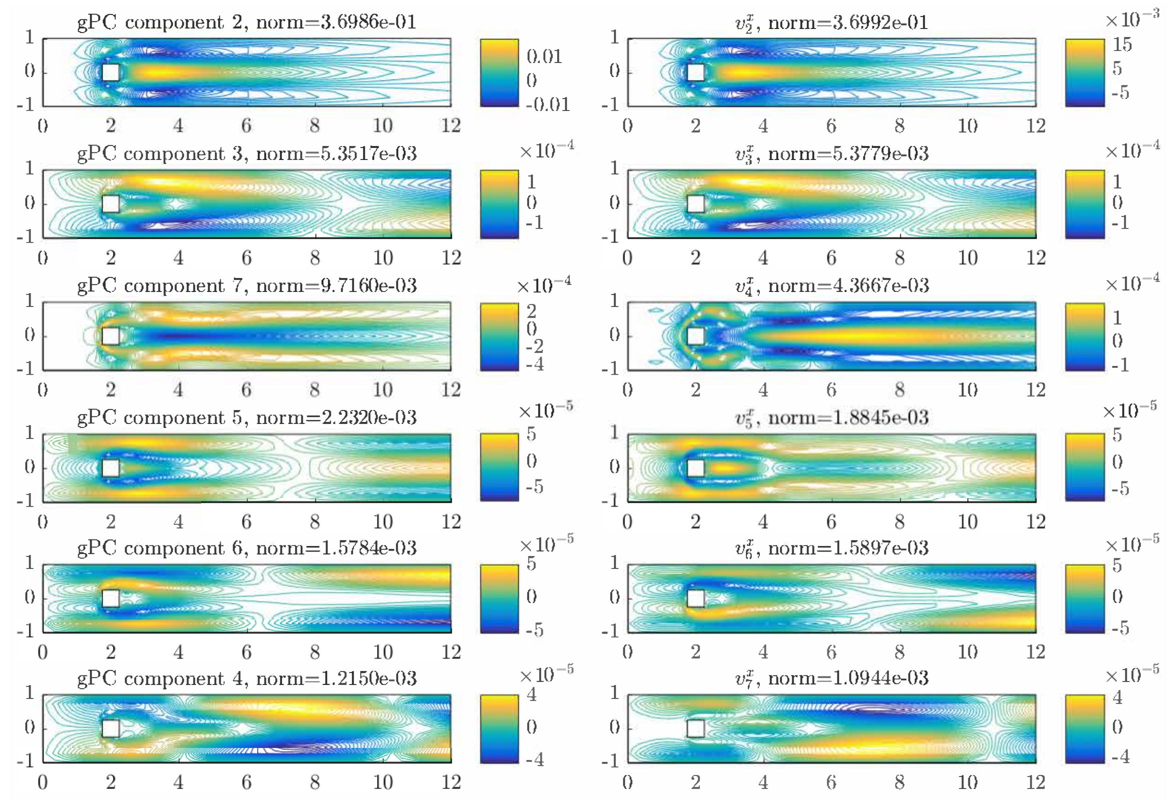


Figure 7. *Plots of coefficients of gPC components* 2–7 *of* $u^x(x,\xi)$ *(left) and coefficients* v_i^x *of* $\theta_i^x(\xi)$ *for* $i = 2,\dots,7$ *(right) for* $\mathrm{Re}_0 = 100$*,* $CoV = 1\%$*, and* $l = 32$*.*

associated with the input variable ξ_1, which has the highest impact on the input (34), also have the highest impact in the output.

We continue the examination of this data in Figure 7 (left), which shows two-dimensional mesh plots of the second through seventh columns of U^x. These images show that these coefficients are either symmetric with respect to the horizontal axis or reflectionally symmetric (equal in magnitude but of opposite sign), and (as also revealed in Figure 6) they tend to have smaller values as the index j is increased.

We now look more closely at features of the factors of the low-rank approximate solution and compare these with those of the (unfactored) full-rank solution. In the low-rank format, the discrete solution is represented using factors $(\Phi^T(x)V^x)(\Psi^T(\xi)W^x)^T$. Let us introduce a concise notation for the approximation of $u^x(x,\xi)$,

$$u^x(x,\xi) \approx Z^x_{\alpha_{\bar{u}^x}}(x)^T \Theta^x_{\alpha_{\bar{u}^x}}(\xi) = \sum_{i=1}^{\alpha_{\bar{u}^x}} \zeta_i^x(x)\theta_i^x(\xi),$$

where $Z^x_{\alpha_{\bar{u}^x}}(x) = [\zeta_1^x(x),\dots,\zeta^x_{\alpha_{\bar{u}^x}}(x)]$ and $\Theta^x_{\alpha_{\bar{u}^x}}(\xi) = [\theta_1^x(\xi),\dots,\theta^x_{\alpha_{\bar{u}^x}}(\xi)]$ with $\zeta_i^x(x) = [\Phi^T(x)V^x]_i$ and $\theta_i^x(\xi) = [(\Psi^T(\xi)W^x)]_i$ for $i = 1,\dots,\alpha_{\bar{u}^x}$. Figure 7 (right) shows the coefficients of the ith random variable $\theta_i^x(\xi)$. As opposed to the gPC coefficients of the

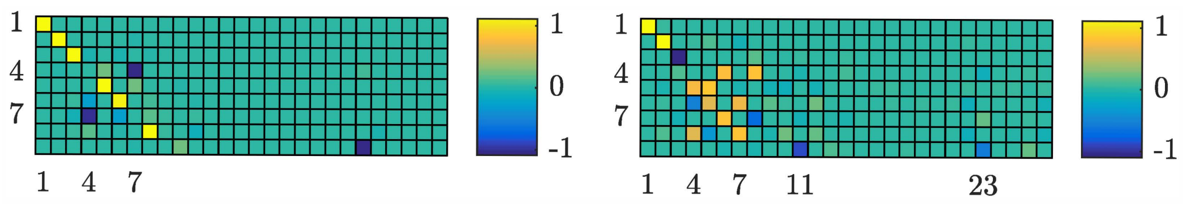


Figure 8. *Heat maps of* $(W^x)^T$ *for* $CoV = 1\%$ *and* $l = 32$ *(left) and* $CoV = 5\%$ *and* $l = 1$ *(right).*

full-rank solution, the norms of the coefficients v_i^x of $\theta_i^x(\xi)$ decrease monotonically as the index i increases. This is a consequence of the fact that the ordering for $\theta_i^x(\xi)$ comes from the singular values of U^x. Figure 7 (right) shows the second–seventh columns of V^x. Figure 7 shows that the coefficients v_i^x of $\theta_i^x(\xi)$ are comparable to the coefficients u_i^x of the gPC components. Each pair of components in the following parenthesized collection is similar: (u_2, v_2), (u_3, v_3), $(u_7, -v_4)$, $(u_4, -v_7)$, (u_5, v_5), and $(u_6, -v_6)$.

Figure 8 shows a "heat map" of $(W^x)^T$, where values of the elements in W^x are represented as colors and the map shows that very few elements of W_i^x are dominant and a sum of those elements is close to 1. Using the fact that $\theta_i^x(\xi) = \Psi^T(\xi)w_i^x$, the figure shows how $\theta_i^x(\xi)$ is represented in terms of $\Psi(\xi)$. For the case $CoV = 1\%$ and $l = 32$ shown on the left, W^x tends to act as a permutation matrix. In particular, many dominant elements of W^x are located in its diagonal, with a value approximately 1, which results in $\theta_i^x(\xi) \approx \pm\psi_i(\xi)$ (e.g., $i = 1, 2, 3, 5, 6, 8$). For the fourth column of W^x, the most dominant entry is the seventh with a value close to -1, which results in $\theta_4^x(\xi) \approx -\psi_7(\xi)$. As shown in Figure 6, $\psi_7(\xi)$ has a larger contribution than most other gPC components, and $\theta_4^x(\xi)$, which consists mainly of $\psi_7(\xi)$, has a smaller index in the new solution representation. For the case of $CoV = 5\%$ and $l = 1$ depicted on the right side of Figure 8, the dominant modes determined from the SVD, i.e., $\theta_i^x(\xi)$ for small i, tend to correspond to multiple gPC components with larger contributions; for example, $\theta_4^x(\xi) \approx [W^x]_{64}\psi_6(\xi) + [W^x]_{84}\psi_8(\xi)$.

5.3. Computational costs. In this section, we assess the costs of the low-rank inexact nonlinear iteration under various experimental settings: two types of covariance kernels (35), varying CoV (36), and varying Re_0. In addition, for various values of these quantities, we investigate the decay of the eigenvalues λ_i used to define the random viscosity (34) and their influence on the rank of solutions. All numerical experiments are performed on an Intel 3.1 GHz i7 CPU, 16 GB RAM, using MATLAB R2016b, and costs are measured in terms of CPU wall time (in seconds). For larger CoV and Re_0, we found the solver to be more effective using the slightly smaller truncation tolerance $\rho_{\mathrm{trunc}} = 10^{-1.5}$ and used this choice for all experiments described below. (Other adaptive tolerances are those shown as in Table 1.) This change had little impact on results for small CoV and Re_0.

Figure 9 shows the 50 largest eigenvalues λ_i of the eigenvalue problems associated with the SE covariance kernel and the AE covariance kernel (35) with $l = 8$, $CoV = 1\%$, and $\mathrm{Re}_0 = 100$. The eigenvalues of the SE covariance kernel decay much more rapidly than those of the AE covariance kernel. Because we choose a fixed number of terms $n_\nu = 5$, the random viscosity derived from the SE covariance kernel retains a smaller variance.

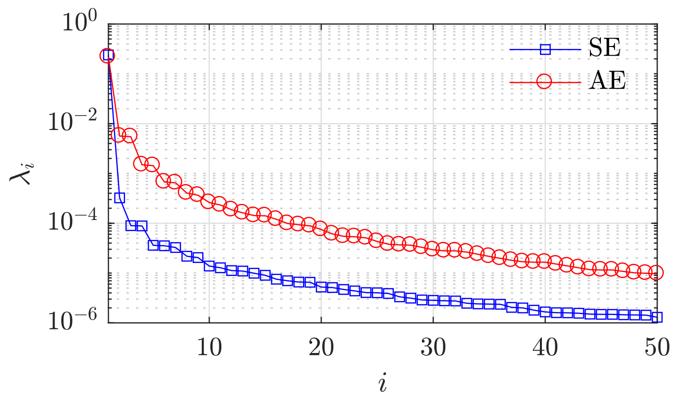
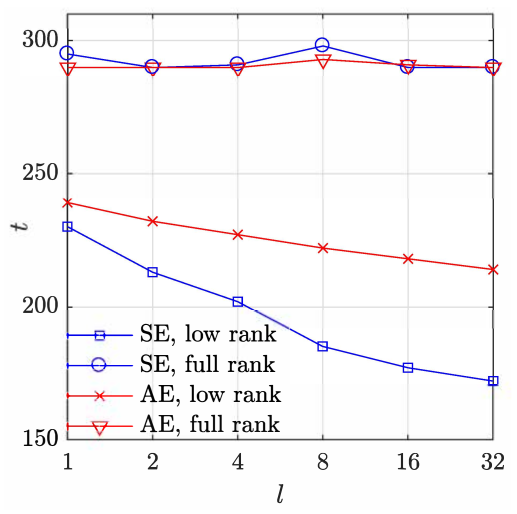
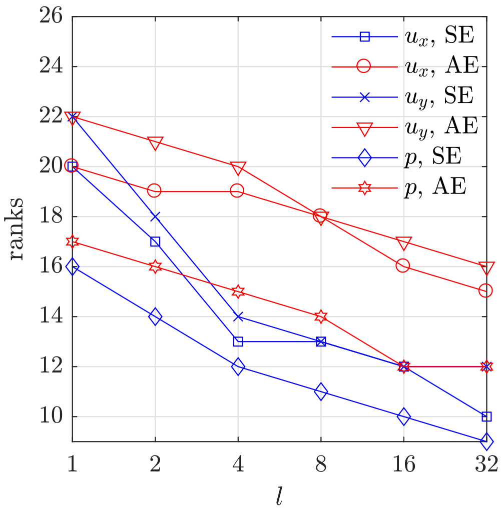


Figure 9. *Eigenvalue decay of the AE and the SE covariance kernels.*



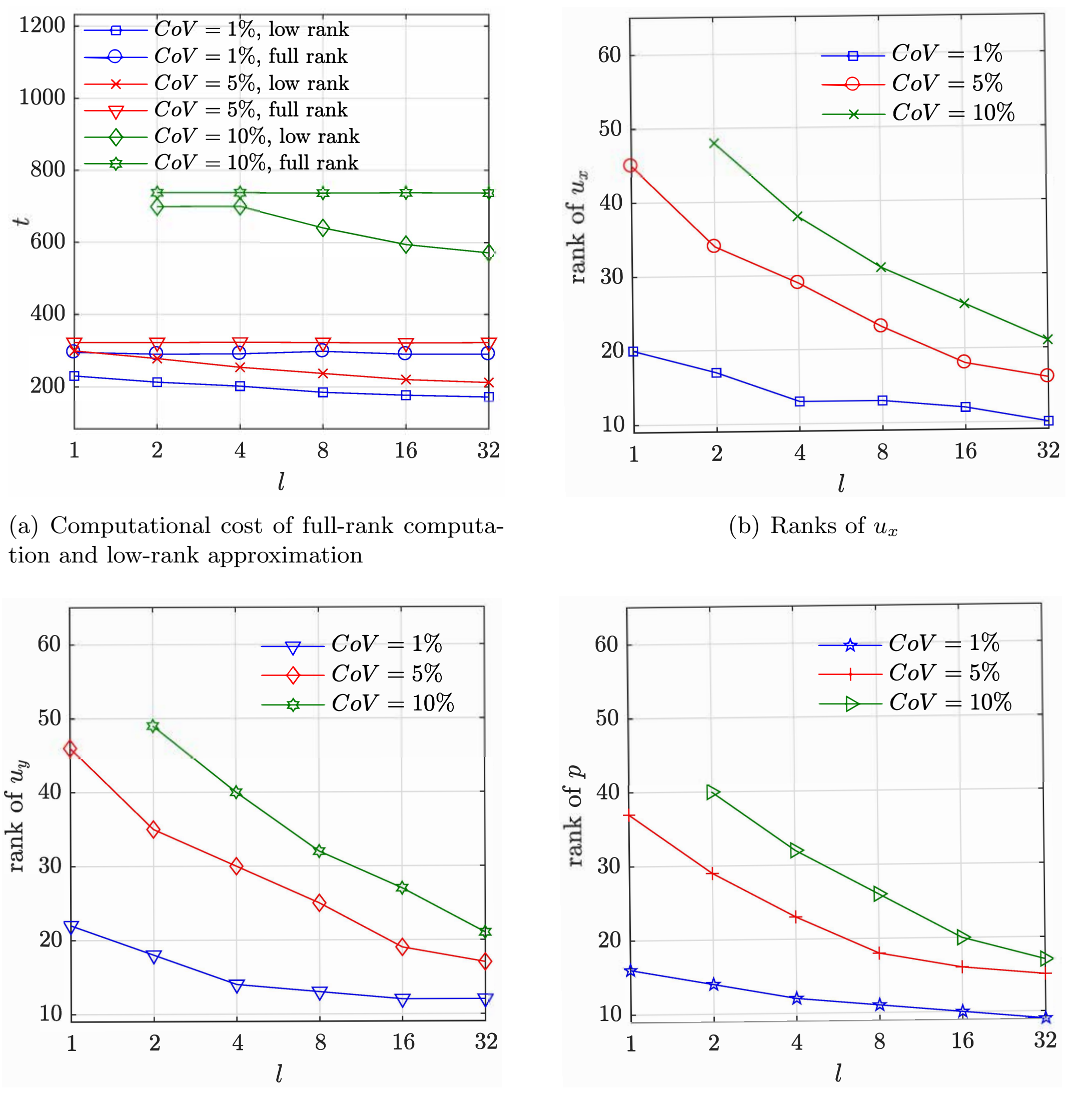
(a) Computational cost of full-rank computation and low-rank approximation



(b) Ranks of the low-rank approximate solutions

Figure 10. *Computational costs and ranks for varying correlation lengths with SE and AE covariance kernel.*

Figure 10(a) shows the computational costs (in seconds) needed for computing the full-rank solutions and the low-rank approximate solutions using the inexact nonlinear iteration for the two covariance kernels and a set of correlation lengths, $l = \{1, 2, 4, 8, 16, 32\}$. Figure 10(b) shows the ranks of the low-rank approximate solutions that satisfy the nonlinear stopping tolerance $\epsilon_{\epsilon_{\text{nl}}} = 10^{-5}$. Again, $\text{Re}_0 = 100$ and $CoV = 1\%$. For this benchmark problem, 4 Picard iterations and 1 Newton iteration are enough to generate a nonlinear iterate satisfying the stopping tolerance ϵ_{nl}. It can be seen from Figure 10(a) that in all cases the use of low-rank methods reduces computational cost. Moreover, as the correlation length becomes larger, the ranks of the corrections and the nonlinear iterates become smaller. As a result, the low-rank method achieves greater computational savings for the problems with larger correlation length.



(a) Computational cost of full-rank computation and low-rank approximation

(b) Ranks of u_x

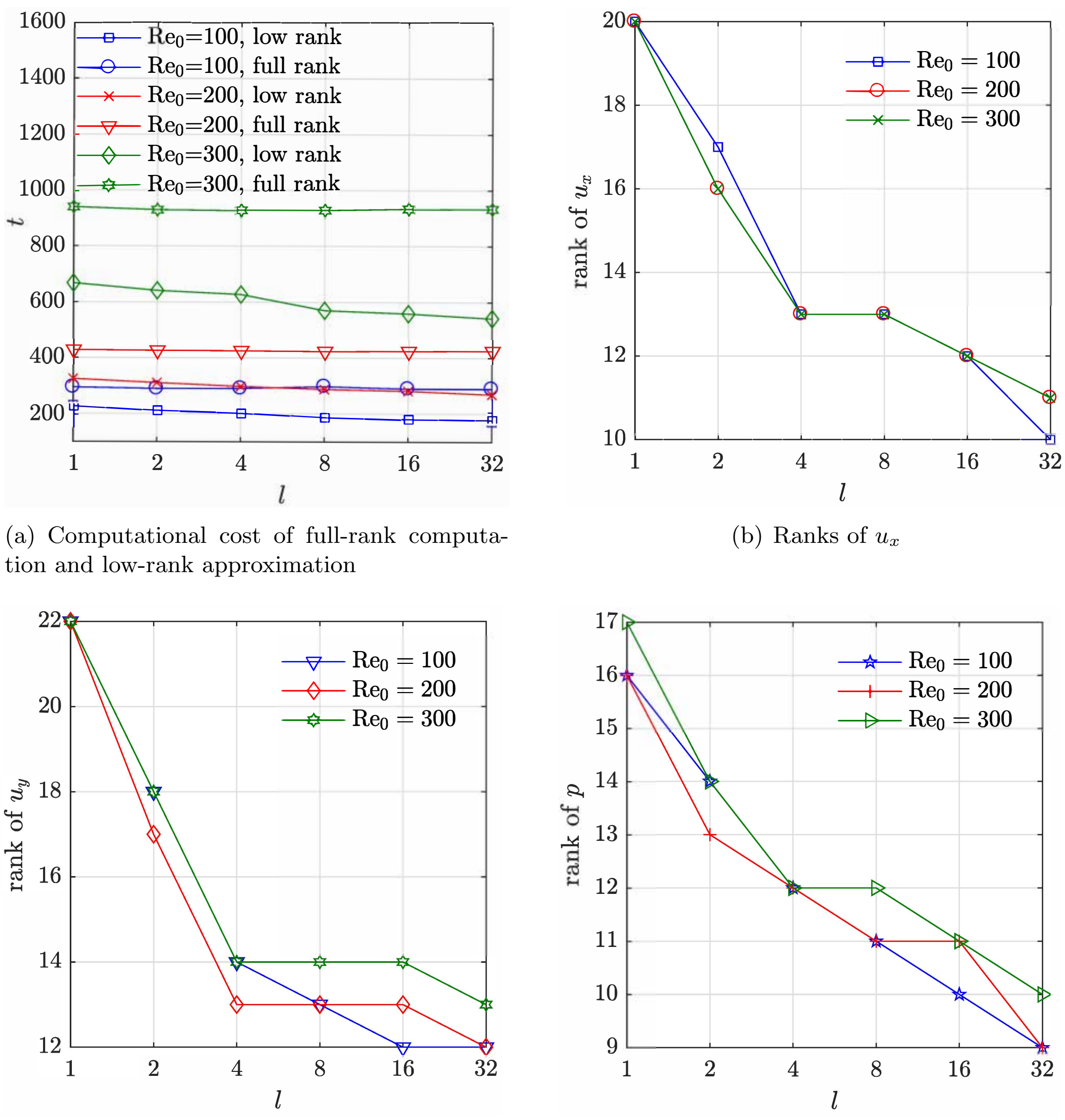
(c) Ranks of u_y

(d) Ranks of p

Figure 11. *Computational costs and ranks for varying correlation lengths and varying* CoV *with* $\mathrm{Re}_0 = 100$.

Next, we examine the performances of the low-rank approximation method for varying CoV, which is defined in (36). In this experiment, we fix the value of $\mathrm{Re}_0 = 100$ and the variance σ_ν is controlled. We consider the SE covariance kernel.

Figure 11 shows the performances of the full-rank and the low-rank methods for varying $CoV = \{1\%, 5\%, 10\%\}$. We use Algorithm 3 with four Picard steps, followed by several Newton steps until convergence. For $CoV = \{1\%, 5\%\}$, one Newton step is required for convergence and, for $CoV = 10\%$, two Newton steps are required. Figure 11(a) shows the computational costs. For $CoV = \{1\%, 5\%\}$, the computational benefits of using the low-rank approximation methods are pronounced whereas, for $CoV = 10\%$, the performances of the two approaches are essentially the same for shorter correlation lengths. Indeed, for higher



(a) Computational cost of full-rank computation and low-rank approximation

(b) Ranks of u_x

(c) Ranks of u_y

(d) Ranks of p

Figure 12. *Computational costs and ranks for varying correlation lengths and varying* Re$_0$.

CoV, the ranks of solutions $\bar{u}$ (see Figures 11(b)–11(d)) as well as updates $\delta\bar{u}^k$ at Newton steps become close to the full rank ($n_\xi = 56$).

Next, we study the benchmark problems with varying mean viscosity with SE covariance kernel and $CoV = 1\%$. As the mean viscosity decreases, Re$_0$ grows, and the nonlinear problem tends to become harder to solve, and for the larger Reynolds numbers Re$_0 = 200$ or 300, we use more Picard steps (5 or 6, respectively) before switching to Newton's method.

Figure 12 shows the performances of the low-rank methods for varying Reynolds number, Re$_0 = \{100, 200, 300\}$. For Re$_0 = 200$, after five Picard steps, one Newton step leads to convergence (and six Picard steps and one Newton step for $\mathrm{Re}_0 = 300$). As the Figures 12(b)–12(d) show, the ranks of the solutions increase slightly as the Reynolds number becomes larger and, thus, for all Re_0 tested here, the low-rank method demonstrates notable computational savings (with $CoV = 1\%$). Note that overall computational costs in Figure 12(a) increase as the Reynolds number becomes larger because (1) the number of nonlinear steps required to converge increases as the Reynolds number increases and (2) to solve each linearized systems, typically more lrGMRES cycles are required for the problems with higher Reynolds number.

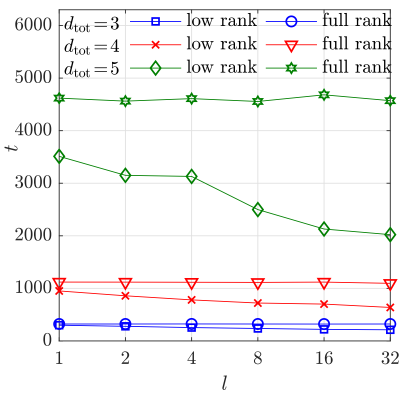
Finally, we study how the total degree d_{tot} of the gPC basis and the number of random variables n_ν effect the performance of the low-rank methods. In these tests, we used the SE covariance kernel, $CoV = 5\%$, and $\mathrm{Re}_0 = 100$; Algorithm 3 with four Picard steps followed by one Newton step led to convergence.

Figure 13 shows performance results for $d_{\mathrm{tot}} = 3, 4,$ and 5 and $n_\nu = 5, 7,$ and 9. For these choices of d_{tot}, we fixed $n_\nu = 5$, which results in $n_\xi = 56, 126,$ and 252. For varying n_ν, we fixed $d_{\mathrm{tot}} = 3$, which results in $n_\xi = 56, 120,$ and 220. Figure 13(a) shows that the benefits of the low-rank methods are enhanced as d_{tot} increases; this is because increasing d_{tot} has small impact on the ranks of the low-rank solutions (Figures 13(c)–13(e)). Similarly the increase in n_ν does not greatly affect the ranks of the solutions (Figures 13(c)–13(e)). (Note in particular that as either d_{tot} or n_ν is increased, the ranks of the solutions increase less dramatically than n_ξ.) The effectiveness of low-rank methods is enhanced as n_ν grows (Figure 13(b)).

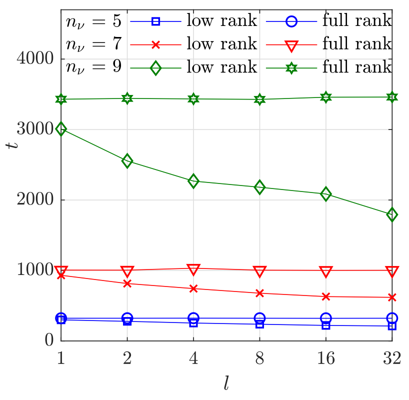
6. Conclusion.

In this study, we have developed the inexact low-rank nonlinear iteration for the solutions of the Navier–Stokes equations with uncertain viscosity in the stochastic Galerkin context. At each step of the nonlinear iteration, the solution of the linear system is inexpensively approximated in low-rank format using the tensor variant of the GMRES method. We examined the effect of the truncation on the accuracy of the low-rank approximate solutions by comparing those solutions to the ones computed using exact, inexact nonlinear iterations in full rank and the Monte Carlo method. Then we explored the efficiency of the proposed method with a set of benchmark problems for various settings of uncertain viscosity. The numerical experiments demonstrated that the low-rank nonlinear iteration achieved significant computational savings for the problems with smaller CoV and larger correlation lengths. The experiments also showed that the mean Reynolds number does not significantly affect the rank of the solution and the low-rank nonlinear iteration achieves computational savings for varying Reynolds number for small CoV and large correlation lengths. Last, the experiments showed that the low-rank nonlinear iteration performs better for problems with a larger total degree in the gPC expansion and for larger numbers of random variables.

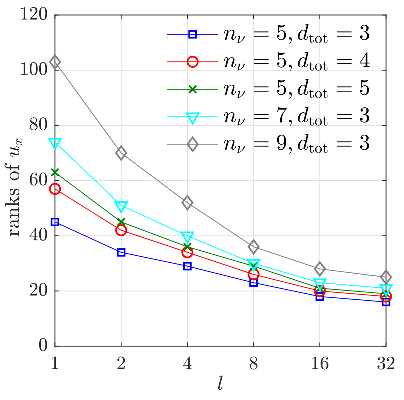
Acknowledgments. This paper describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government. Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, a wholly owned subsidiary of Honeywell International, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525.



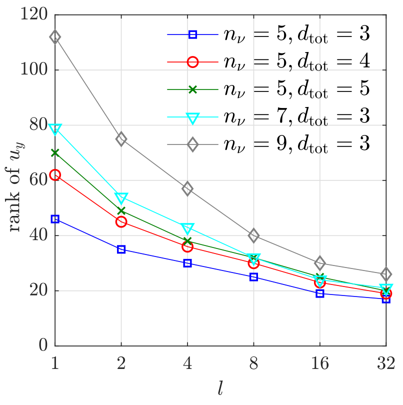
(a) Computational costs for varying total degree $d_{\rm tot}$ in the gPC expansion



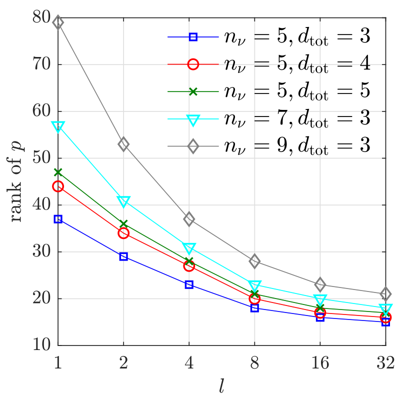
(b) Computational cost for varying number of random variables n_ν



(c) Ranks of u_x



(d) Ranks of u_y



(e) Ranks of p

Figure 13. *Computational costs and ranks for varying total degrees of the gPC expansion d_{tot} and number of random variables n_ν.*

REFERENCES

[1] I. Babuška, R. Tempone, and G. E. Zouraris, *Galerkin finite element approximations of stochastic elliptic partial differential equations*, SIAM J. Numer. Anal., 42 (2004), pp. 800–825.

[2] J. Ballani and L. Grasedyck, *A projection method to solve linear systems in tensor format*, Numer. Linear Algebra Appl., 20 (2013), pp. 27–43.

[3] P. Benner, S. Dolgov, A. Onwunta, and M. Stoll, *Solving Optimal Control Problems Governed by Random Navier-Stokes Equations using Low-Rank Methods*, preprint, arXiv:1703.06097, 2017.

[4] P. Benner, A. Onwunta, and M. Stoll, *Low-rank solution of unsteady diffusion equations with stochastic coefficients*, SIAM/ASA J. Uncertain. Quantif., 3 (2015), pp. 622–649.

[5] R. S. Dembo, S. C. Eisenstat, and T. Steihaug, *Inexact Newton methods*, SIAM J. Numer. Anal., 19 (1982), pp. 400–408.

[6] S. Dolgov, B. N. Khoromskij, A. Litvinenko, and H. G. Matthies, *Polynomial chaos expansion of random coefficients and the solution of stochastic partial differential equations in the tensor train format*, SIAM/ASA J. Uncertain. Quantif., 3 (2015), pp. 1109–1135.

[7] H. C. Elman and D. Furnival, *Solving the stochastic steady-state diffusion problem using multigrid*, IMA J. Numer. Anal., 27 (2007), pp. 675–688.

[8] H. C. Elman, A. Ramage, and D. J. Silvester, *IFISS: A computational laboratory for investigating incompressible flow problems*, SIAM Rev., 56 (2014), pp. 261–273.

[9] H. C. Elman, D. J. Silvester, and A. J. Wathen, *Finite Elements and Fast Iterative Solvers: with Applications in Incompressible Fluid Dynamics*, Oxford University Press, Oxford, 2014.

[10] H. C. Elman and T. Su, *A low-rank multigrid method for the stochastic steady-state diffusion problem*, SIAM J. Matrix Anal. Appl., 39 (2018), pp. 492–509.

[11] M. Espig, W. Hackbusch, A. Litvinenko, H. G. Matthies, and P. Wähnert, *Efficient low-rank approximation of the stochastic Galerkin matrix in tensor formats*, Comput. Math. Appl., 67 (2014), pp. 818–829.

[12] R. G. Ghanem and P. D. Spanos, *Stochastic Finite Elements: A Spectral Approach*, Dover, New York, 2003.

[13] C. Kelley, *Iterative Methods for Linear and Nonlinear Equations*, SIAM, Philadelphia, 1995.

[14] D. Kressner and C. Tobler, *Low-rank tensor Krylov subspace methods for parametrized linear systems*, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 1288–1316.

[15] O. P. Le Maître and O. M. Knio, *Spectral Methods for Uncertainty Quantification: With Applications to Computational Fluid Dynamics*, Springer, New York, 2010.

[16] K. Lee and H. C. Elman, *A preconditioned low-rank projection method with a rank-reduction scheme for stochastic partial differential equations*, SIAM J. Sci. Comput., 39 (2017), pp. S828–S850.

[17] K. Lust and D. Roose, *An adaptive Newton–Picard algorithm with subspace iteration for computing periodic solutions*, SIAM J. Sci. Comput., 19 (1998), pp. 1188–1209.

[18] H. G. Matthies and E. Zander, *Solving stochastic systems with low-rank tensor compression*, Linear Algebra Appl., 436 (2012), pp. 3819–3838.

[19] C. D. Meyer, *Matrix Analysis and Applied Linear Algebra*, SIAM, Philadelphia, 2000.

[20] A. Nouy, *A generalized spectral decomposition technique to solve a class of linear stochastic partial differential equations*, Comput. Methods Appl. Mech. Engrg., 196 (2007), pp. 4521–4537.

[21] I. V. Oseledets, *Tensor-train decomposition*, SIAM J. Sci. Comput., 33 (2011), pp. 2295–2317.

[22] C. E. Powell and H. C. Elman, *Block-diagonal preconditioning for spectral stochastic finite-element systems*, IMA J. Numer. Anal., 29 (2009), pp. 350–375.

[23] C. E. Powell and D. J. Silvester, *Preconditioning steady-state Navier–Stokes equations with random data*, SIAM J. Sci. Comput., 34 (2012), pp. A2482–A2506.

[24] E. Rosseel and S. Vandewalle, *Iterative solvers for the stochastic finite element method*, SIAM J. Sci. Comput., 32 (2010), pp. 372–397.

[25] Y. Saad and M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. and Stat. Comput., 7 (1986), pp. 856–869.

[26] H. V. Sande, T. Boonen, H. De Gersem, F. Henrotte, and K. Hameyer, *Accelerating non-linear time-harmonic problems by a hybrid Picard–Newton approach*, in Proceedings of the 10th International IGTE Symposium on Numerical Field Calculation in Electrical Engineering, 2002, pp. 342–347.

[27] D. J. Silvester, H. C. Elman, and A. Ramage, *Incompressible Flow and Iterative Solver Software (IFISS) Version* 3.4, 2015, http://www.manchester.ac.uk/ifiss/.

[28] B. Sousedík and H. C. Elman, *Stochastic Galerkin methods for the steady-state Navier–Stokes equations*, J. Comput. Phys., 316 (2016), pp. 435–452.

[29] B. Sousedík, R. G. Ghanem, and E. T. Phipps, *Hierarchical Schur complement preconditioner for the stochastic Galerkin finite element methods*, Numer. Linear Algebra Appl., 21 (2014), pp. 136–151.

[30] M. Stoll and T. Breiten, *A low-rank in time approach to PDE-constrained optimization*, SIAM J. Sci. Comput., 37 (2015), pp. B1–B29.

[31] L. Tamellini, O. Le Maître, and A. Nouy, *Model reduction based on proper generalized decomposition for the stochastic steady incompressible Navier–Stokes equations*, SIAM J. Sci. Comput., 36 (2014), pp. A1089–A1117.

[32] E. Ullmann, H. C. Elman, and O. G. Ernst, *Efficient iterative solvers for stochastic Galerkin discretizations of log-transformed random diffusion problems*, SIAM J. Sci. Comput., 34 (2012), pp. A659–A682.

[33] D. Xiu, *Numerical Methods for Stochastic Computations: A Spectral Method Approach*, Princeton University Press, Princeton, NJ, 2010.

[34] D. Xiu and G. E. Karniadakis, *The Wiener–Askey polynomial chaos for stochastic differential equations*, SIAM J. Sci. Comput., 24 (2002), pp. 619–644.