SAND2016-5380C

Domain Decomposition of Stochastic PDEs: High Resolution Computational Mesh with Large Stochastic Dimension

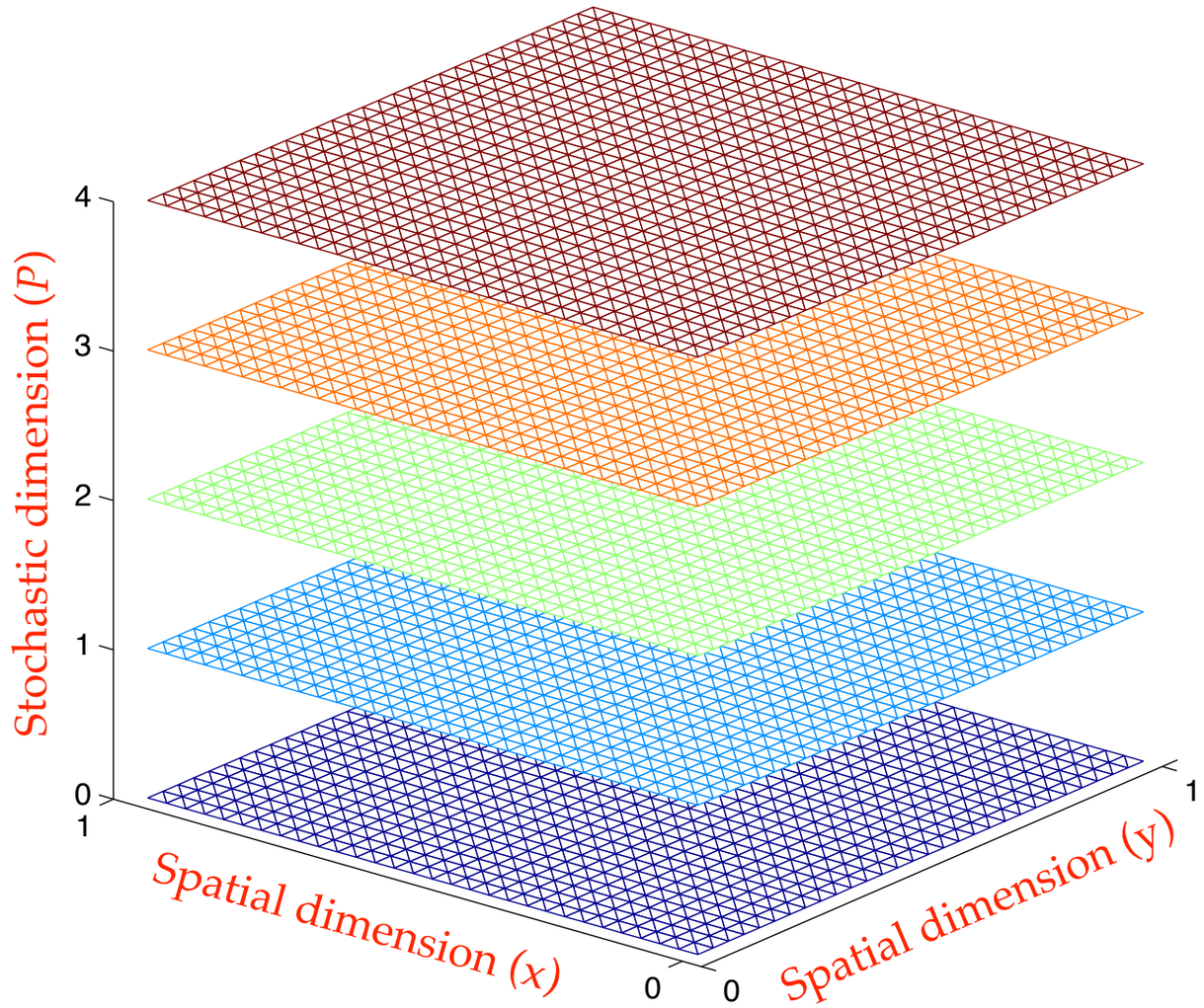
Ajit Desai, Carleton University, Canada
Mohammad Khalil, Sandia National Lab., USA
Abhijit Sarkar, Carleton University, Canada
Chris Pettit, United States Naval Academy, USA
Dominique Poirel, Royal Military College, Canada

Carleton UNIVERSITY

FIELDS ICSCA 2016
University of Toronto, Canada

Uncertainty Quantification at Extreme Scale

- Deterministic finite element method with spatial mesh resolution ***N = 100,000***.



- Stochastic finite element method with stochastic discretization using ***P = 100*** PCE terms will leads to the total problem of size,

$$N \times P = 10 \text{ million} \longrightarrow \textbf{HPC}$$

Introduction

- **Motivation**
  - Efficient uncertainty quantification algorithms for extreme-scale stochastic systems, leveraging modern high-performance computing.

- **Objective**
  - Develop parallel scalable algorithms for uncertainty quantification of systems with high resolution in both stochastic and physical domain.

- **Methodology**
  - Employ intrusive polynomial chaos expansion based Stochastic FEM.
  - Develop non-overlapping domain decomposition based solvers.
  - Efficient use of memory and floating-point-operations using PETSc.

Stochastic Partial Differential Equation (SPDE)

- Stationary stochastic diffusion equation.

$$-\nabla \cdot \left(\; c_d(\boldsymbol{x}, \theta) \;\; \nabla u(\boldsymbol{x}, \theta) \; \right) = f(\boldsymbol{x}), \qquad D \times \Omega$$

- Diffusion coefficient (c_d) is modelled as a lognormal stochastic process.

$$c_d(\boldsymbol{x}, \theta) = \exp \left(g(\boldsymbol{x}, \theta) \right)$$

- Obtained from underlying Gaussian process (g) with 2D exponential covariance.

$$C(x_1, y_1; x_2, y_2) = \sigma^2 \; e^{-|x_2 - x_1|/b_1 - |y_2 - y_1|/b_2}$$

Stochastic Spectral Finite Element Method (SSFEM)

- **Stochastic Input** : Truncated Karhunen-Loève Expansion (KLE)

$$g(\boldsymbol{x},\theta) = g_0(\boldsymbol{x}) + \sum_{i=1}^{L} \sqrt{\lambda_i}\; g_i(\boldsymbol{x})\; \xi_i(\theta)$$

- **Model parameters/Solution process** : Polynomial Chaos Expansion (PCE)

$$\alpha\left(\boldsymbol{x},\xi(\theta)\right) = \sum_{j=0}^{P} \alpha_j(\boldsymbol{x})\; \Psi_j(\xi(\theta))$$

$$P = \frac{(L+p)!}{L!\; p!} - 1$$

where $\boldsymbol{P}$ is a number of PCE terms, $\boldsymbol{p}$ is order of chaos and $\boldsymbol{L}$ is dimensionality of stochastic space.

Domain Decomposition Method (DDM)

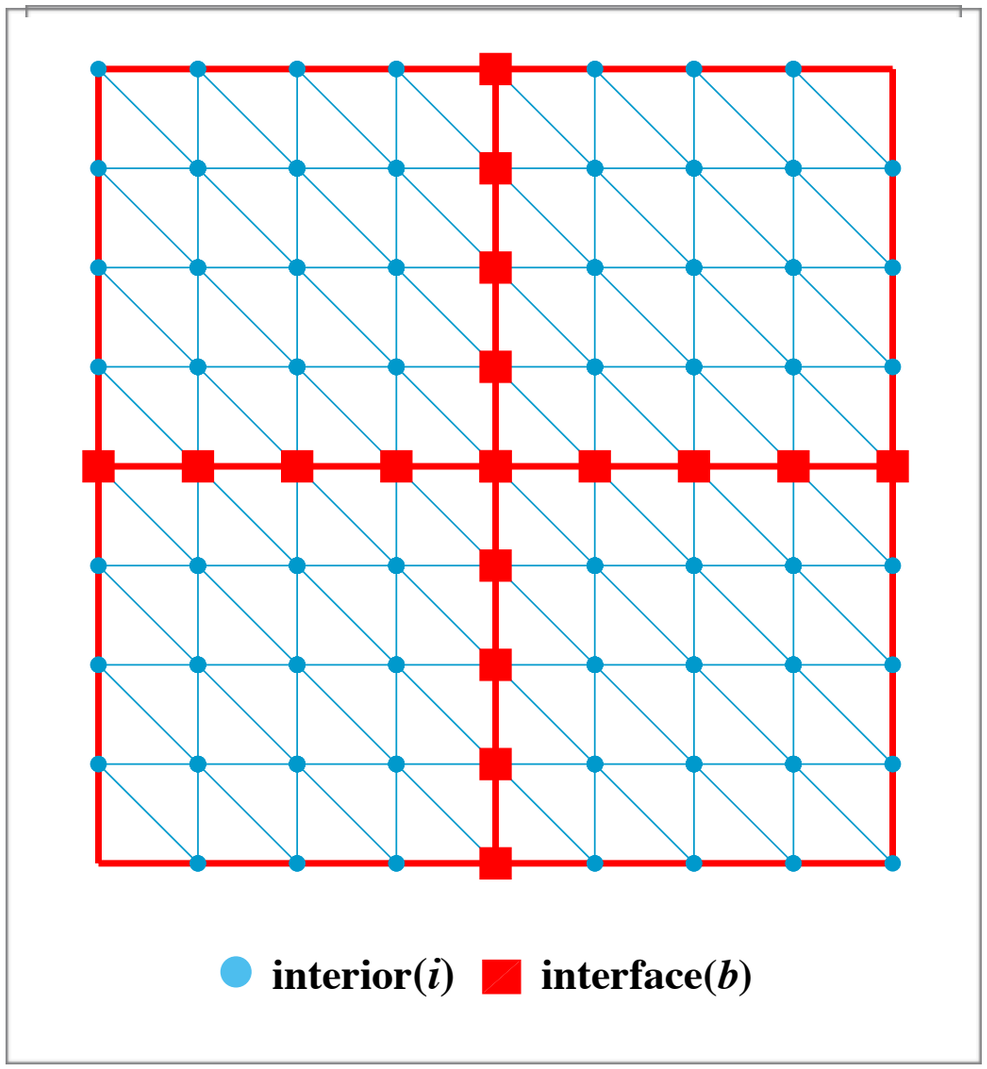
- Finite element approximation of the SPDE

$$\mathbf{A}(\theta)\ \mathbf{u}(\theta) = \mathbf{f}$$

- Spatial domain D is divided into n_s non-overlapping subdomains

$$\begin{bmatrix} \mathbf{A}_{ii}^{s} & \mathbf{A}_{ib}^{s} \\ \mathbf{A}_{bi}^{s} & \mathbf{A}_{bb}^{s} \end{bmatrix} \begin{Bmatrix} \mathbf{u}_{i}^{s} \\ \mathbf{u}_{b}^{s} \end{Bmatrix} = \begin{Bmatrix} \mathbf{f}_{i}^{s} \\ \mathbf{f}_{b}^{s} \end{Bmatrix}$$

$$s = 1,\ 2,\ \ldots,\ n_s$$



DDM for Stochastic PDEs : Intrusive Approach

- PCE substitution, Galerkin projection, enforce global assembly

$$\begin{bmatrix} A^1_{ii} & \dots & 0 & A^1_{ib}R_1 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \dots & A^{n_s}_{ii} & A^{n_s}_{ib}R_{n_s} \\ R^T_1 A^1_{bi} & \dots & R^T_{n_s} A^{n_s}_{bi} & \sum_{s=1}^{n_s} R^T_s A^s_{bb} R_s \end{bmatrix} \begin{Bmatrix} u^1_i \\ \vdots \\ u^{n_s}_1 \\ u_b \end{Bmatrix} = \begin{Bmatrix} f^1_i \\ \vdots \\ f^{n_s}_i \\ \sum_{s=1}^{n_s} R^T_s f^s_b \end{Bmatrix}$$

where

$$\left[A^s_{\alpha\beta}\right]_{jk} = \sum_{i=0}^{L} \langle \Psi_i \Psi_j \Psi_k \rangle \left[\mathbf{A^s_{\alpha\beta}}\right]_i$$

Extended Schur Complement System : Primal Approach

- Eliminating interior degree of freedom,

$$Su_b = g_b$$

$$S = \sum_{s=0}^{n_s} R_s^T \left[A_{bb}^s - A_{bi}^s [A_{ii}^s]^{-1} A_{ib}^s \right] R_s$$

$$g_b = \sum_{s=1}^{n_s} R_s^T [f_b^s - A_{bi}^s [A_{ii}^s]^{-1} f_i^s]$$

- Preconditioned Schur complement system to solve :

$$M^{-1} S u_b = M^{-1} g_b$$

Preconditioned Conjugate Gradient Method (PCGM)

- One-Level preconditioner :

$$M_1^{-1} = \sum_{s=1}^{n_s} R_s^T [M_s]^{-1} R_s$$

$$k(M_1^{-1}S) \leq C \frac{1}{H^2} \left(1 + log \frac{H}{h}\right)^2$$

- Two-Level preconditioner :

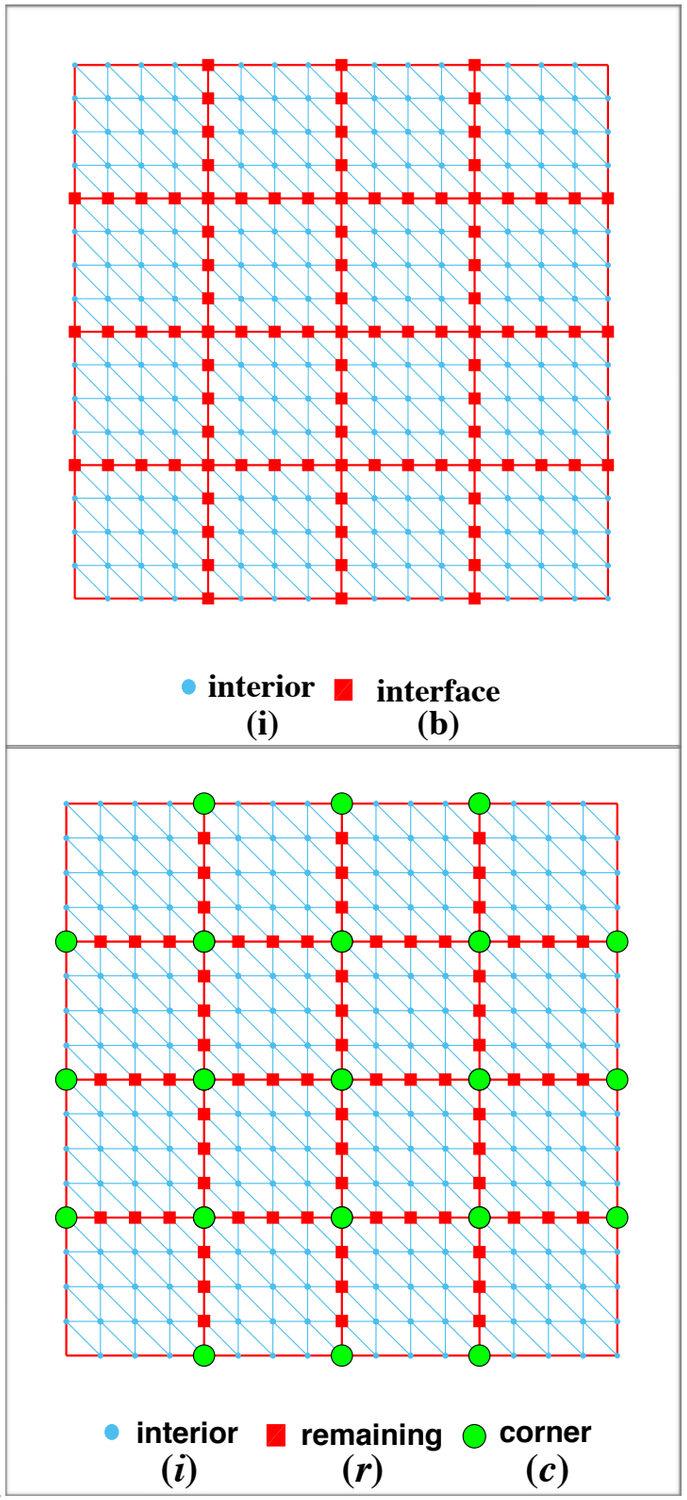
$$M_2^{-1} = \sum_{s=1}^{n_s} R_s^T [M_s]^{-1} R_s + R_0^T [M_0]^{-1} R_0$$

Fine **Coarse**

$$k(M_2^{-1}S) \leq C \left(1 + log \frac{H}{h}\right)^2$$

*Condition number bound of **deterministic** systems.

where ***h*** is the finite element mesh size and ***H*** is the subdomain size.



FETI-DP: Dual-Primal Approach

- Preconditioned Lagrange multiplier system of PCE coefficients

$$M_D^{-1}\left[F_{rr} + F_{rc}[F_{cc}]^{-1}F_{cr}\right]\Lambda = M_D{}^{-1}(d_r - F_{rc}[Fcc]^{-1}d_c)$$

- One-level Dirichlet preconditioner

$$M_D^{-1} = \sum_{s=1}^{n_s} B_r^s D_r^s [S_{rr}^s] D_r^s B_r^{s\tau},$$

where

$$F_{cc} = \sum_{s=1}^{n_s} B_c^{s\tau}(S_{cc}^s - S_{cr}^s[S_{rr}^s]^{-1}S_{rc}^s)B_c^s$$

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

PCGM Algorithm : Parallel Implementation

$$M^{-1}Su_b = M^{-1}g_b$$

1: **Compute :** $r_{b_0} = g_b - Su_{b_0}$
2: **Precondition :** $z_0 = M^{-1}r_{b_0}$
3: **Compute :** $p_0 = z_0$
4: **Compute :** $\delta_0 = (r_{b_0}, z_0)$
5: **for** $i = 0, 1, 2, \ldots$, **do**
6: **Compute :** $q_i = Sp_i$
7: **Compute :** $\gamma_i = (q_i \ , \ p_i)$
8: **Compute :** $\alpha_i = \delta_i / \gamma_i$
9: **Update :** $u_{i+1} = u_i + \alpha_i p_i$
10: **Update :** $r_{b_{i+1}} = r_{b_i} - \alpha_i q_i$
11: **If** *iterate has converged,* **Exit**
12: **Precondition :** $z_{i+1} = M^{-1}r_{b_{i+1}}$
13: **Compute :** $\delta_{i+1} = (r_{b_{i+1}}, z_{i+1})$
14: **Compute :** $\beta_i = \delta_{i+1} / \delta_i$
15: **Update :** $p_{i+1} = z_{i+1} + \beta_i p_i$
16: **end for**

(a): Mat-Vec Product: $q_i = \sum_{s=0}^{n_s} R_s^T \left[S_s\right] R_s p_i$

13: **for** $s = 1, 2, \ldots$, **do**
14: Scatter : $p_i^s = R_s \ p_i$
15: Solve : $out2 = A_{ii}^{s\,-1}(A_{ib}^s \ p_i^s)$
16: Compute : $q_i^s = A_{bb}^s \ p_i^s - A_{bi}^s * out2$
17: Gather : $q_i = \sum_{s=0}^{n_s} R_s^T q_i^s$
18: **end for**

(b): Preconditioning: $z_{i+1} = \sum_{s=0}^{n_s} R_s^T \left[M_s^{-1}\right] R_s r_{bi+1}$

24: **for** $s = 1, 2, \ldots$, **do**
25: Scatter : $r_{b_{i+1}}^s = R_s r_{b_{i+1}}$
26: Solve : $z_i^s = M_s^{-1} r_{b_{i+1}}^s$
27: Gather : $z_{i+1} = \sum_{s=0}^{n_s} R_s^T z_i^s$
28: **end for**

Challenges

- Memory required to assemble and store the blocks of subdomain level (local) stochastic FE matrices.

$$[A^s_{ii}],\ [A^s_{bb}],\ [A^s_{cc}],\ \text{etc}\ldots$$

PETSc-*Mat* for Stochastic Sparse Matrix Assembly

- Time required to solve the local problems.

$$[A^s_{ii}]^{-1},\ [A^s_{bb}]^{-1}\ \text{and}\ \ [A^s_{cc}]^{-1}$$

PETSc-*Ksp* Sparse Iterative Solver

- Time, memory and FLOPS for local matrix-vector products.

$$A^s_{ii}v^s, A^s_{ib}v^s, A^s_{bb}v^s, etc\ldots$$

PETSc-*MatMult* for Sparse Mat-Vec Products

One Level PCGM : PETSc-PCGM for Local Problems

Original Problem

$$Au = f$$

Schur Complement
One Level PCGM

$$M_1^{-1} S u_b = M_1^{-1} g_b$$

Mat-Vec Product
~~**Dense-Cholesky**~~
PETSc-PCGM : (1)

$$v_i^s = [A_{ii}^s]^{-1} p_i^s$$

Preconditioning
~~**Dense-Cholesky**~~
PETSc-PCGM : (2)

$$v_b^s = [A_{bb}^s]^{-1} r_b^s$$

NNC/FETI-DP PCGM : PETSc-PCGM for Local Problems

Original Problem

$$Au = f$$

NNC-PCGM

Schur Complement System
Two-Level PCGM

$$M_2^{-1} S u_b = M_2^{-1} g_b$$

FETIDP-PCGM

Lagrange Multiplier System:
One-Level PCGM

$$M_D^{-1}\left[F_{rr} + F_{rc}[F_{cc}]^{-1}F_{cr}\right]\Lambda = M_D^{-1}\left(d_r - F_{rc}[Fcc]^{-1}\, d_c\right)$$

NNC-MVP

NNC-PP

FETIDP-PP

FETIDP-MVP

Local
PETSc-PCGM : (1)

$$v_i^s = [A_{ii}^s]^{-1} p_i^s$$

Global Coarse
One-Level PCGM

$$M_1^{-1} F_{cc} z_c = M_1^{-1} d_c$$

Local Fine
PETSc-PCGM : (2)

$$z_r^s = [S_{rr}^s]^{-1} v_r^s$$

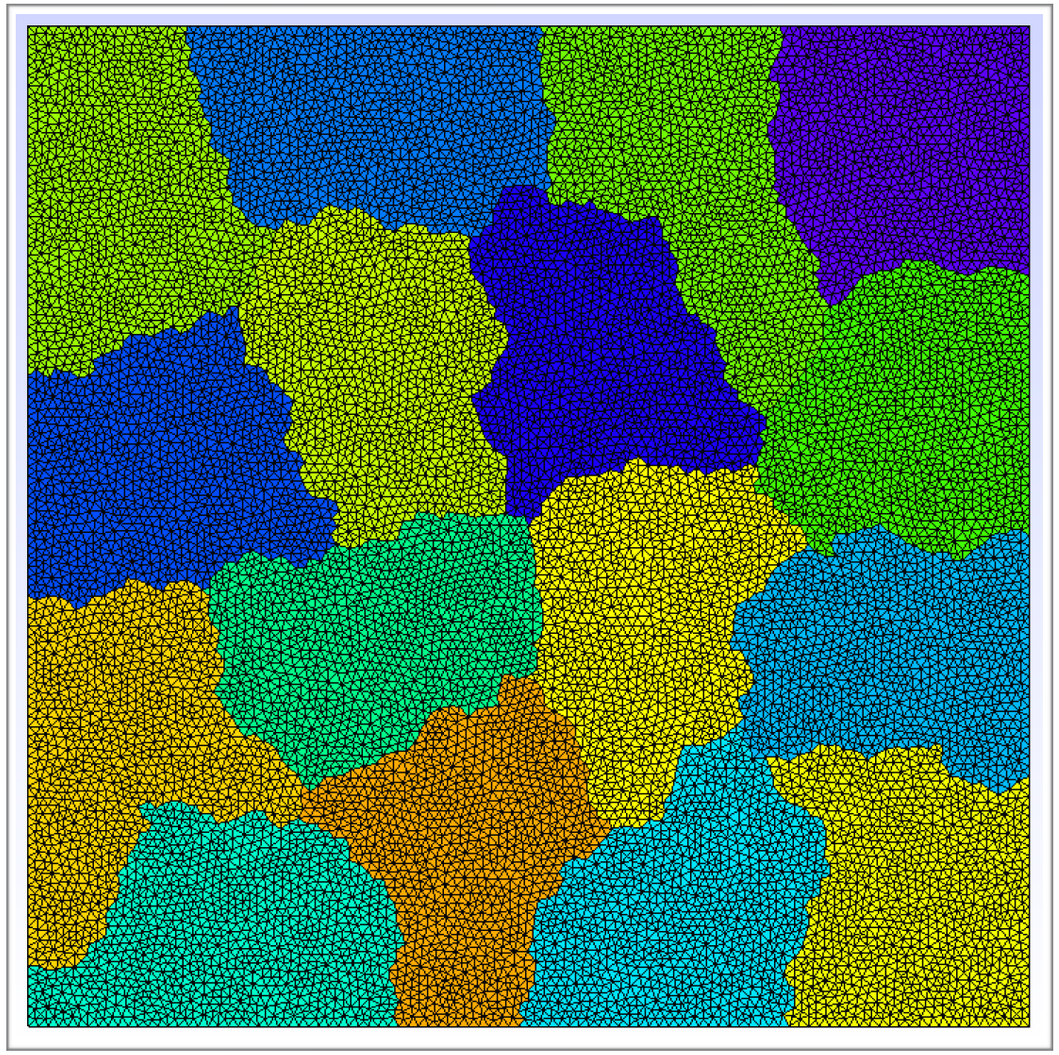
Mat-Vec Product
PETSc-PCGM : (3)

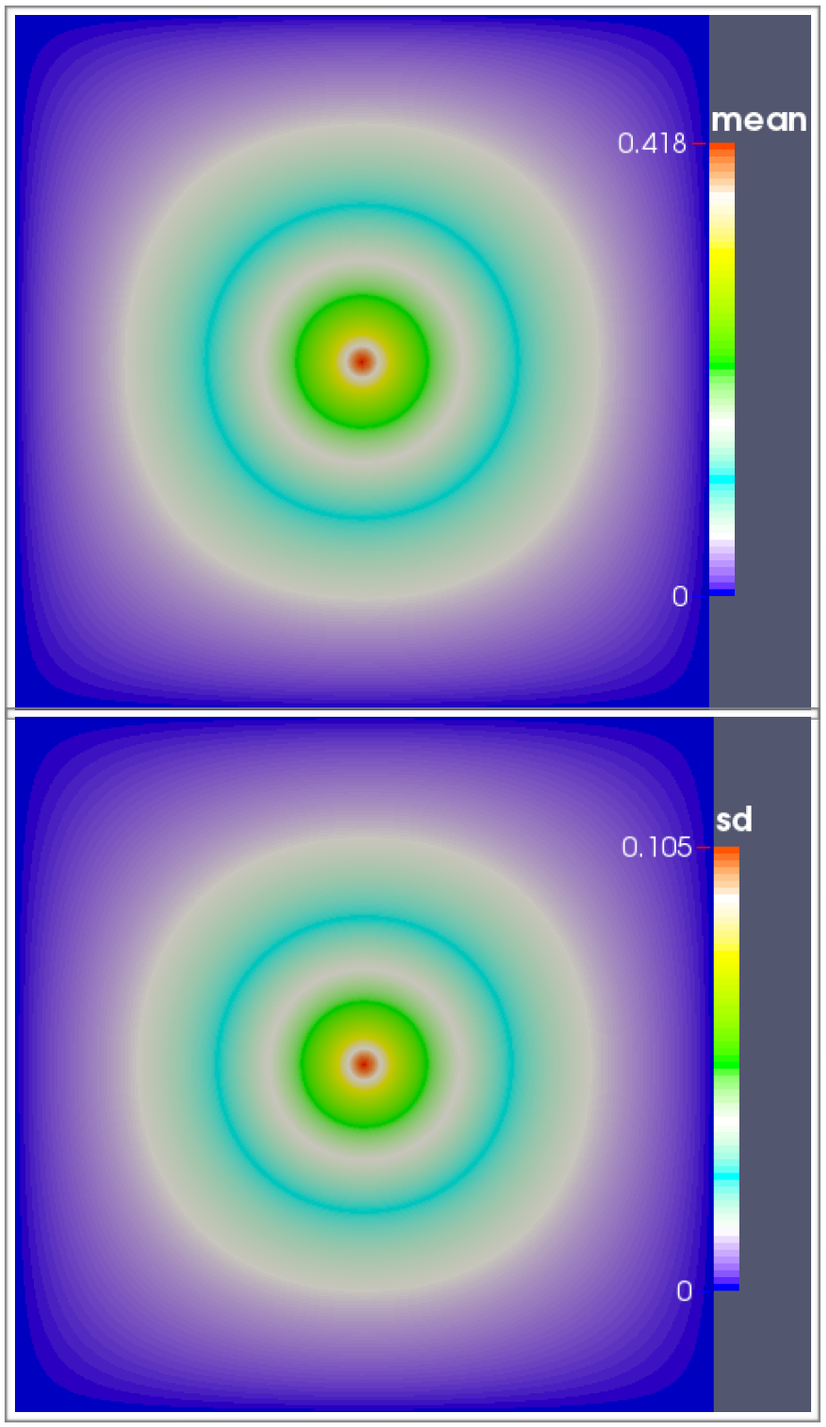
$$v_i^s = [A_{ii}^s]^{-1} p_i^s$$

Preconditioning
PETSc-PCGM : (4)

$$v_c^s = [A_{cc}^s]^{-1} r_c^s$$

Numerical Results : 2D Stochastic Diffusion Equation



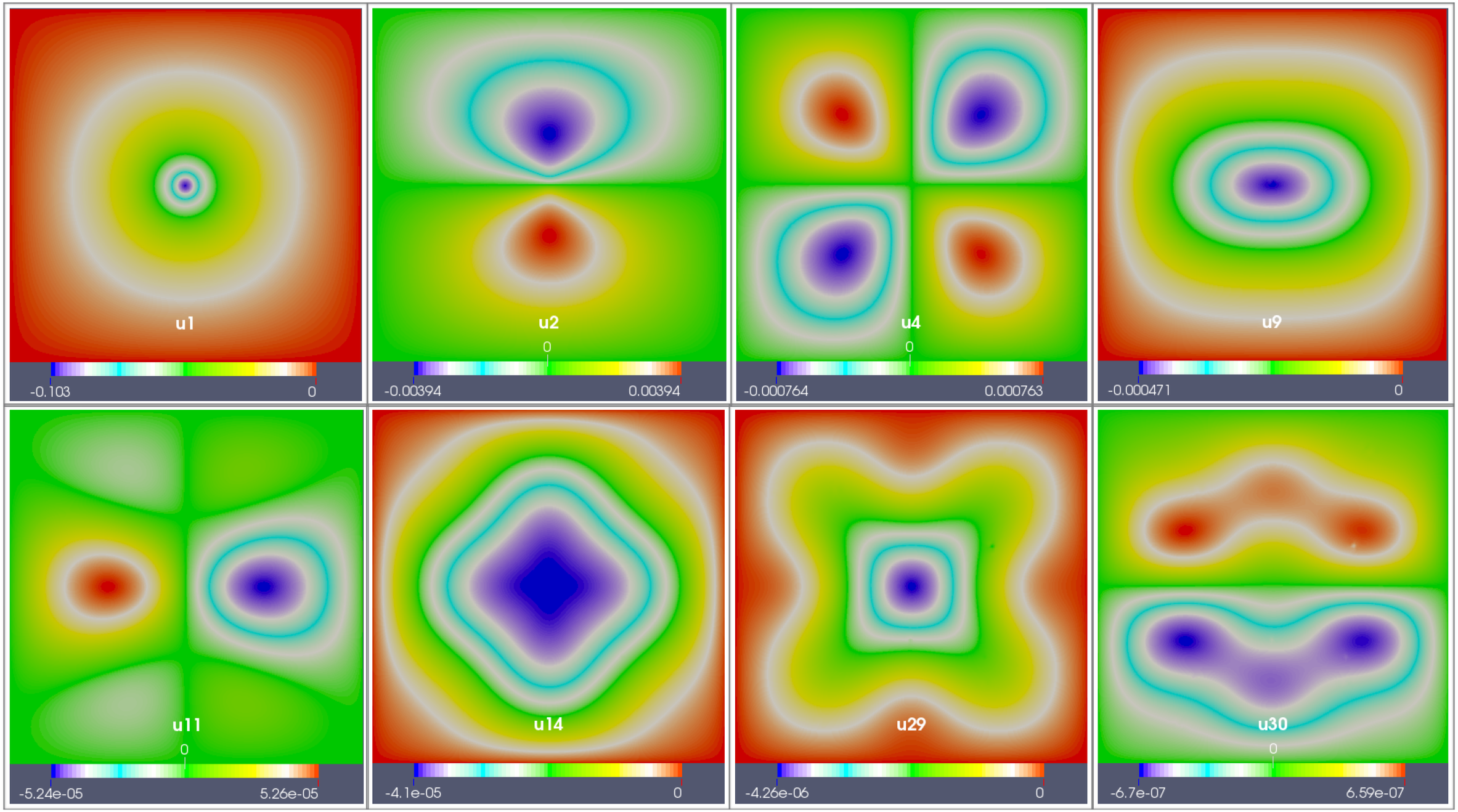


Exponential covariance function with numerical parameters :

$\sigma = 0.3,\ b_1 = b_2 = 1.0$ and

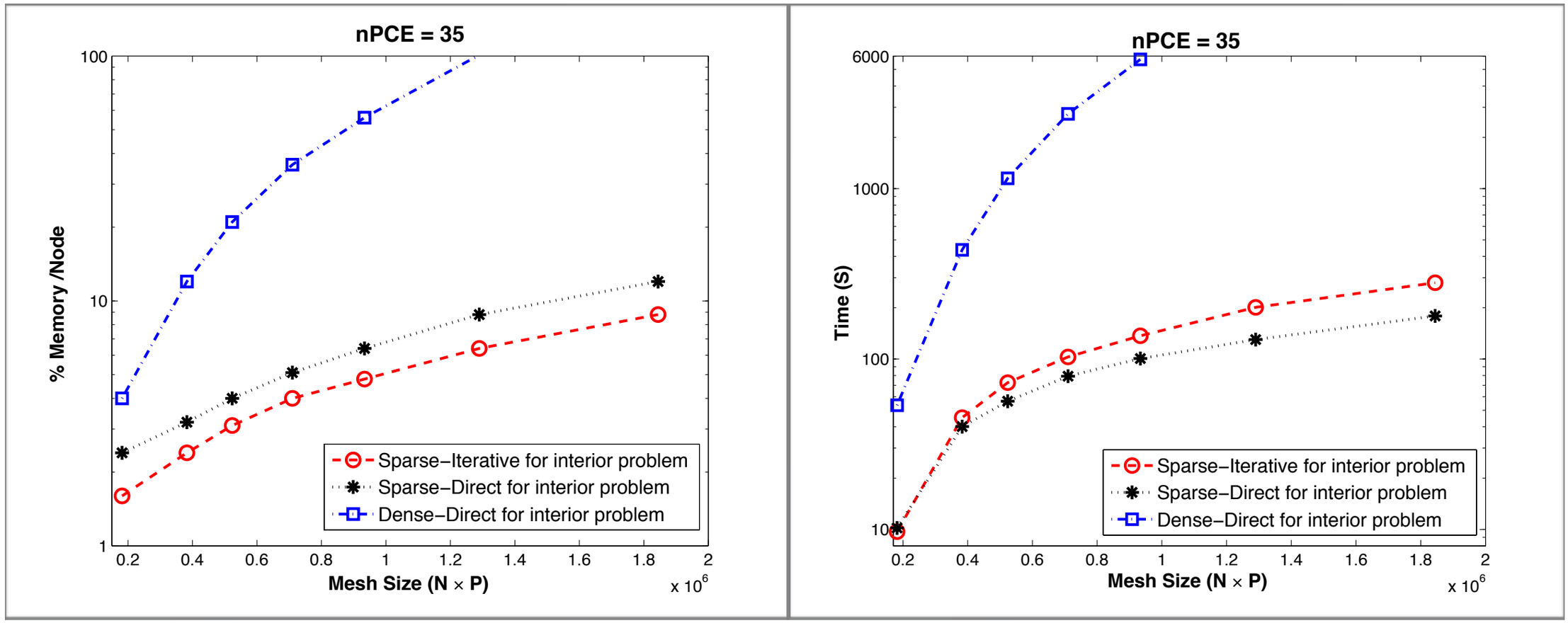
$L = 4,\ p = 3,\ \mathbf{P} = 35$

Few Higher Order PCE Coefficients



One Level PCGM: Dense vs PETSc-Sparse

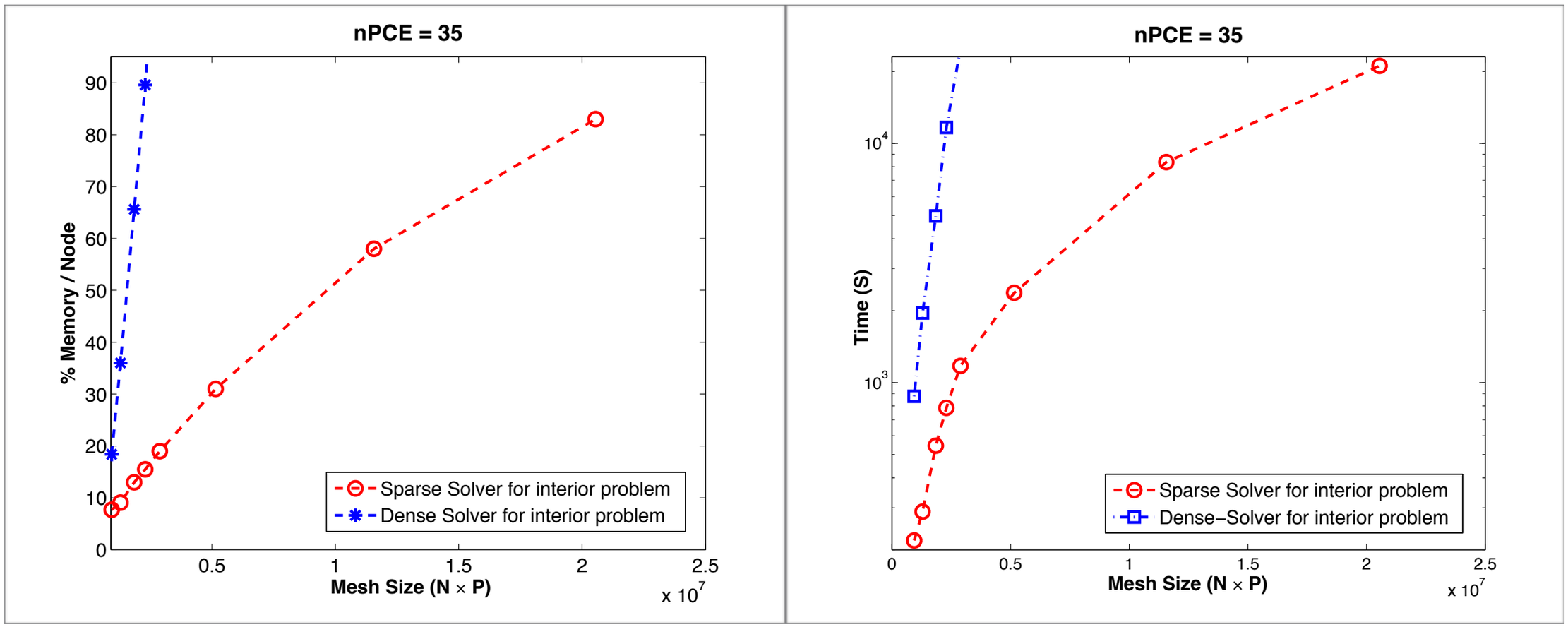
- **Total execution time (including assembly) and memory occupied / node :**
  For fixed number of processors (nProc = 60), with increasing problem size.



- Execution time and memory required using **dense solver** (**LAPACK-Cholesky**) increase exponentially as compared to **sparse-iterative-solver** (**PETSc-PCGM**) **sparse-direct-solver** (**PETSc-MUMPS**) for subdomain-level-interior-problems.

Two Level PCGM: Dense vs PETSc-Sparse

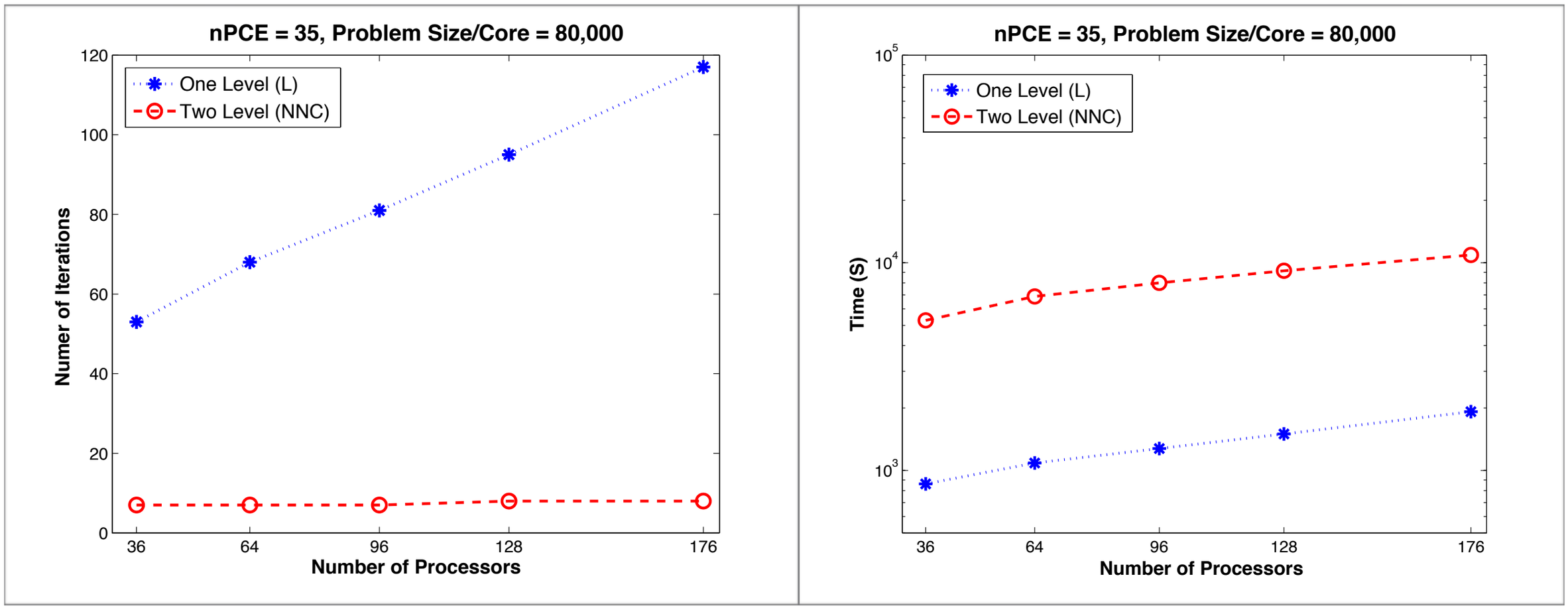
- **Total execution time (including assembly) and memory occupied / node :**
  For fixed number of processors (nProc = 176), with increasing problem size.



- Execution time and memory required using **dense-solver** (**LAPACK-Cholesky**) increase exponentially as compared to **sparse-solve** (**PETSc-PCGM**).

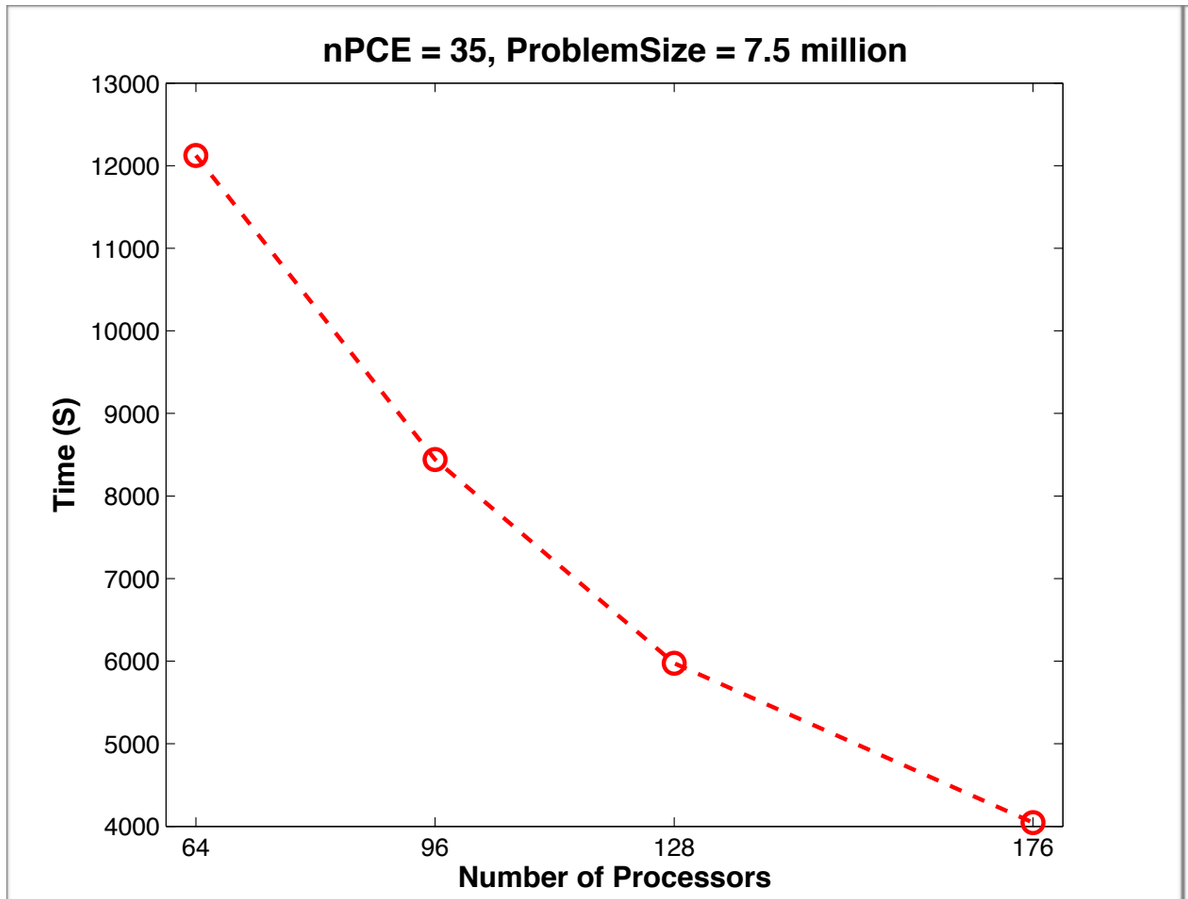
Numerical Scalability : One/Two-Level PCGM

- Scalability studies : performed using Linux-cluster having 22 nodes with 176 processors and 32 GB of memory per node (Quad-Core 3.0 GHz Intel Xeon processors) with InfiniBand interconnect using Message Passing Interface(MPI).

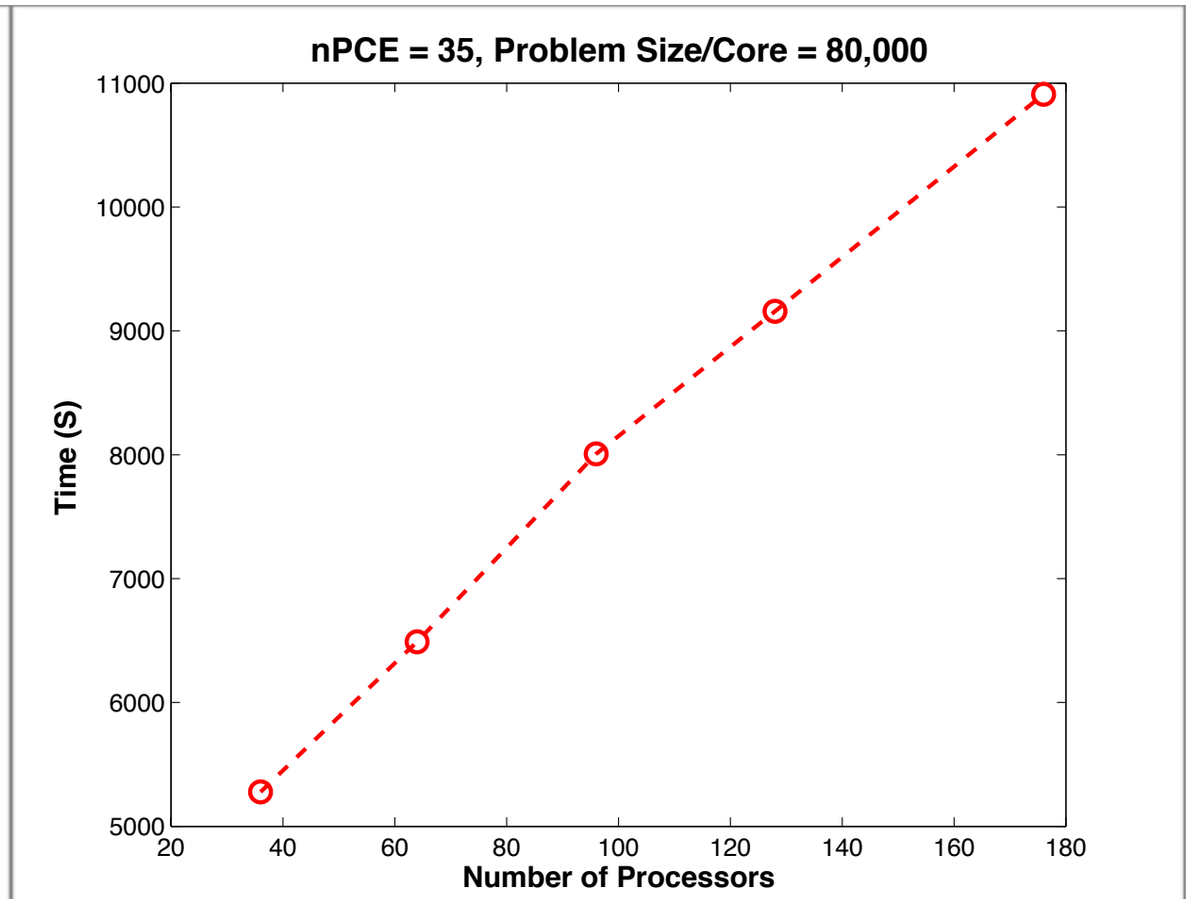


Parallel Scalability : Two-Level PCGM

Strong Scalability



Weak Scalability



- Scalability studies : performed using Linux-cluster with 22 nodes, 2 Quad-Core processors per node, 32 GB of memory per node.

Conclusions

- At extreme scale, the **subdomain-level** i.e. the **local problem** size becomes the bottleneck of the DDM based PCGM algorithms.

  - Optimized sparse memory allocation using **PETSc** objects:
    Mat / Vec / KSP / PC

  - Optimized sparse solvers using **PETSc** routines:
    KSPCG / MUMPS / KSPSolve / PCSetType

  - Optimized FLOPS for algebraic operation using **PETSc** routines:
    MatMult / MatTranspose / VecDot / VecNorm / VecAYPX

- With a given machine, the PETSc-based-DDM-solver could able to solve a much **bigger** problem in **less** time.

Carleton UNIVERSITY

FIELDS ICSCA 2016
University of Toronto, Canada

Thank You!

Questions?