Primal and dual-primal iterative substructuring methods of stochastic PDEs

Waad Subber and Abhijit Sarkar
Department of Civil and Environmental Engineering, Carleton University, Ottawa, Ontario K1S5B6, Canada
E-mail: wsubber@connect.carleton.ca, abhijit_sarkar@carleton.ca

Abstract. A novel non-overlapping domain decomposition method is proposed to solve the large-scale linear system arising from the finite element discretization of stochastic partial differential equations (SPDEs). The methodology is based on a Schur complement based geometric decomposition and an orthogonal decomposition and projection of the stochastic processes using Polynomial Chaos expansion. The algorithm offers a direct approach to formulate a two-level scalable preconditioner. The proposed preconditioner strictly enforces the continuity condition on the corner nodes of the interface boundary, while weakly satisfying the continuity condition over the remaining interface nodes. This approach relates to a primal version of an iterative substructuring method. Next, a Lagrange multiplier based dual-primal domain decomposition method is introduced in the context of SPDEs. In the dual-primal method the continuity condition on the corner nodes is strictly satisfied while Lagrange multipliers are used to enforce continuity on the remaining part of the interface boundary. For numerical illustrations, a two dimensional elliptic SPDE with non-Gaussian random coefficients is considered. The numerical results demonstrate the scalability of these algorithms with respect to the mesh size, subdomain size, fixed problem size per subdomain, order of Polynomial Chaos expansion and level of uncertainty in the input parameters. The numerical experiments are performed on a Linux cluster using MPI and PETSc libraries.

1. Introduction

A domain decomposition method of SPDEs is introduced in [1, 2] to quantify uncertainty in large-scale linear systems. The methodology is based on a Schur complement based geometric decomposition and an orthogonal decomposition and projection of the stochastic processes. A parallel preconditioned conjugate gradient method (PCGM) is adopted in [3] to solve the interface problem without explicitly constructing the Schur complement system. The parallel performance of the algorithms is demonstrated using a lumped preconditioner for non-Gaussian systems arising from a hydraulic problem having random soil permeability properties.

A one-level Neumann-Neumann domain decomposition preconditioner for SPDEs is introduced in [4] in order to enhance the performance of the parallel PCGM iterative solver in [3]. The implementation of the algorithm requires a local solve of a stochastic Dirichlet problem followed by a local solve of a stochastic Neumann problem in each iteration of the PCGM solver. The multilevel sparsity structure of the coefficient matrices of the stochastic system, namely (a) the sparsity structure due to the finite element discretization and (b) the block sparsity structure due to the Polynomial Chaos expansion is exploited for computational efficiency. The one-level
Neumann-Neumann preconditioner in [4] demonstrates a good (strong and weak) scalability for the moderate range of CPUs considered.

In this paper we first describe a primal version of iterative substructuring methods for the solution of the large-scale linear system arising from stochastic finite element method. The algorithm offers a straightforward approach to formulate a two-level scalable preconditioner. The continuity condition is strictly enforced on the corner nodes (nodes shared among more than two subdomains including the nodes at the ends of the interface edges). For the remaining part of the interface boundary, the continuity condition is satisfied weakly (in an average sense). Note that the continuity of the solution field across the entire interface boundary is eventually satisfied at the convergence of the iterations. This approach naturally leads to a coarse grid which connects the subdomains globally via the corner nodes. The coarse grid provides a mechanism to propagate information globally which makes the algorithm scalable with respect to subdomain size. In the second part of the paper, a dual-primal iterative substructuring method is introduced for SPDEs, which maybe viewed as an extension of the Dual-Primal Finite Element Tearing and Interconnecting method (FETI-DP) [5] in the context of SPDEs. In this approach, the continuity condition on the corner nodes is strictly satisfied by partial assembly while Lagrange multipliers are used to enforce continuity on the remaining part of the interface boundary. A system of Lagrange multiplier (also called the dual variable) is solved iteratively using PCGM method equipped with Dirichlet preconditioner. PETSc [6] and MPI [7] libraries are used for efficient parallel implementation of the primal and dual-primal algorithms. The graph partitioning tool METIS [8] is used for optimal decomposition of the finite element mesh for load balancing and minimum interprocessor communication. The parallel performance of the algorithms is studied for a two dimensional stochastic elliptic PDE with non-Gaussian random coefficients. The numerical experiments are performed using a Linux cluster.

2. Uncertainty representation by stochastic processes

This section provides a brief review of the theories of stochastic processes relevant to the subsequent developments of the paper [9, 1]. We assume the data induces a representation of the model parameters as random variables and processes which span the Hilbert space \( H_G \). A set of basis functions \( \{\xi_i\} \) is identified to characterize this space using Karhunen-Loeve expansion. The state of the system resides in the Hilbert space \( H_L \) with basis functions \( \{\Psi_i\} \) being identified with the Polynomial Chaos expansion (PC). The Karhunen-Loeve expansion of a stochastic process \( \alpha(x, \theta) \) is based on the spectral expansion of its covariance function \( R_{\alpha\alpha}(x, y) \). The expansion takes the following form

\[
\alpha(x, \theta) = \bar{\alpha}(x) + \sum_{i=1}^{\infty} \sqrt{\lambda_i} \xi_i(\theta) \phi_i(x),
\]

where \( \bar{\alpha}(x) \) is the mean of the stochastic process, \( \theta \) represents the random dimension, and \( \xi_i(\theta) \) is a set of uncorrelated (but not generally independent for non-Gaussian processes) random variables, \( \phi_i(x) \) are the eigenfunctions and \( \lambda_i \) are the eigenvalues of the covariance kernel which can be obtained as the solution to the following integral equation

\[
\int_{D} R_{\alpha\alpha}(x, y) \phi_i(y) dy = \lambda_i \phi_i(x).
\]

where \( D \) denotes the spatial dimension over which the process is defined. The covariance function of the solution process is not known a priori, and hence the Karhunen-Loeve expansion cannot be used to represent it. Therefore, a generic basis, that is complete in the space of all second-order random variables will be identified and used in the approximation process. Since the solution
process is a function of the material properties, nodal solution variables, denoted by \( u(\theta) \) can be formally expressed as some nonlinear functional of the set \( \xi_i(\theta) \) used to represent the material stochasticity. It has been shown that this functional dependence can be expanded in terms of polynomials in Gaussian random variables, namely Polynomial Chaos [9] as

\[
u(\theta) = \sum_{j=0}^{N} \Psi_j(\theta) u_j.
\]

These polynomials are orthogonal in the sense that their inner product \( \langle \Psi_j \Psi_k \rangle \), defined as the statistical average of their product, is equal to zero for \( j \neq k \).

### 3. Review of Schur complement based domain decomposition method of SPDEs

Consider an elliptic stochastic PDE defined on a domain \( \Omega \) with a given boundary conditions on \( \partial \Omega \). Finite element discretization of the stochastic PDE leads to the following linear system

\[
A(\theta) u(\theta) = f,
\]

where \( A(\theta) \) is the stiffness matrix with random coefficients, \( u(\theta) \) is the stochastic process representing the response vector and \( f \) is the applied force. For large-scale system, Eq.(1) can be solved efficiently using domain decomposition method [1, 2].

In domain decomposition method, the spatial domain \( \Omega \) is partitioned into \( n_s \) non-overlapping subdomains \( \{ \Omega_s, 1 \leq s \leq n_s \} \) such that

\[
\Omega = \bigcup_{s=1}^{n_s} \Omega_s, \quad \Omega_s \bigcap \Omega_r = 0, \quad s \neq r
\]

and

\[
\Gamma = \bigcup_{s=1}^{n_s} \Gamma_s \text{ where } \Gamma_s = \partial \Omega_s \setminus \partial \Omega
\]

For a typical subdomain \( \Omega_s \) the nodal vector \( u_s^*(\theta) \) is partitioned into a set of interior unknowns \( u_s^I(\theta) \) associated with nodes in the interior of \( \Omega_s \) and interface unknowns \( u_s^\Gamma(\theta) \) associated with nodes that are shared among two or more subdomains, as shown in Fig.(1).

Consequently, the subdomain equilibrium equation can be represented as

\[
\begin{bmatrix}
A_{II}^s(\theta) & A_{I\Gamma}^s(\theta) \\
A_{I\Gamma}^s(\theta) & A_{\Gamma\Gamma}^s(\theta)
\end{bmatrix}
\begin{bmatrix}
u^I_s(\theta) \\
u^\Gamma_s(\theta)
\end{bmatrix}
= \begin{bmatrix} f^I_s \\ f^\Gamma_s \end{bmatrix}.
\]

The Polynomial Chaos expansion can be used to represent the uncertainty in the model parameters as

\[
\sum_{i=0}^{L} \Psi_i \begin{bmatrix}
A_{II,i} & A_{I\Gamma,i} \\
A_{I\Gamma,i} & A_{\Gamma\Gamma,i}
\end{bmatrix}
\begin{bmatrix}
u^I_s(\theta) \\
u^\Gamma_s(\theta)
\end{bmatrix}
= \begin{bmatrix} f^I_s \\ f^\Gamma_s \end{bmatrix}.
\]

A Boolean restriction operator \( R_s \) of size \( (n^\Gamma_s \times n^\Gamma) \) which maps the global interface vector \( u^\Gamma(\theta) \) to the local interface vector \( u^\Gamma_s(\theta) \) is defined as

\[
u^\Gamma_s(\theta) = R_s u^\Gamma(\theta).
\]
Enforcing the transmission conditions (compatibility and equilibrium) along the interfaces, the global equilibrium equation of the stochastic system can be expressed in the following block linear systems of equations:

\[
\sum_{i=0}^{L} \Psi_i \begin{bmatrix}
A_{I,i} & \cdots & 0 & A_{\Gamma,i} & R_i \\
\vdots & \ddots & \vdots & \vdots & \vdots \\
0 & \cdots & A_{I,i}^{n_s} & A_{\Gamma,i}^{n_s} & R_{I,i}^{n_s} \\
R_{T}^{I} A_{I,i} & \cdots & R_{T}^{I} A_{I,i}^{n_s} & \sum_{s=1}^{n_s} R_{T}^{I} A_{\Gamma,s}^{n_s} R_{s} & \\
\end{bmatrix} \begin{bmatrix}
u_{I,i}^{(\theta)} \\
\vdots \\
\nu_{I,i}^{n_s} (\theta) \\
\nu_{\Gamma,i} (\theta) \\
\end{bmatrix} = \begin{bmatrix}
f_{I}^{1} \\
\vdots \\
f_{I}^{n_s} \\
\end{bmatrix}.
\]  

The solution process can be expanded using the same Polynomial Chaos basis as

\[
\begin{bmatrix}
u_{I,i}^{1} (\theta) \\
\vdots \\
\nu_{I,i}^{n_s} (\theta) \\
\nu_{\Gamma,i} (\theta) \\
\end{bmatrix} = \sum_{j=0}^{N} \Psi_j (\theta) \begin{bmatrix}
u_{I,j}^{1} \\
\vdots \\
\nu_{I,j}^{n_s} \\
\nu_{\Gamma,j} \\
\end{bmatrix}.
\]  

Substituting Eq.(3) into Eq.(2) and performing Galerkin projection to minimize the error over the space spanned by the Polynomial Chaos basis [1], the following coupled deterministic systems of equations is obtained

\[
\begin{bmatrix}
A_{I,i} & \cdots & 0 & A_{\Gamma,i} & R_{I} \\
\vdots & \ddots & \vdots & \vdots & \vdots \\
0 & \cdots & A_{I,i}^{n_s} & A_{\Gamma,i}^{n_s} & R_{I,i}^{n_s} \\
R_{T}^{I} A_{I,i} & \cdots & R_{T}^{I} A_{I,i}^{n_s} & \sum_{s=1}^{n_s} R_{T}^{I} A_{\Gamma,s}^{n_s} R_{s} & \\
\end{bmatrix} \begin{bmatrix}
u_{I,i}^{1} \\
\vdots \\
\nu_{I,i}^{n_s} \\
\nu_{\Gamma,i} \\
\end{bmatrix} = \begin{bmatrix}
f_{I}^{1} \\
\vdots \\
f_{I}^{n_s} \\
\end{bmatrix},
\]  

Figure 1. Partitioning domain nodes into: interior (♦) and interface (■)
where

\[
[A_{\alpha^\beta}]_{jk} = \sum_{i=0}^{L} \langle \Psi_{i} | \Psi_{j} | \Psi_{k} \rangle A_{\alpha^\beta,i},
\]

\[
F_{\alpha^\beta,k} = \langle \Psi_{k} | F_{\alpha} \rangle,
\]

\[
U_{m}^{i} = (u_{m}^{i,0}, \ldots, u_{m}^{i,N})^{T},
\]

\[
U_{T}^{i} = (u_{T,0}, \ldots, u_{T,N})^{T},
\]

the subscripts \( \alpha \) and \( \beta \) represent the index \( I \) and \( \Gamma \). The coefficient matrix in Eq.(4) is of order \( n(N+1) \times n(N+1) \) where \( n \) and \( (N+1) \) denote the total number of the degrees of freedom and chaos coefficients respectively. The stochastic counterpart of the restriction operator in Eq.(4) takes the following form

\[
R_{s} = \text{blockdiag}(R_{0}^{s}, \ldots, R_{N}^{s}),
\]

where \( (R_{0}^{s}, \ldots, R_{N}^{s}) \) are the deterministic restriction operators. In parallel implementation \( R_{s} \) acts as a scatter operator while \( R_{s}^{T} \) acts as a gather operator and are not constructed explicitly.

A block Gaussian elimination reduces the system in Eq.(4) to the following extended Schur complement system for the interface variable \( U_{\Gamma} \):

\[
S U_{\Gamma} = G_{\Gamma},
\]

where the global extended Schur complement matrix \( S \) is given by

\[
S = \sum_{s=1}^{n_{s}} R_{s}^{T} [A_{I_{\Gamma}I_{T}}^{s} - A_{I_{\Gamma}I_{T}}^{s}(A_{I_{T}I_{I}}^{s})^{-1} A_{I_{I}I_{\Gamma}}^{s}] R_{s},
\]

and the corresponding right hand vector \( G_{\Gamma} \) is

\[
G_{\Gamma} = \sum_{s=1}^{n_{s}} R_{s}^{T} [F_{I_{\Gamma}}^{s} - A_{I_{\Gamma}I_{T}}^{s}(A_{I_{T}I_{I}}^{s})^{-1} F_{I_{I}}^{s}].
\]

Once the interface unknowns \( U_{\Gamma} \) is available, the interior unknowns can be obtained concurrently by solving the interior problem on each subdomain as

\[
A_{I_{I}}^{s} U_{I_{I}}^{s} = F_{I_{I}}^{s} - A_{I_{I}I_{\Gamma}}^{s} R_{s} U_{\Gamma}.
\]

4. Solution methods for the extended Schur complement system

Solution methods for linear systems are broadly categorized into direct methods and iterative methods. The direct methods, generally, are based on sparse Gaussian elimination technique and are popular for their robustness. However, they are expensive in computation time and memory requirements and therefore cannot be applied to the solution of large-scale linear systems [10]. On the other hand, the iterative methods generate a sequences of approximate solutions which converge to the true solutions. In the iterative methods the main arithmetic operation is the matrix-vector multiplication. Therefore the linear system itself need not be constructed explicitly and only a procedure for matrix-vector product is required. This property makes iterative methods more suitable to parallel processing than direct methods.
4.1. Preconditioned Conjugate Gradient Method (PCGM)

Non-overlapping domain decomposition method or iterative substructuring can be viewed as a preconditioned iterative method to solve the Schur complement system of the form \[11\]

\[ SU_\Gamma = G_\Gamma. \]

For symmetric positive-definite system such as Schur complement system, the Conjugate Gradient Method (CGM) is generally used. The performance of CGM mainly depends on the spectrum of the coefficient matrix. However, the rate of convergence of the iterative method can generally be improved by transforming the original system into an equivalent system that has better spectral properties (i.e. lower condition number \(\kappa(S)\)) of the coefficient matrix. This transformation is called preconditioning and the matrix used in the transformation is called the preconditioner. In other words, the transformed linear system becomes:

\[ M^{-1} S U_\Gamma = M^{-1} G_\Gamma, \]

In general, \(\kappa(M^{-1} S)\) is much smaller than \(\kappa(S)\) and the eigenvalues of \(M^{-1} S\) are clustered near one. This procedure known as Preconditioned Conjugate Gradient Method (PCGM). In practice, the explicit construction of \(M^{-1}\) is not needed. Instead, for a given vector \(r_\Gamma\), a system of the the following form is solved

\[ MZ = r_\Gamma. \]

The PCGM algorithm to solve the Schur complement system proceeds as follows [10]

**Algorithm 1: The PCGM Algorithm**

1. Initialize \(U_{\Gamma_0} := 0\)
2. Compute \(r_{\Gamma_0} := G_\Gamma - S U_{\Gamma_0}\)
3. Precondition \(Z_0 := M^{-1} r_{\Gamma_0}\)
4. First search direction \(P_0 := Z_0\)
5. Initialize \(\rho_0 := (r_{\Gamma_0}, Z_0)\)
6. For \(j = 0, 1, \cdots\), until convergence Do:
7. \(Q_j := SP_j\)
8. \(\rho_{mp_j} := (Q_j, P_j)\)
9. \(\alpha_j := \rho_j / \rho_{mp_j}\)
10. \(U_{\Gamma_{j+1}} := U_{\Gamma_j} + \alpha_j P_j\)
11. \(r_{\Gamma_{j+1}} := r_{\Gamma_j} - \alpha_j Q_j\)
12. \(Z_{j+1} := M^{-1} r_{\Gamma_{j+1}}\)
13. \(\rho_{j+1} := (r_{\Gamma_{j+1}}, Z_{j+1})\)
14. \(\beta_j := \rho_{j+1} / \rho_j\)
15. \(P_{j+1} := Z_{j+1} + \beta_j P_j\)
16. EndDo

The PCGM algorithm indicates that the main arithmetic operations are calculating the product \(Q = SP\) in step 7 and the preconditioned residual \(Z = M^{-1} r_\Gamma\) in step 12. These operations can be performed in parallel as outlined next.
Given the subdomain Schur complement matrices $S_s$ and a global vector $P$, the matrix-vector product $Q = SP$ can be calculated in parallel as

$$Q = \sum_{s=1}^{n_s} R^T_s S_s R_s P,$$

where $n_s$ is the number of subdomains and $R_s$ and $R^T_s$ are scatter and gather operator respectively. The parallel implementation of this procedure is summarized in Algorithm (2).

**Algorithm 2: Parallel Matrix-Vector Product Procedure**

1. **Input** ($P$)
2. **Scatter** : $P^s = R_s P$
3. **Local operation** : $Q^s = S_s P^s$
4. **Gather** : $Q = \sum_{s=1}^{n_s} R^T_s Q^s$
5. **Output** ($Q$)

The working vectors $P^s$ and $Q^s$ are defined on the subdomain level.

Similarly, the effect of a parallel preconditioner on a residual vector $Z = M^{-1} r_F$ can be computed as

$$Z = \sum_{s=1}^{n_s} R^T_s M_s^{-1} R_s r_F.$$

This procedure is outlined in following algorithm.

**Algorithm 3: Parallel Preconditioner Effect Procedure**

1. **Input** ($r_F$)
2. **Scatter** : $r^s_F = R_s r_F$
3. **Local Solve** : $M_s Z^s = r^s_F$
4. **Gather** : $Z = \sum_{s=1}^{n_s} R^T_s Z^s$
5. **Output** ($Z$)

The local preconditioner $M_s$ and the working vectors $r^s_F$ and $Z^s$ are defined on the subdomain level.

5. **Iterative substructuring methods of SPDEs**

Next sections describe the primal and dual-primal substructuring methods in the context of SPDEs. In the primal method, the interface problem is solved iteratively using PCGM solver equipped with a scalable preconditioner. At each iteration of the iterative solver loop, local problems are solved on each subdomain in parallel. These local problems are used to construct a subdomain level preconditioner. Moreover, a coarse problem is required to propagate information
globally across the subdomains. This global exchange of information leads to a scalable preconditioner. In the dual-primal method, a system of Lagrange multiplier that enforces continuity constraints across the interface boundary, is solved iteratively using PCGM solver. The global coarse problem is already embedded in the operator of the Lagrange multiplier system and therefore a one-level preconditioner such as lumped or Dirichlet is sufficient for scalability. A framework of the primal and dual-primal iterative substructuring methods for SPDEs is detailed next.

6. A primal iterative substructuring method of SPDEs

In order to define local problems over each of the subdomains, we partition the subdomain nodal vector \( \mathbf{u}^s(\theta) \) into a set of interior unknowns \( u^s_i(\theta) \), corner unknowns \( u^s_c(\theta) \) and remaining unknowns \( u^s_r(\theta) \), as schematically shown in Fig.(2).

\[
\begin{bmatrix}
A_{ii}^s(\theta) & A_{ir}^s(\theta) & A_{ic}^s(\theta) \\
A_{ri}^s(\theta) & A_{rr}^s(\theta) & A_{rc}^s(\theta) \\
A_{ci}^s(\theta) & A_{cr}^s(\theta) & A_{cc}^s(\theta)
\end{bmatrix}
\begin{bmatrix}
u^s_i(\theta) \\
u^s_r(\theta) \\
u^s_c(\theta)
\end{bmatrix}
= \begin{bmatrix} f^s_i \\
f^s_r \\
f^s_c \end{bmatrix}.
\]

The Polynomial Chaos representation of uncertain model parameters leads to the following subdomain equilibrium equation

\[
\sum_{l=0}^{L} \Psi_l \begin{bmatrix}
A_{ii,l}^s & A_{ir,l}^s & A_{ic,l}^s \\
A_{ri,l}^s & A_{rr,l}^s & A_{rc,l}^s \\
A_{ci,l}^s & A_{cr,l}^s & A_{cc,l}^s
\end{bmatrix}
\begin{bmatrix}
u^s_i(\theta) \\
u^s_r(\theta) \\
u^s_c(\theta)
\end{bmatrix}
= \begin{bmatrix} f^s_i \\
f^s_r \\
f^s_c \end{bmatrix}.
\]

The solution process is expressed using the same Polynomial Chaos basis as

\[\text{Figure 2. Partitioning domain nodes into: interior (♦), remaining (■) and corner(○) nodes}\]
The equilibrium equation can be expressed as the Schur compliment system in Eq.(5), but now the interface boundary nodes are split into coupled deterministic systems of equations

$$\begin{bmatrix} A^s_{ii} & A^s_{ir} & A^s_{ic} \\ A^s_{ri} & A^s_{rr} & A^s_{rc} \\ A^s_{ci} & A^s_{cr} & A^s_{cc} \end{bmatrix} \begin{bmatrix} U^s_i \\ U^s_r \\ U^s_c \end{bmatrix} = \begin{bmatrix} F^s_i \\ F^s_r \\ F^s_c \end{bmatrix},$$

where

$$[A^s_{\alpha\beta}]_{jk} = \sum_{l=0}^{L} \langle \Psi_l \Psi_j \Psi_k \rangle A^s_{\alpha\beta,l},$$

$$F^s_{\alpha,k} = \langle \Psi_k F^s_\alpha \rangle,$$

$$U^s_\alpha = (u^s_{\alpha,0}, \ldots, u^s_{\alpha,N})^T,$$

the subscripts $\alpha$ and $\beta$ represent the index $i$, $r$ and $c$.

Enforcing the transmission conditions along the boundary interfaces, the subdomain equilibrium equation can be expressed as

$$\begin{bmatrix} \sum_{s=1}^{n_s} B^s_{rT} A^s_{ri} \\ \sum_{s=1}^{n_s} B^s_{cT} A^s_{ci} \\ \sum_{s=1}^{n_s} B^s_{rT} A^s_{rr} \\ \sum_{s=1}^{n_s} B^s_{cT} A^s_{cr} \\ \sum_{s=1}^{n_s} B^s_{rT} A^s_{rc} \\ \sum_{s=1}^{n_s} B^s_{cT} A^s_{cc} \end{bmatrix} \begin{bmatrix} U^s_i \\ U^s_r \\ U^s_c \end{bmatrix} = \begin{bmatrix} \sum_{s=1}^{n_s} B^s_{rT} F^s_r \\ \sum_{s=1}^{n_s} B^s_{cT} F^s_c \end{bmatrix},$$

where $B^s_c$ is a Boolean rectangular matrix that maps the global remaining vector $U_r$ to the local remaining vector $U^s_r$ as

$$U^s_r = B^s_c U_r.$$  \hspace{1cm} (10)

Similarly, the restriction operator $B^s_c$ is a Boolean rectangular matrix that maps the global corner vector $U_c$ to the local corner vector $U^s_c$ as

$$U^s_c = B^s_c U_c.$$  \hspace{1cm} (11)

In parallel implementation both $B^s_c$ and $B^s_r$ act as scatter operators, while $B^s_{rT}$ and $B^s_{cT}$ act as gather operators.

The first block equation in Eq.(9) can be solved for $U^s_i$ in parallel as

$$U^s_i = [A^s_{ii}]^{-1}(F^s_i - A^s_{ir} B^s_r U^s_r - A^s_{ic} B^s_c U^s_c).$$  \hspace{1cm} (12)

Substituting Eq.(12) into Eq.(9) leads to the following condensed system which represents the Schur compliment system in Eq.(5), but now the interface boundary nodes are split into
remaining and corner nodes, as shown schematically in Fig.(3).

\[
\begin{bmatrix}
\sum_{s=1}^{n_s} B_r^s S_{rr}^s B_r^s \\
\sum_{s=1}^{n_s} B_r^s S_{rc}^s B_c^s \\
\sum_{s=1}^{n_s} B_c^s S_{cr}^s B_r^s \\
\sum_{s=1}^{n_s} B_c^s S_{cc}^s B_c^s
\end{bmatrix}
\begin{bmatrix}
U_r \\
U_c
\end{bmatrix} =
\begin{bmatrix}
\sum_{s=1}^{n_s} B_r^s G_r^s \\
\sum_{s=1}^{n_s} B_c^s G_c^s
\end{bmatrix},
\]

(13)

where

\[
S_{\alpha\beta}^s = A_{\alpha\beta}^s - A_{\alpha i}^s [A_{ii}^s]^{-1} A_{\beta j}^s,
\]

\[
G_{\alpha}^s = F_{\alpha}^s - A_{\alpha i}^s [A_{ii}^s]^{-1} F_{i}^s.
\]

\textbf{Figure 3.} The interface boundary nodes split into: remainder (■) and corner(●) nodes

The corner nodal vector \(U_c\) in Eq.(13) is eliminated next to obtain the following (symmetric positive definite) reduced interface problem:

\[
(F_{rr} - F_{rc}[F_{cc}]^{-1}F_{cr})U_r = d_r - F_{rc}[F_{cc}]^{-1}d_c
\]

(14)

where

\[
F_{\alpha\beta} = \sum_{s=1}^{n_s} B_{\alpha}^s S_{\alpha\beta}^s B_{\beta},
\]

\[
d_{\alpha} = \sum_{s=1}^{n_s} B_{\alpha}^s G_{\alpha}^s.
\]

and \(\alpha\) and \(\beta\) denotes subscripts \(r\) and \(c\). The above system can be solved using PCGM with an appropriate preconditioner \(\mathcal{M}^{-1}\) defined in the next section.
6.1. A two-level preconditioner
As mentioned previously, the continuity condition at the corner nodes is enforced strictly while those for the remaining interface boundary nodes is satisfied in a weak sense. This fact is schematically illustrated in Fig.(4) [12]. As the iterations converge, the continuity condition at all interface nodes (both corner and remaining boundary nodes) is satisfied strictly. The assembly of the unknown vector at the corner nodes leads to the following partially assembled Schur complement system:

\[
\begin{bmatrix}
S_{rr} & S_{rc}B_c^s \\
\sum_{s=1}^{n_s} B_c^sT S_{cr}B_r^s & \sum_{s=1}^{n_s} B_c^sT S_{cc}B_c^s
\end{bmatrix}
\begin{bmatrix}
U_r^s \\
U_c
\end{bmatrix} =
\begin{bmatrix}
F_r^s \\
0
\end{bmatrix},
\]

(15)

where

\[F_r^s = D_r^s B_r^s r_j,\]

and \(r_j\) is the residual at the \(j^{th}\) iteration of the PCGM and \(D_r^s\) represents a block diagonal weighting matrix which satisfies the following property:

\[
\sum_{s=1}^{n_s} B_r^sT D_r^s B_r^s = I.
\]

The diagonal entries of each block of \(D_r^s\) are the reciprocal of the number of subdomains that share the interface boundary nodes.

The subdomain level remaining unknown vector \(U_r^s\) can be eliminated in parallel from Eq.(15) as

\[
U_r^s = [S_{rr}]^{-1} (F_r^s - S_{rc}B_c^s U_c)
\]

(16)
Substituting \( U^s_r \) into the second block of Eq.(15) leads to the following coarse problem

\[
F^{*}_{cc} U^c = d^{*}_c, \tag{17}
\]

where

\[
F^{*}_{cc} = \sum_{s=1}^{n_s} B^s_c T (S^s_{cc} - S^s_{cr} [S^s_{rr}]^{-1} S^s_{rc}) B^s_c,
\]

\[
d^{*}_c = - \sum_{s=1}^{n_s} B^s_c T S^s_{cr} [S^s_{rr}]^{-1} F^s_r.
\]

The continuity of the solution field over the remaining interface nodes is satisfied next by averaging the local results as

\[
U_r = \sum_{s=1}^{n_s} B^s_r D^s_r U^s_r.
\]

After some algebraic manipulations, the preconditioner can be expressed as

\[
M^{-1} = \sum_{s=1}^{n_s} B^s_c T D^s_r [S^s_{rr}]^{-1} D^s_r B^s_c + R^T_0 (F^{*}_{cc})^{-1} R_0, \tag{18}
\]

where

\[
R_0 = \sum_{s=1}^{n_s} B^s_c T S^s_{cr} [S^s_{rr}]^{-1} D^s_r B^s_c.
\]

7. A dual-primal iterative substructuring method of SPDEs
In this section, the dual-primal domain decomposition method is introduced in the context of stochastic PDEs. This approach is an extension of FETI-DP [5] for SPDEs. In this approach, the continuity condition at the corner nodes is enforced strictly, and Lagrange multipliers are used to enforce the continuity condition weakly over the remaining interface nodes.

Partial assembly of Eq.(8) leads to the following subdomain equilibrium equation

\[
\begin{bmatrix}
A^s_{ii} & A^s_{ir} & A^s_{ic} \\
A^s_{ri} & A^s_{rr} & A^s_{rc} \\
\sum_{s=1}^{n_s} B^s_c T A^s_{ei} & \sum_{s=1}^{n_s} B^s_c T A^s_{cr} & \sum_{s=1}^{n_s} B^s_c T A^s_{cc} B^s_c
\end{bmatrix}
\begin{bmatrix}
U^s_i \\
U^s_r \\
U^s_c
\end{bmatrix}
= \begin{bmatrix}
F^s_i \\
F^s_r \\
\sum_{s=1}^{n_s} B^s_c T F^s_c
\end{bmatrix}, \tag{19}
\]

where \( B^s_c \) is a Boolean restriction operator that maps the global corner vector \( U_c \) to the local corner vector \( U^s_c \) as

\[
U^s_c = B^s_c U_c.
\]
Eq. (19) can be rewritten in compact form as
\[ \tilde{A}^s \tilde{U}^s = \tilde{F}^s. \] (20)

Let \( B^r \) be a block diagonal signed Boolean continuity matrix defined as
\[ \sum_{s=1}^{n_s} B^r_s U^r_s = 0, \]

Next, the original finite element problem can be reformulated as an equivalent constrained minimization problem as
\[ \frac{1}{2} \tilde{U}^T \tilde{A} \tilde{U} - \tilde{U}^T \tilde{F} \rightarrow \min \]
subject to \( \tilde{B} \tilde{U} = 0, \)

where
\[
\tilde{A} = \begin{bmatrix}
\tilde{A}^1 \\
\vdots \\
\tilde{A}^s \\
\vdots \\
\tilde{A}^{n_s}
\end{bmatrix},
\tilde{U} = \begin{bmatrix}
\tilde{U}^1 \\
\vdots \\
\tilde{U}^s \\
\vdots \\
\tilde{U}^{n_s}
\end{bmatrix},
\tilde{F} = \begin{bmatrix}
\tilde{F}^1 \\
\vdots \\
\tilde{F}^s \\
\vdots \\
\tilde{F}^{n_s}
\end{bmatrix},
\]
\[
\tilde{B} = \begin{bmatrix}
(0 \ B^1_r \ 0), \ldots, (0 \ B^s_r \ 0), \ldots, (0 \ B^{n_s}_r \ 0)
\end{bmatrix}.
\]

By introducing a vector of Lagrange multipliers to enforce the weak compatibility constraint, the saddle point formulation of Eq.(21) can be expressed as:
\[
\mathcal{L}(\tilde{U}, \Lambda) = \frac{1}{2} \tilde{U}^T \tilde{A} \tilde{U} - \tilde{U}^T \tilde{F} + \tilde{U}^T \tilde{B}^T \Lambda. \] (22)

Minimizing Eq.(22) with respect to \( \tilde{U} \) and \( \Lambda \) leads to the following equilibrium system
\[
\begin{bmatrix}
A^s_{ii} & A^s_{ir} & A^s_{ic} B^s_c & 0 \\
A^s_{ri} & A^s_{rr} & A^s_{rc} B^s_c & B^s_r T \\
\sum_{s=1}^{n_s} B^s_c T A^s_c & \sum_{s=1}^{n_s} B^s_c T A^s_{cr} & \sum_{s=1}^{n_s} B^s_c T A^s_{cc} B^s_c & 0 \\
0 & \sum_{s=1}^{n_s} B^s_r & 0 & 0
\end{bmatrix}
\begin{bmatrix}
\tilde{U}^s_i \\
\tilde{U}^s_r \\
\tilde{U}^s_c \\
\tilde{U}^s \end{bmatrix}
= \begin{bmatrix}
\tilde{F}^s_i \\
\tilde{F}^s_r \\
\tilde{F}^s_c \\
0
\end{bmatrix}, \] (23)

where
\[ \Lambda = \begin{bmatrix}
\lambda_0 \\
\vdots \\
\lambda_N
\end{bmatrix}. \]
Fig. 5. Lagrange multipliers are the forces required to connect the tore interface boundary

and $\lambda_j$ is the nodal force required to satisfy compatibility at the remaining interface nodes as shown schematically in Fig. (5).

Eliminating the interior unknowns $\mathbf{U}^s_i$ from Eq. (23) as

$$
\mathbf{U}^s_i = [\mathbf{A}^s_{ii}]^{-1} (\mathbf{F}^s_i - \mathbf{A}^s_{ir}\mathbf{U}^s_r - \mathbf{A}^s_{ic}\mathbf{B}^s_c \mathbf{U}^c)
$$

Substituting Eq. (24) into Eq. (23), leads to

$$
\begin{bmatrix}
\mathbf{S}^s_{rr} & \mathbf{S}^s_{rc} & \mathbf{B}^s_r \\
\mathbf{S}^s_{cr} & \mathbf{B}^s_c T & 0 \\
\mathbf{B}^s_c & 0 & 0
\end{bmatrix}
\begin{bmatrix}
\mathbf{U}^s_r \\
\mathbf{U}^s_c \\
\Lambda
\end{bmatrix}
= 
\begin{bmatrix}
\mathbf{G}^s_r \\
\mathbf{G}^s_c \\
0
\end{bmatrix},
$$

(25)

where

$$
\mathbf{S}^s_{\alpha\beta} = \mathbf{A}^s_{\alpha\beta} - \mathbf{A}^s_{\alpha i} [\mathbf{A}^s_{ii}]^{-1} \mathbf{A}^s_{i\beta},
$$

$$
\mathbf{G}^s_{\alpha} = \mathbf{F}^s_{\alpha} - \mathbf{A}^s_{\alpha i} [\mathbf{A}^s_{ii}]^{-1} \mathbf{F}^s_{i}.
$$

The subdomain level remaining unknown vector $\mathbf{U}^s_r$ can be obtained in parallel from Eq. (25) as

$$
\mathbf{U}^s_r = [\mathbf{S}^s_{rr}]^{-1} (\mathbf{G}^s_r - \mathbf{S}^s_{rc} \mathbf{B}^s_c \mathbf{U}^c - \mathbf{B}^s_r T \Lambda)
$$

(26)

Substituting Eq. (26) into Eq. (25) leads to

$$
\begin{bmatrix}
\mathbf{F}^s_{cc} & -\mathbf{F}^s_{cr} \\
\mathbf{F}^s_{rc} & \mathbf{F}^s_{rr}
\end{bmatrix}
\begin{bmatrix}
\mathbf{U}^c \\
\Lambda
\end{bmatrix}
= 
\begin{bmatrix}
\mathbf{d}^c_r \\
\mathbf{d}^r_r
\end{bmatrix},
$$

(27)

where
Solving for $U_c$ from Eq.(27) gives the following coarse problem

$$\bar{F}_{cc} U_c = (\bar{d}_c + \bar{F}_{cr} \Lambda)$$  \hspace{1cm} (28)$$

Substituting $U_c$ into Eq.(27) leads to the following symmetric positive-definite Lagrange multiplier system

$$\begin{eqnarray*}
(\bar{F}_{rr} + \bar{F}_{rc}[\bar{F}_{cc}]^{-1}\bar{F}_{cr}) \Lambda & = & \bar{d}_r - \bar{F}_{rc}[\bar{F}_{cc}]^{-1}\bar{d}_c \hspace{1cm} (29)
\end{eqnarray*}$$

Eq.(29) is solved using PCGM with a Dirichlet precondtioner defined as

$$\bar{M} = \sum_{s=1}^{n_s} B^s_r D^s r S^s_{rr} D^s r B^s_r \hspace{1cm} (30)$$

8. Connection between the methods

The explicit forms of the coarse problem operators for the primal preconditioner in Eq.(17) and for the dual-primal operator in Eq.(28) are the same, and can be expressed as

$$F^*_{cc} = \bar{F}_{cc} = \sum_{s=1}^{n_s} B^s_c T (S^s_{cc} - S^s_{cr} [S^s_{rr}]^{-1} S^s_{rc}) B^s_c \hspace{1cm} (27)$$

Furthermore, the algebraic form of the primal preconditioner in Eq.(18) can be re-casted as

$$\bar{M}^{-1} = \sum_{s=1}^{n_s} B^s_c T D^s r [S^s_{rr}]^{-1} D^s r B^s_c + \sum_{s=1}^{n_s} B^s_c T D^s r [S^s_{rr}]^{-1} S^s r B^s_c \sum_{s=1}^{n_s} B^s_c T (S^s_{cc} - S^s_{cr} [S^s_{rr}]^{-1} S^s_{rc}) B^s_c \hspace{1cm} (28)$$

which has the same form of the dual-primal operator in Eq.(29).
9. Parallel implementation

In this section we give an outline on parallel implementation of PCGM to solve the primal Eq.(14) and dual-primal Eq.(29) interface problems. As mentioned previously, in PCGM, the coefficient matrix need not be constructed explicitly as only its effect on a vector is required. This matrix-vector product can be obtained concurrently by solving subdomain level problems (Dirichlet and Neumann) and a global level coarse problem.

9.1. Primal method

In this subsection we give a brief description of parallel implementation of Algorithm (1) to solve the primal interface problem in Eq.(14).

For the \( j \)th iteration of Algorithm (1), the matrix-vector product in step 7 defined as

\[
Q_j = (F_{rr} - F_{rc}[F_{cc}]^{-1}F_{cr})P_j,
\]

can be computed using the following algorithm:

**Algorithm 4: Parallel Matrix-Vector Product for Primal Method**

1. **Input** \((P)\)
2. **Scatter** : \( P^s = B^s_r P \)
3. **Compute** : \( v_1^s = S^s_{cr} P^s \)
4. **Gather** : \( V_1 = \sum_{s=1}^{n_s} B^s_r T v_1^s \)
5. **Global Solve** : \( F_{cc} V_2 = V_1 \)
6. **Scatter** : \( v_2^s = B^s_c V_2 \)
7. **Compute** : \( v_3^s = S^s_{rc} v_2^s \)
8. **Update** : \( Q^s = S^s_{rr} P^s - v_3^s \)
9. **Gather** : \( Q = \sum_{s=1}^{n_s} B^s_r T Q^s \)
10. **Output** \((Q)\)

Multiplication of Schur complement matrix by a vector in step 3, step 7 and step 8 in Algorithm (4) is computed by solving a corresponding Dirichlet problem as

\[
v_\alpha^s = S^s_{\alpha\beta} v_\beta^s,
\]

\[
v_\alpha^s = (A^s_{\alpha\beta} - A^s_{\alpha\xi} [A^s_{\xi\xi}]^{-1} A^s_{\xi\beta})v_\beta^s
\]

This procedure is outlined in the following algorithm:

**Algorithm 5: Dirichlet Solver Procedure**

1. **Input** \((v_\beta^s)\)
2. **Compute** : \( v_1^s = A^s_{\alpha\beta} v_\beta^s \)
3. **Solve** : \( A^s_{\xi\xi} v_2^s = v_1^s \)
4. **Compute** : \( v_3^s = A^s_{\alpha\xi} v_2^s \)
5. **Compute** : \( v_4^s = A^s_{\alpha\beta} v_\beta^s \)
6. Compute: $v_α^s = v_4^s - v_3^s$
7. Output ($v_α^s$)

The global problem in step 5 of Algorithm (4) is solved iteratively using PCGM equipped with lumped preconditioner as

$$M_{cc}^{-1} F_{cc} V_2 = M_{cc}^{-1} V_1,$$

where

$$M_{cc}^{-1} = \sum_{s=1}^{n_s} B^s_c^T A^s_{cc} B^s_c$$

Next the effect of the two-level preconditioner in step 12 of Algorithm (1) is computed by solving a subdomain level Neumann problem and a global coarse problem. The procedure is outlined in the following algorithm:

**Algorithm 6: Two-Level Preconditioner Effect Procedure**

1. Input ($r_Γ^r$)
2. Scatter: $F^s_r = D^s_r B^s r_Γ$
3. Local Solve: $S^s_{rr} v^s_1 = F^s_r$
4. Compute: $d^s_c = S^s_{cr} v^s_1$
5. Gather: $d_c = \sum_{s=1}^{n_s} B^s_c^T d^s_c$
6. Global Solve: $F^s_{cc} Z^s_c = -d_c$
7. Scatter: $Z^s_c = B^s_r Z^s_c$
8. Update: $v^s_2 = F^s_r + S^s_{rr} Z^s_c$
9. Local Solve: $S^s_{rr} Z^s_f = v^s_2$
10. Gather: $Z = \sum_{s=1}^{n_s} B^s_r^T D^s_r Z^s_f$
11. Output ($Z$)

The local solve in step 3 and step 9 of Algorithm (6) constitute a subdomain level Neumann problem of the form: $S^s_{rr} U^s_r = r^s_r$ which can be solved using the following algorithm:

**Algorithm 7: Neumann-Solver Procedure**

1. Input ($r^s_r$)
2. Solve:
   $$\begin{bmatrix}
   A^s_{ii} & A^s_{ir}
   \\
   A^s_{ri} & A^s_{rr}
   \end{bmatrix}
   \begin{bmatrix}
   X^s_r \\
   U^s_r
   \end{bmatrix}
   = \begin{bmatrix}
   0 \\
   r^s_r
   \end{bmatrix}$$
3. Output ($U^s_r$)
The global solve of the coarse problem in step 6 of Algorithm (6) is conducted in parallel using PCGM equipped with lumped preconditioner as

\[ M^{-1}_{cc} F^* \tilde{Z}_c = -M^{-1}_{cc} d_c, \]

where

\[ M^{-1}_{cc} = \sum_{s=1}^{n_s} B_s^c A_s^c B_s^c. \]

Finally, we summarize the parallel implementation of the PCGM to solve the primal interface problem in the following flow chart.

**Figure 6.** Flowchart of Parallel PCGM to solve the primal interface problem

### 9.2. Dual-primal method

In this subsection, we outline the parallel implementation of the Algorithm (1) to solve the dual-primal interface problem Eq.(29).

For the \( j^{th} \) iteration of Algorithm (1), the matrix-vector product in step 7 defined as

\[ Q_j = (\tilde{F}_{rr} + \tilde{F}_{rc} [\tilde{F}_{cc}]^{-1} \tilde{F}_{cr}) \mathcal{P}_j \]

can be computed using the following algorithm:

**Algorithm 8 :** Parallel Matrix-Vector Product for Dual-Primal Method

1. **Input** \( (\mathcal{P}) \)
2. **Scatter** : \( \mathcal{P}^s = B^s_c^T \mathcal{P} \)
3. **Local Solve** : \( S^s_{rr} v^s_1 = \mathcal{P}^s \)
4. **Compute** : \( v^s_2 = S^s_{cr} v^s_1 \)
5. **Gather** : \( V_2 = \sum_{s=1}^{n_s} B^s_c^T v^s_2 \)
6. **Global Solve**: \[ \bar{F}_{c}V_3 = V_2 \]
7. **Scatter**: \[ v_3^s = B_3^sV_3 \]
8. **Compute**: \[ v_4^s = S_{rc}^sv_3^s \]
9. **Update**: \[ v_5^s = P^s + v_4^s \]
10. **Local Solve**: \[ S_{rr}^sQ^s = v_5^s \]
11. **Gather**: \[ Q = \sum_{s=1}^{n_s} B_r^sQ^s \]
12. **Output** \((Q)\)

The local solve in step 3 and step 10 of Algorithm (8) is calculated by solving a subdomain level Neumann problem as outlined in Algorithm (7). The global coarse problem in step 6 of Algorithm (8) is solved in parallel using PCGM with lumped preconditioner similar to the procedure of solving the coarse problem in the primal preconditioner.

Next the effect of the Dirichlet Preconditioner in step 12 of Algorithm (1) is obtained using the following algorithm:

**Algorithm 9**: Dirichlet Preconditioner Effect Procedure

1. **Input** \((r_F)\)
2. **Scatter**: \[ r_F^s = D_r^sB_r^sT_{rF} \]
3. **Compute**: \[ Z^s = S_{rr}^sv_F^s \]
4. **Gather**: \[ Z = \sum_{s=1}^{n_s} B_r^sD_r^sZ^s \]
5. **Output** \((Z)\)

We summarize the parallel implementation of the PCGM to solve the dual-primal interface problem in the following flow chart.

**Figure 7**: Flowchart of Parallel PCGM to solve the dual-primal interface problem
10. Numerical results

For numerical illustrations to the aforementioned mathematical framework, we consider a stationary stochastic Poisson’s equation with randomly heterogeneous coefficients given as

\[
\frac{\partial}{\partial x}[c_x(x, y, \theta) \frac{\partial u(x, y, \theta)}{\partial x}] + \frac{\partial}{\partial y}[c_y(x, y, \theta) \frac{\partial u(x, y, \theta)}{\partial y}] = f(x, y) \quad \text{in} \ \Omega,
\]

where the forcing term is

\[f(x, y) = 1.0.\]

For simplicity, a homogeneous Dirichlet boundary condition is imposed as

\[u(x, y, \theta) = 0 \quad \text{on} \ \partial \Omega.\]

The random coefficients \(c_x(x, y, \theta)\) and \(c_y(x, y, \theta)\) are modeled as independent lognormal random variables. The underlying Gaussian random variable has a mean 1.0 and standard deviation 0.25.

In PCGM implementation, the forcing term is taken to be the initial residual, and the iterations are terminated when the ratio of \(L_2\) norms of the current and the initial residual is less than \(10^{-5}\)

\[\frac{\|G_k^\Gamma - SU_k^\Gamma\|_2}{\|G_0^\Gamma\|_2} \leq 10^{-5}.\]

Numerical experiments are performed in a Linux cluster with InfiniBand interconnect (2 Quad-Core 3.0 GHz Intel Xeon processors and 32 GB of memory per node) using MPI [7] and PETSc [6] parallel libraries. The graph partitioning tool METIS [8] is used to decompose the finite element mesh.

10.1. Stochastic features

Finite element discretization with linear triangular elements results in 202,242 elements and 101,851 nodes. The random coefficients and the response are represented by third order polynomial chaos expansion \((L = 7, N = 9)\) leading to a linear system of order 1,018,510. Fig.(8) shows a typical finite element mesh while Fig.(9) shows a typical mesh decomposition. The mean and the associated standard deviation of the solution process are shown in Fig.(10) and Fig.(11), respectively. Clearly the maximum value of the coefficient of variation of the solution field is 0.20. Details of the stochastic features of the solution field are shown in Figs.(12-17) through there Polynomial Chaos coefficients. The mean and the standard deviation of the solution field computed using the dual-primal method (not shown here) exactly match the results from the primal method. In Figs.(18-23) the Polynomial Chaos coefficient of Lagrange multipliers are shown.
10.2. Scalability study
Firstly, we study the scalability of the algorithms with respect to the problem size where we fix the number of subdomains used to solve the problem to 100 while increasing both mesh resolution in the spatial dimension and the Polynomial Chaos order as reported in Table(1). Evidently increasing mesh resolution by factor (10×) does not deteriorate the performance of the primal and the dual-primal algorithms. Simultaneously, increasing Polynomial Chaos order from the first order to third order does not effect the performance of the methods. Note that for a given spatial problem size (n), using the first order Polynomial Chaos expansion leads to a total problem size of (3×n), and using the third order Polynomial Chaos expansion leads to
a total problem size of \((10 \times n)\).

Secondly, we fix the problem size in the spatial domain to \((7,1389\ \text{dofs})\) and increase the number of subdomains used to solve the problem the results are presented in Table(2). The results reported for first, second and third order Polynomial Chaos expansion. These performance results suggest that both the primal and the dual-primal methods are scalable with respect to number of subdomains. Clearly, the dual-primal method requires slightly less number of iter-
Table 1. Iteration counts of the primal and dual-primal methods for fixed number of subdomain (100).

| Problem size | PDDM |                  |         | DP-DDM |                  |         |
|--------------|------|------------------|---------|--------|------------------|---------|
|              | 1st  | 2nd  | 3rd  | 1st  | 2nd  | 3rd  |
| 10,051        | 10   | 10   | 10   | 8    | 8    | 8    |
| 20,303        | 11   | 11   | 11   | 8    | 8    | 8    |
| 40,811        | 11   | 12   | 12   | 8    | 9    | 9    |
| 59,935        | 13   | 14   | 14   | 10   | 10   | 10   |
| 71,386        | 12   | 12   | 12   | 9    | 9    | 9    |
| 80,172        | 11   | 11   | 12   | 8    | 8    | 8    |
| 101,851       | 12   | 12   | 12   | 9    | 9    | 9    |

Iterations to converge than the primal method. This may be attributed to fact that the starting initial residual in the dual-primal method is smaller than the starting initial residual in the primal method. However, the rate of convergence of both the methods is almost the same as indicated in Figs.(24-26).

Thirdly, we fix problem size per subdomain while increase the overall problem size by adding
Table 2. Iteration counts of the primal and dual-primal methods for fixed problem size of 71,386 dof.

| CPUs | PDDM | DP-DDM |
|------|------|--------|
|      | 1st  | 2nd    | 3rd   | 1st  | 2nd    | 3rd   |
| 20   | 10   | 11     | 11    | 8    | 8      | 8     |
| 40   | 12   | 12     | 12    | 9    | 9      | 9     |
| 60   | 12   | 13     | 13    | 9    | 9      | 9     |
| 80   | 12   | 12     | 13    | 9    | 9      | 9     |
| 100  | 12   | 12     | 12    | 9    | 9      | 9     |
| 120  | 12   | 12     | 12    | 9    | 9      | 9     |
| 140  | 11   | 11     | 12    | 8    | 8      | 8     |
| 160  | 12   | 12     | 12    | 8    | 8      | 9     |

Table 3. Iteration counts of the primal and dual primal methods for fixed problem size per subdomain (101,851 dof).

| Subdomains | PDDM | DP-DDM |
|------------|------|--------|
|            | 1st  | 2nd    | 3rd   | 1st  | 2nd    | 3rd   |
| 100        | 10   | 10     | 10    | 8    | 8      | 8     |
| 200        | 10   | 10     | 11    | 8    | 8      | 8     |
| 400        | 12   | 13     | 13    | 9    | 9      | 9     |
| 600        | 11   | 12     | 12    | 8    | 8      | 9     |
| 800        | 12   | 13     | 13    | 9    | 9      | 9     |

more subdomains. Table (3) shows the performance of the primal and the dual-primal methods for the first, second and third order Polynomial Chaos expansion. Again these results suggest that both the primal and the dual-primal methods are scalable with respect to fixed problem size per subdomain.

Fourthly, we study the performance of the primal and the dual-primal methods with respect the strength of randomness of the system parameters. Table (4) shows the performance of the algorithms when the Coefficient of variation (CoV) of the random parameters is varied from (5% to 50%). Clearly, the strength of the randomness does not degrade the performance of the algorithms as the number of PCGM iteration is nearly constant.

Finally, it worth mentioning that the performances of the primal method and the dual-primal method demonstrate similar trend and this fact points out the similarity (duality) between the two methods through numerical experiments.

11. Conclusion

Novel primal and dual-primal domain decomposition methods are proposed to solve the algebraic linear system arising from the stochastic finite element method. The primal method is equipped with a scalable two-level preconditioner. The numerical experiments illustrate that the proposed
Figure 24. The relative PCGM residual history for the case of 160 subdomains and first PC order.

Figure 25. The relative PCGM residual history for the case of 160 subdomains and second PC order.

Figure 26. The relative PCGM residual history for the case of 160 subdomains and third PC order.

The primal method and the dual-primal method are numerically scalable with respect to problem size, subdomain size and number of subdomains. Both algorithms exhibit similar convergence rates with respect to the coefficient of variation (i.e. the level of uncertainty) and Polynomial Chaos order. Both primal and dual-primal iterative substructuring methods exploit a coarse grid in the geometric space. At this point, it is worth mentioning that adding a coarse grid in the stochastic space would be beneficial in the cases where a large number of random variables are required to prescribe uncertainty in the input parameters. This aspect is currently being investigated by the authors.

Acknowledgments
The authors gratefully acknowledge the financial support from the Natural Sciences and Engineering Research Council of Canada through a Discovery Grant, Canada Research Chair Program, Canada Foundation for Innovation and Ontario Innovation Trust. Dr. Ali Rebaine for his help with ParMETIS graph-partitioning software.
Table 4. Iteration counts of the primal and dual primal methods for different CoV, fixed problem size (10,1851 dofs) and fixed number of subdomains (100).

| CoV | PDDM | DP-DDM |
|-----|------|--------|
|     | 1st  | 2nd    | 3rd   | 1st  | 2nd    | 3rd   |
| 0.05| 10   | 10     | 10    | 8    | 8      | 8     |
| 0.10| 10   | 10     | 10    | 8    | 8      | 8     |
| 0.15| 10   | 10     | 10    | 8    | 8      | 8     |
| 0.20| 10   | 10     | 10    | 8    | 8      | 8     |
| 0.25| 10   | 10     | 11    | 8    | 8      | 8     |
| 0.30| 10   | 10     | 11    | 8    | 8      | 8     |
| 0.35| 10   | 10     | 11    | 8    | 8      | 9     |
| 0.40| 10   | 11     | 11    | 8    | 8      | 9     |
| 0.45| 10   | 11     | 12    | 8    | 8      | 9     |
| 0.50| 10   | 11     | 12    | 8    | 8      | 9     |

References

[1] Sarkar A, Benabbou N and Ghanem R 2009 *International Journal for Numerical Methods in Engineering* 77 689–701
[2] Sarkar A, Benabbou N and Ghanem R 2010 *International Journal of High Performance Computing Applications* Accepted
[3] Subber W, Monajemi H, Khalil M and Sarkar A 2008 *International Symposium on Uncertainties in Hydrologic and Hydraulic*
[4] Subber W and Sarkar A 2010 *High Performance Computing Systems and Applications* (Lecture Notes in Computer Science vol 5976) ed et al D M (Springer Berlin / Heidelberg) pp 251–268
[5] Farhat C, Lesoinne M and Pierson K 2000 *Numerical Linear Algebra with Application* 7 687–714
[6] Balay S, Buschelman K, Gropp W D, Kaushik D, Knepley M G, McInnes L C, Smith B F and Zhang H 2009 PETSc: Web page http://www.mcs.anl.gov/petsc
[7] Message passing interface forum http://www.mpi-forum.org
[8] Karypis G and Kumar V 1995 METIS unstructured graph partitioning and sparse matrix ordering system
[9] Ghanem R and Spanos P 1991 *Stochastic Finite Element: A Spectral Approach* (New York: Springer-Verlag)
[10] Saad Y 2003 *Iterative methods for sparse linear systems* 2nd ed (Philadelphia)
[11] Toselli A and Widlund O 2005 *Domain Decomposition Methods - Algorithms and Theory* (Springer Series in Computational Mathematics vol 34) (Berlin: Springer)
[12] Li J and Widlund O 2010 *International Journal for Numerical Methods in Engineering* 66(2) 250–271