Ph.D. dissertation defense

유한 요소 모델 축소를 위한 병렬 다중 레벨 부구조법 및 성긴 격자 투영

Parallel multilevel substructuring and coarse mesh projection for finite element model reduction

Cheolgyu Hyun

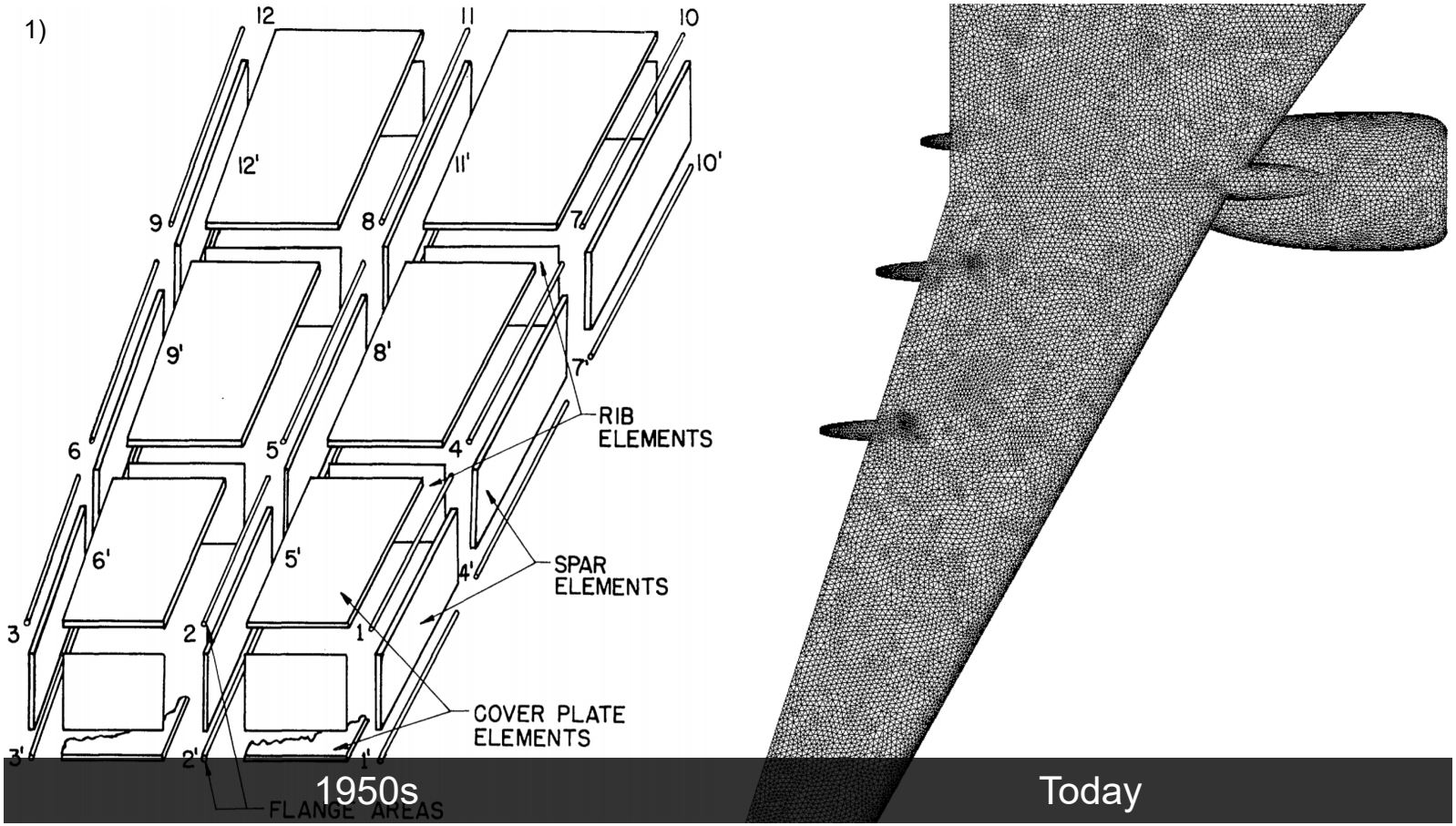
Dec. 6, 2021

KAIST Department of Mechanical Engineering

Contents

1. Introduction

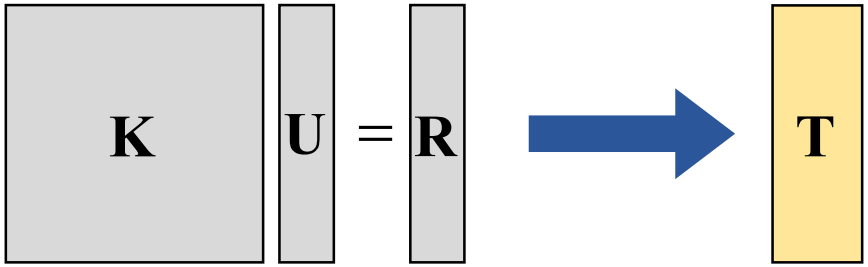
Computational cost is still an important issue.



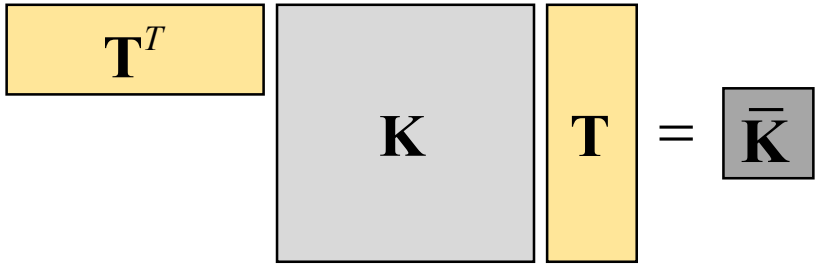
1) M. J. Turner et al., Stiffness and deflection analysis of complex structures, *J. Aero. Sci.*, 1956.

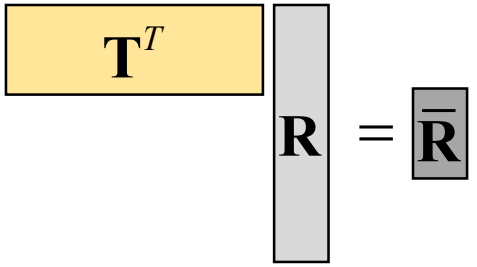
Derive the low-dimensional approximation.

Step 1 of 3. Extract a low-dimensional basis matrix $\mathbf{T}$.

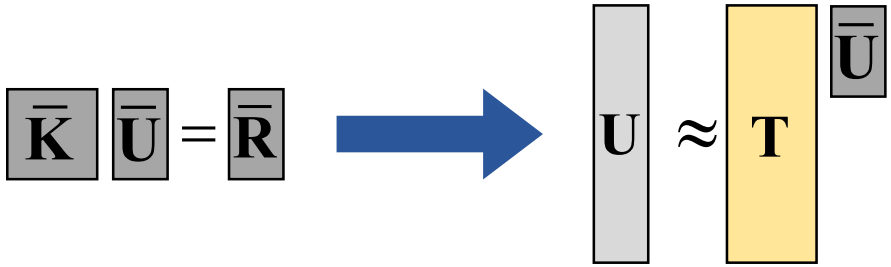


Step 2 of 3. Project the original model onto the low-dimensional subspace.



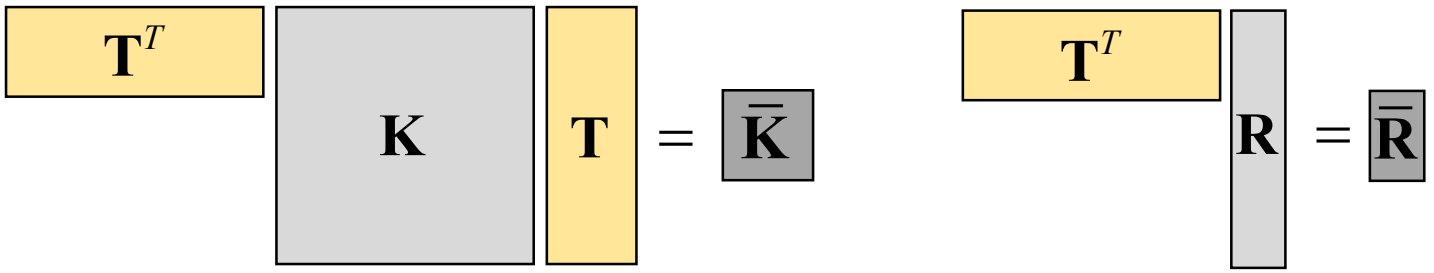


Step 3 of 3. Compute the approximate solution.



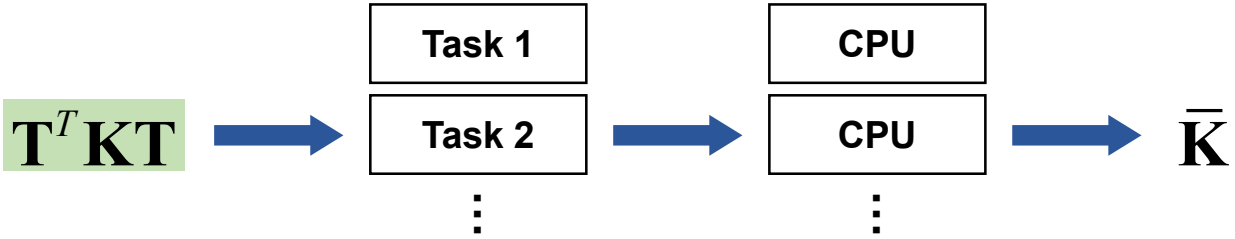
Problems of interest

- **Projection procedure**



1. **Linear model reduction (topic 1)**

How to perform the projection in parallel?



2. **Nonlinear model reduction (topic 2)**

How to calculate the nonlinear term in each time step?

$$\mathbf{T}^T\,{}^t\mathbf{K}({}^t\mathbf{U})\mathbf{T} = {}^t\bar{\mathbf{K}}({}^t\mathbf{U}) \quad \rightarrow \quad {}^t\bar{\mathbf{K}}({}^t\mathbf{U}) \approx {}^t\tilde{\mathbf{K}}({}^t\mathbf{U})$$

Approximate nonlinear term

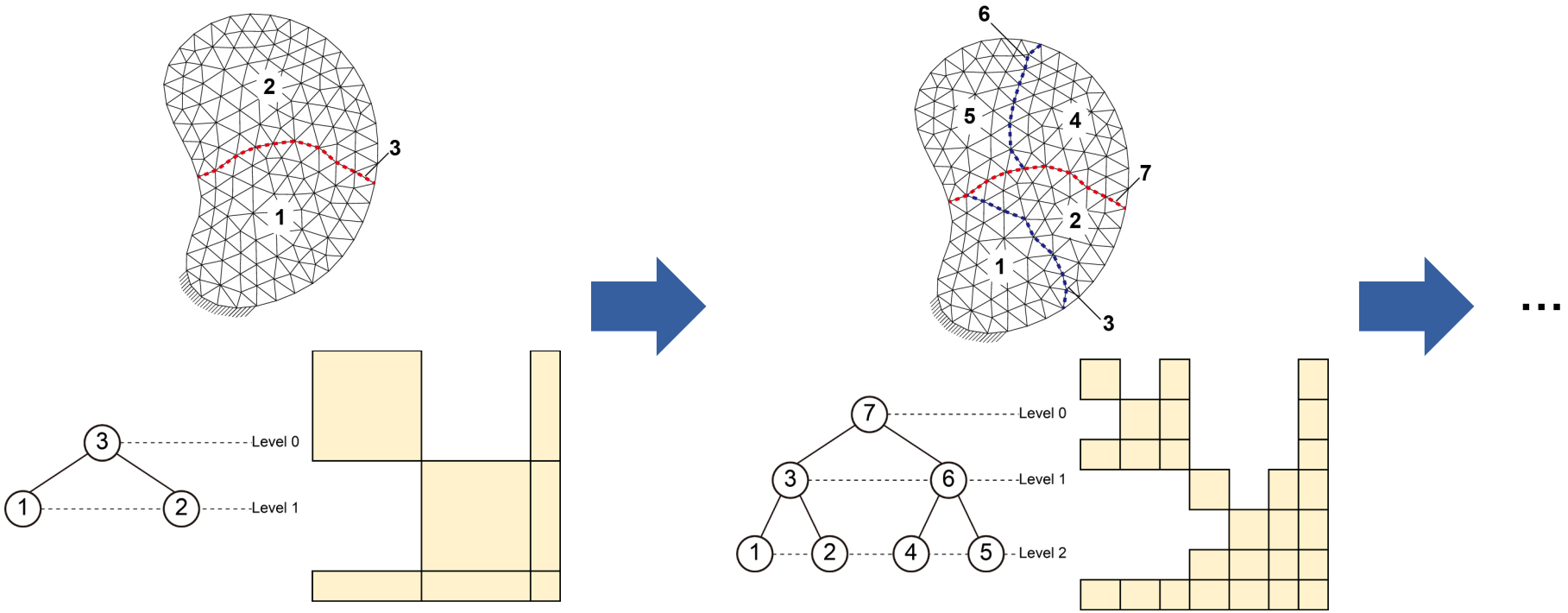
Topic 1

2. Load balancing algorithm for the parallel AMLS method

Automated multilevel substructuring (AMLS)

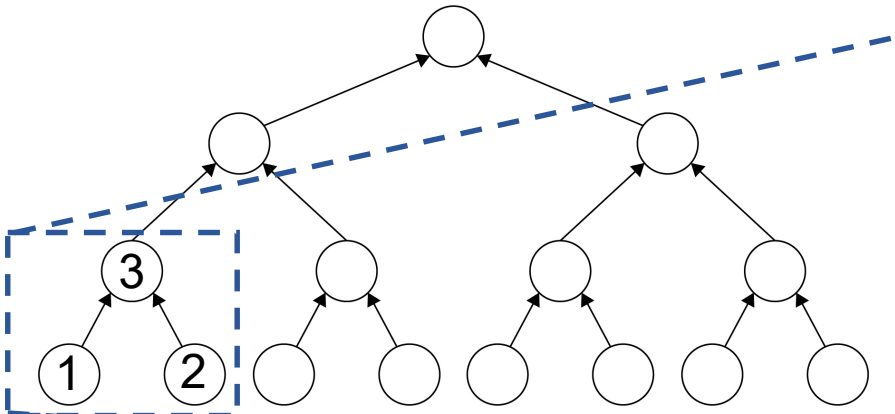
Solve $\mathbf{K\Phi = M\Phi\Lambda}$

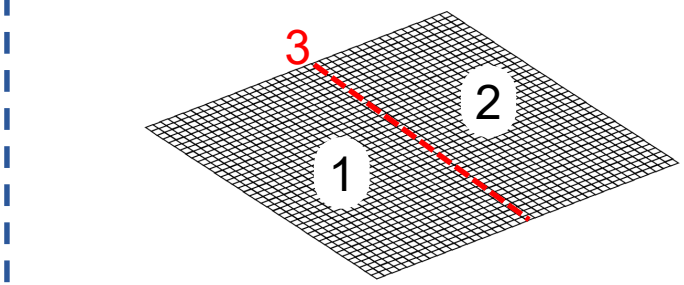
Step 1 of 4. Decompose the structure into substructures on multilevel.



AMLS method

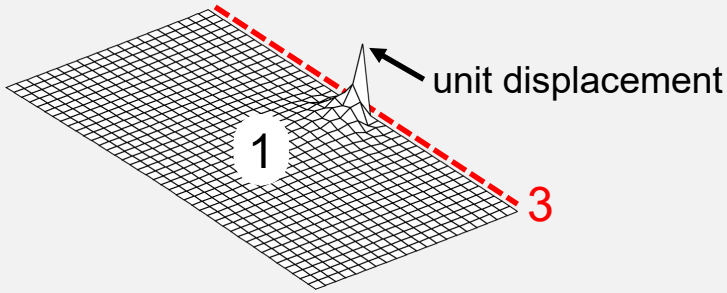
Step 2 of 4. Recursively transform the substructures.





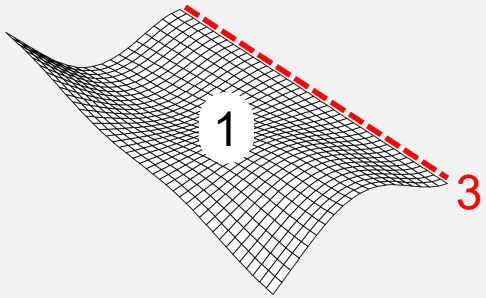
$$\mathbf{K}=\begin{bmatrix}\mathbf{K}_{11} & & \mathbf{K}_{13}\\ & \mathbf{K}_{22} & \mathbf{K}_{23}\\ \mathbf{K}_{13}^T & \mathbf{K}_{23}^T & \mathbf{K}_{33}\end{bmatrix} \quad \mathbf{M}=\begin{bmatrix}\mathbf{M}_{11} & & \mathbf{M}_{13}\\ & \mathbf{M}_{22} & \mathbf{M}_{23}\\ \mathbf{M}_{13}^T & \mathbf{M}_{23}^T & \mathbf{M}_{33}\end{bmatrix}$$

1. Determine the modes of the substructures.



$$\boldsymbol{\Psi}_{ij}=-\mathbf{K}_{ii}^{-1}\mathbf{K}_{ij}$$

constraint mode



$$\mathbf{K}_{ii}\mathbf{V}_i=\mathbf{M}_{ii}\mathbf{V}_i\mathbf{W}_i$$

eigen mode

$$\mathbf{T}^{(1)}=\begin{bmatrix}\mathbf{V}_1 & & \boldsymbol{\Psi}_{13}\\ & \mathbf{I} & \\ & & \mathbf{I}\end{bmatrix} \quad \mathbf{T}^{(2)}=\begin{bmatrix}\mathbf{I} & & \\ & \mathbf{V}_2 & \boldsymbol{\Psi}_{23}\\ & & \mathbf{I}\end{bmatrix}$$

2. Transform the substructures.

$$\bar{\mathbf{K}}=\mathbf{T}^T\mathbf{K}\mathbf{T}\ ,\ \bar{\mathbf{M}}=\mathbf{T}^T\mathbf{M}\mathbf{T} \quad \text{with} \quad \mathbf{T}=\mathbf{T}^{(1)}\mathbf{T}^{(2)}\cdots\mathbf{T}^{(n_s)}$$

Step 3 of 4. Solve the reduced eigenvalue problem.

1. Construct the reduced model.

$$\bar{\mathbf{K}} = \mathbf{T}^T\mathbf{K}\mathbf{T} = \begin{bmatrix} \mathbf{W}_1 & & & & \\ & \ddots & & & \\ & & \mathbf{W}_i & & \\ & & & \ddots & \\ & & & & \mathbf{W}_n \end{bmatrix}, \quad \bar{\mathbf{M}} = \mathbf{T}^T\mathbf{M}\mathbf{T} = \begin{bmatrix} \mathbf{I} & & & & \\ & \ddots & & & \\ & & \mathbf{I} & \bar{\mathbf{M}}_{ij} & \\ & sym. & & \ddots & \\ & & & & \mathbf{I} \end{bmatrix}$$

$\bar{\mathbf{K}}$: Reduced stiffness matrix

$\bar{\mathbf{M}}$: Reduced mass matrix

$\mathbf{T} = \prod_{i=1}^{n_s} \mathbf{T}^{(i)}$: AMLS transformation matrix

$\mathbf{T}^{(i)}$: Transformation matrix for the *i*th substructure

2. Solve the reduced eigenvalue problem.

$$\textbf{Solve} \quad \bar{\mathbf{K}}\mathbf{Q} = \bar{\mathbf{M}}\mathbf{Q}\bar{\mathbf{\Lambda}}$$

$\mathbf{Q}$: Reduced eigenvector matrix

$\bar{\mathbf{\Lambda}}$: Reduced eigenvalue matrix

Step 4 of 4. Compute approximate solution by back transformation.

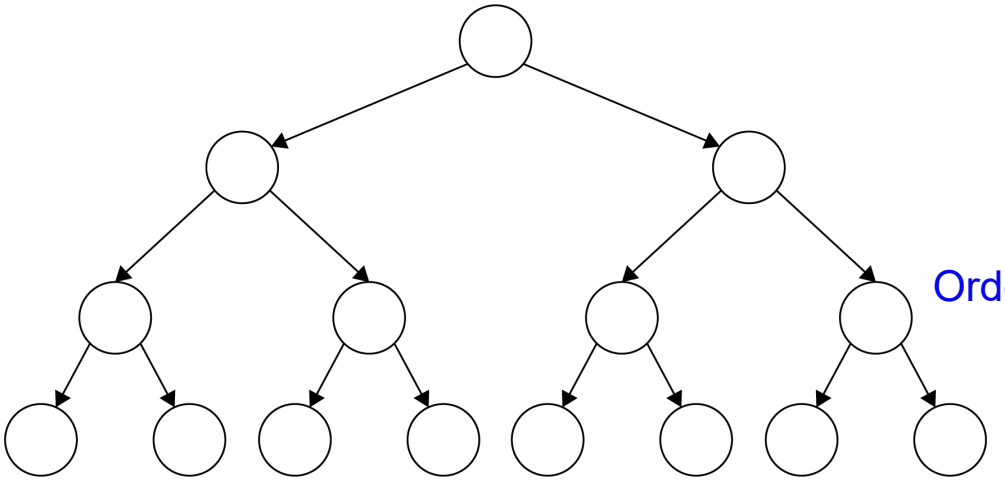
Approximate eigensolutions: $\boldsymbol{\Lambda} \approx \bar{\boldsymbol{\Lambda}}, \quad \boldsymbol{\Phi} \approx \bar{\boldsymbol{\Phi}} = \mathbf{TQ}$

$\boldsymbol{\Lambda}$: Eigenvalue matrix for the original model

$\boldsymbol{\Phi}$: Eigenvector matrix for the original model

$\bar{\boldsymbol{\Lambda}}$: Reduced eigenvalue matrix

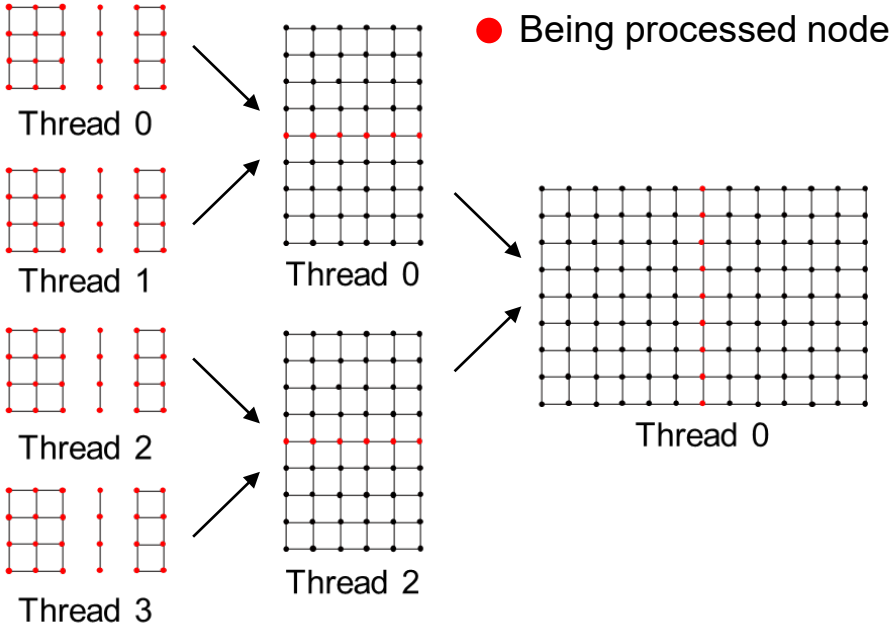
$\bar{\boldsymbol{\Phi}}$: Approximate eigenvalue matrix

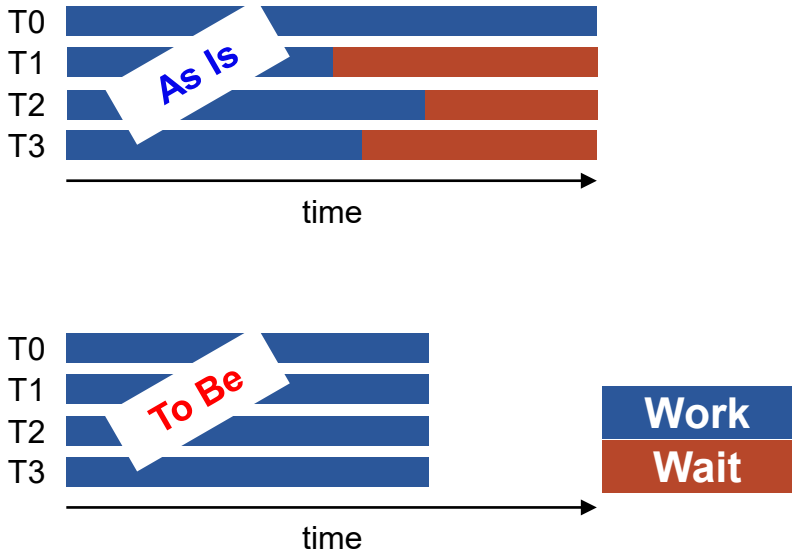


$$\bar{\boldsymbol{\Phi}} = \mathbf{TQ} = \mathbf{T}^{(1)}\mathbf{T}^{(2)} \cdots \mathbf{T}^{(n)}\mathbf{Q}$$

Order of matrix multiplications is from right to left

- **Parallel processing of AMLS method**





Load imbalance decreases the parallel efficiency.

- **Load imbalance is caused by**
  - Increase of idle threads
  - Time imbalance in substructure transformations

Related studies

- **Escaig et al. (1994)**
  - Multilevel domain decomposition method for parallel static condensation
  - Large interface problem

- **Kurc and Will (2007)**
  - Iterative repartitioining algorithm for condensation of substructures
  - Bottleneck caused by the interface problem

- **Kaplan (2001) / Bennighof and Lehoucq (2004)**
  - Automated multilevel substructuring method (AMLS)

Initial Substructures

Final Substructures w/ Scratch-Remap

- **Kim (2004)**
  - Parallel eigensolver for the reduced model obtained by the AMLS method
  - No parallelization of the AMLS transformation

- **Elssel and Voss (2004)**
  - Parallel version of the AMLS method
  - Load balancing problem

- **Yang et al. (2011)**
  - Multilevel approach for parallel implicit dynamic analysis
  - The maximum allowed imbalance among substructures set by a static load balancer
  - No significant improvement of scalability

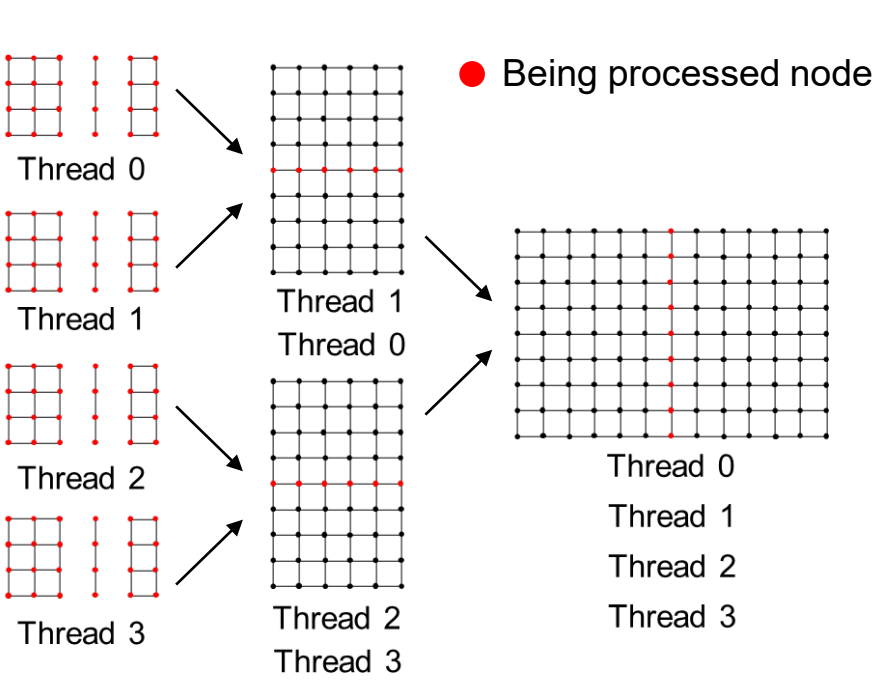
- **Yin et al. (2013)**
  - AMLS with the subspace iteration method & implicit back transformation algorithm

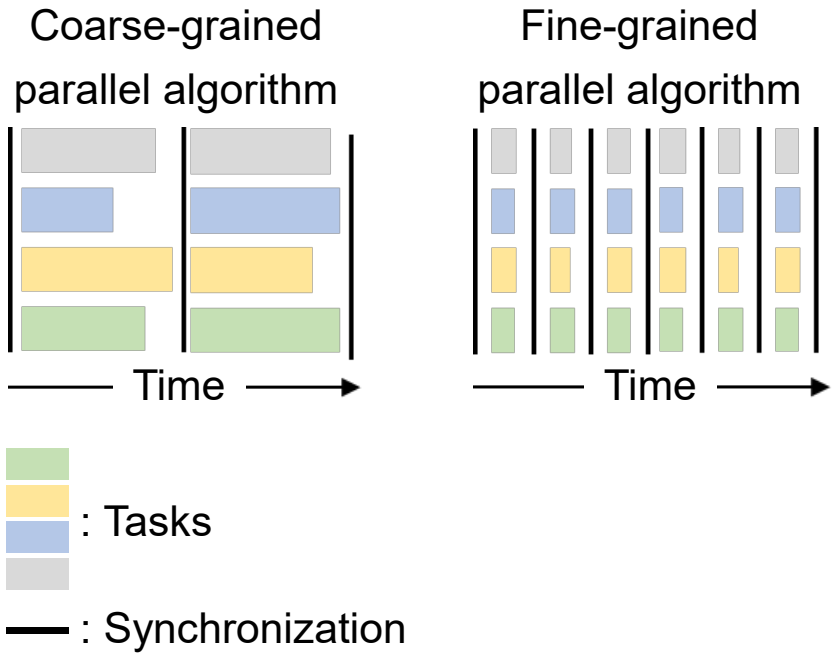
$$\bar{\Phi} = \mathbf{TQ} = \mathbf{T}^{(1)}\mathbf{T}^{(2)}\cdots\mathbf{T}^{(n)}\mathbf{Q} \qquad \bar{\Phi} = \mathbf{TQ} = \mathbf{T}^{(1)}\mathbf{T}^{(2)}\cdots\mathbf{T}^{(n)}\mathbf{Q}$$

Original Yin et al.

Proposed load balancing algorithm

- **Proposed algorithm**





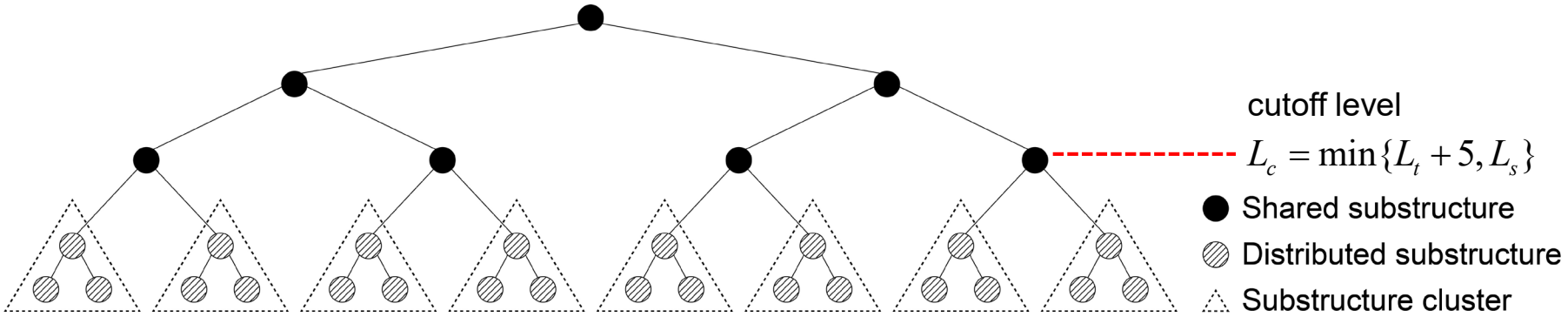
Reduce the number of idle threads.

- **Two-types of granularity**
  - Coarse-grained parallelism: transformation and back transformation **procedures** are split.
  - Fine-grained parallelism: transformation of **each substructure** is split.

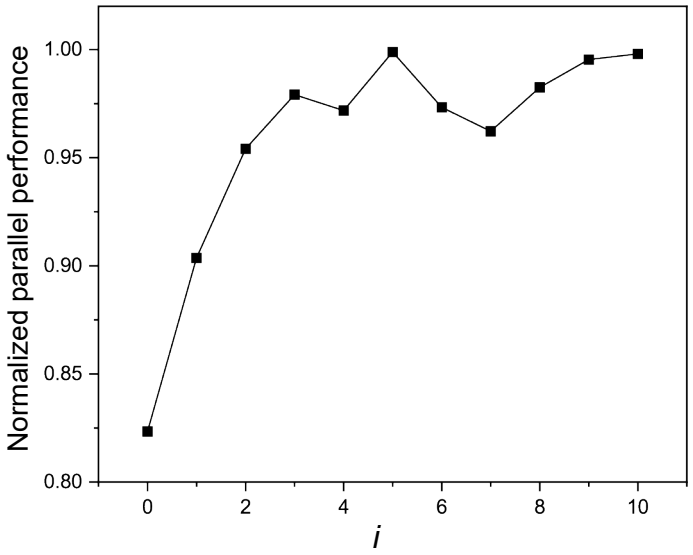
Proposed load balancing algorithm

- **Coarse-grained parallel algorithm**

  - Each thread transforms a substructure cluster or a shared substructure.



- **Effect of the cutoff level**



Normalized parallel performance: $\frac{\min\{t(j)\}}{t(i)} \in [0 \quad 1]$ for $i, j \in \mathbb{N}$

$t(i)$: elapsed time with $L_c = \min\{L_t + i, L_s\}$

$L_t = \min\{k \in \mathbb{Z} : \log_2 N_t \le k\}$

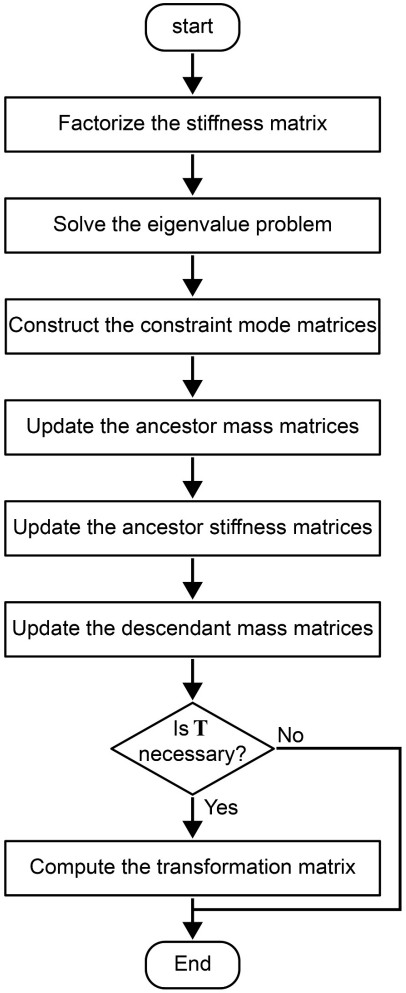
N_t : Number of threads used

L_s : Maximum level of substructures

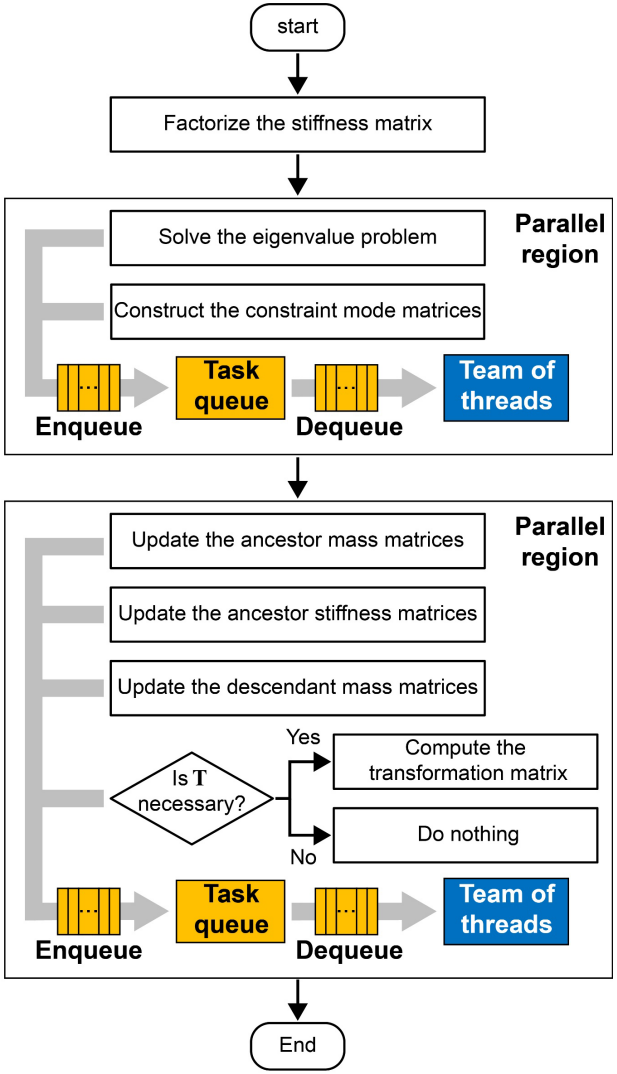
$\mathbb{Z}$: Set of all integers

Proposed load balancing algorithm

- **Fine-grained parallel algorithm**
  - Threads transform a shared substructure in parallel.



Serial algorithm



Fine-grained parallel algorithm

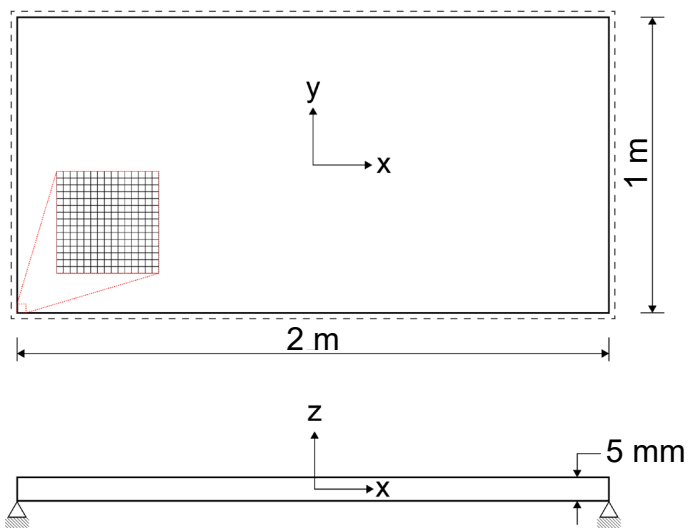
Numerical examples

- **Evaluation**
  - Speed-up factor for the fixed problem size (strong scaling): t_1 / t_i
  - t_i : wall clock time when using *i* threads
  - Number of threads used: 2, 4, 8, 16, 32

- **Previous method**
  - Only coarse-grained parallel algorithm
  - Number of substructure clusters = number of threads used

- **Proposed method**
  - An algorithm combining coarse-grained and fined-grained parallel algorithms
  - Number of substructure clusters set by the cutoff level $L_c = \min\{L_t + 5, L_s\}$

Effect of the number of DOFs of the original model



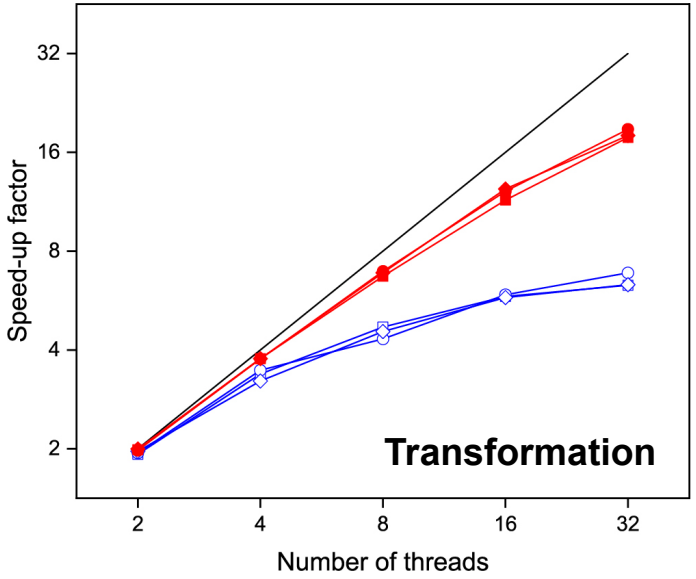
- **Structured meshes** $(2N \times N)$

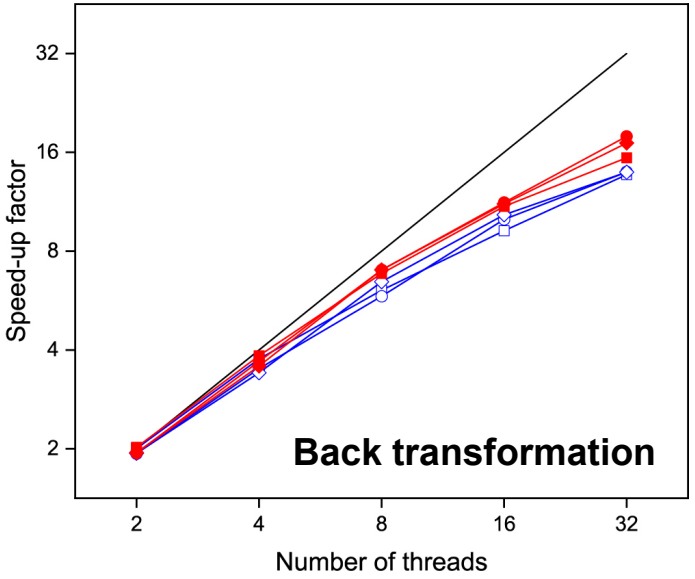
  Four-node shell finite elements
- **Boundary condition**

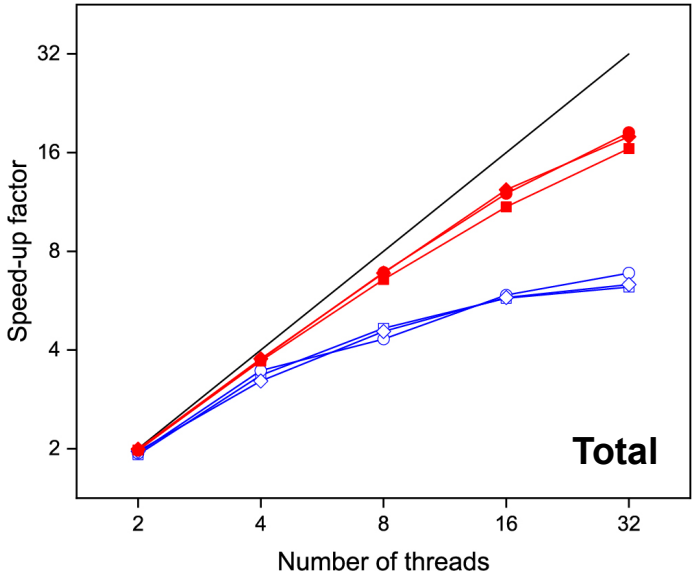
  Four-edges are simply supported.
- **150 eigensolutions sought**

- **Mesh A :** 256×128 mesh, 196614 DOFs → 1999 DOFs using 511 substructures
- **Mesh B :** 512×256 mesh, 786483 DOFs → 3406 DOFs using 2047 substructures
- **Mesh C :** 1024×512 mesh, 3145734 DOFs → 14706 DOFs using 8191 substructures

Numerical example (1)





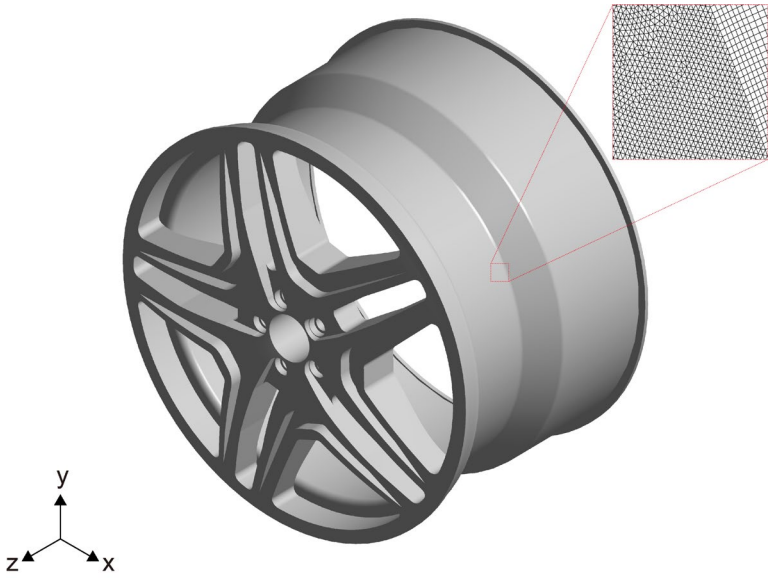


More efficient

More threads

- Previous (N = 128, 196614 DOFs)
- Previous (N = 256, 786438 DOFs)
- Previous (N = 512, 3145734 DOFs)
- Proposed (N = 128, 196614 DOFs)
- Proposed (N = 256, 786438 DOFs)
- Proposed (N = 512, 3145734 DOFs)

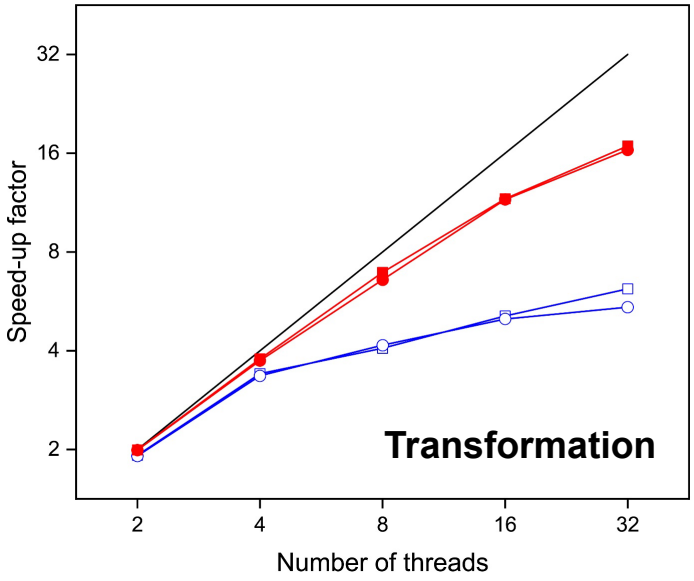
Effect of the number of substructures

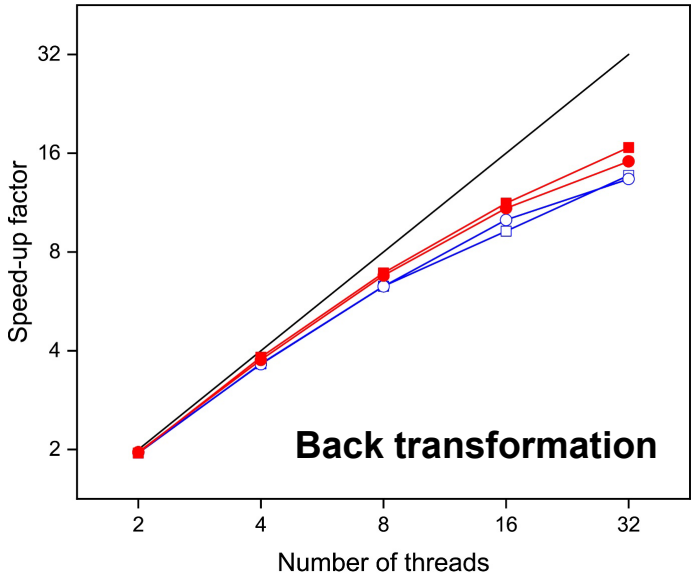


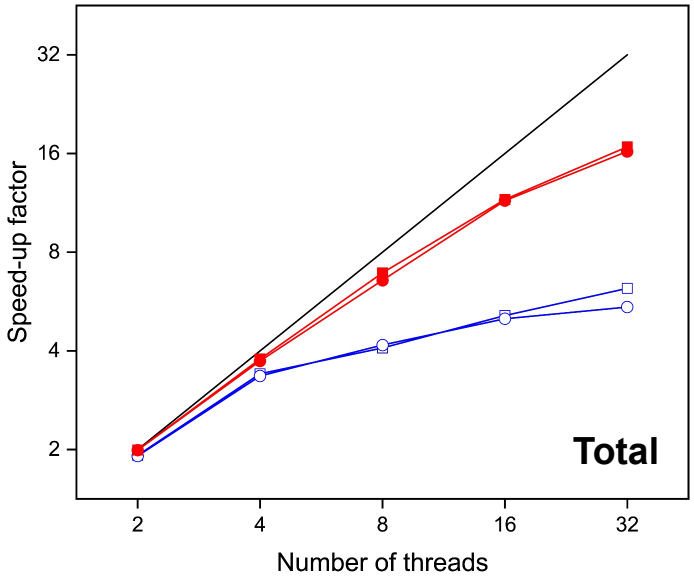
- **Mesh constructed by ANSYS**
  three-node shell finite elements
  & four-node shell finite elements
  & four-node tetrahedral elements
- **No boundary condition**
- **300 eigensolutions sought**

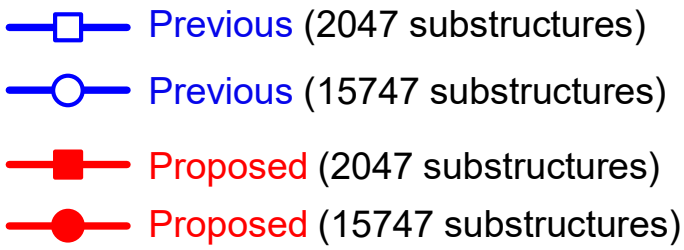
- **Partition A:** 3688653 DOFs → 6696 DOFs using 2047 substructures
- **Partition B:** 3688653 DOFs → 20096 DOFs using 15747 substructures

Numerical example (2)

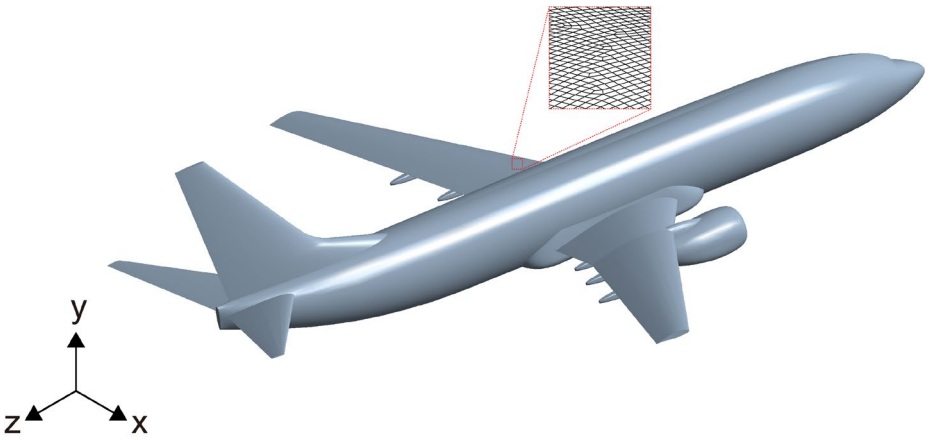








Effect of the number of DOFs of the reduced model



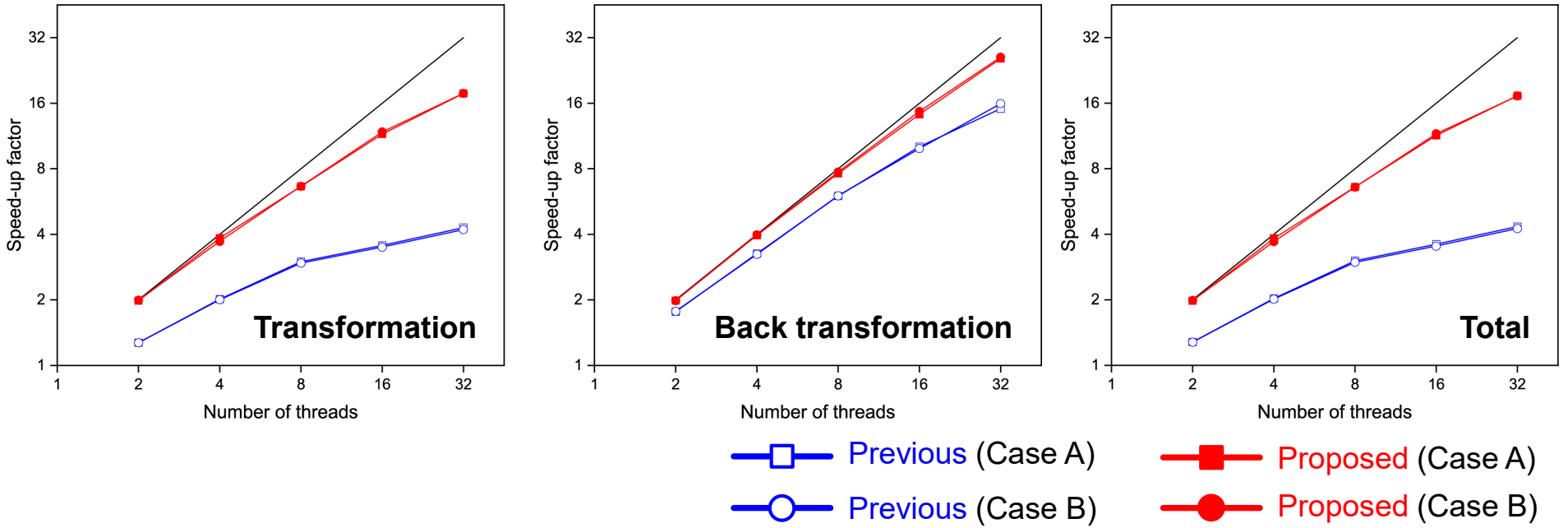
- **Mesh constructed by ANSYS**

  Three-node shell finite elements

  & four-node shell finite elements
- **No boundary condition**
- **Two cases of reduced models**
- **600 eigensolutions sought**

- **Case A:** 8462700 DOFs → 38489 DOFs using 32767 substructures
- **Case B:** 8462700 DOFs → 45449 DOFs using 32767 substructures

Numerical example (3)



In the previous method using 32 threads,

- 32 substructure clusters have the same number of distributed substructures.
- Each substructure cluster has approximately the same number of DOFs: 262k–264k.
- The relative difference between the min. and max. DOFs for each cluster is 0.7%.

In the proposed algorithm,

- Significant improvement of the parallel performance is achieved without repartitioning.

Closure (topic 1)

1. A **load balancing algorithm** consisting of two types of granularity for the parallel AMLS (PAMLS) method has been proposed.

2. In coarse-grained parallelism, the transformation and back transformation procedures are split into tasks using a given cutoff level.

3. Fine-grained parallelism is used to reduce the idle time for the transformation of shared substructures.

4. The proposed algorithm **significantly improved the efficiency** of the previous PAMLS method **without repartitioning**.

Topic 2

3. Coarse mesh projection for nonlinear model reduction

Proper orthogonal decomposition (POD)

Step 1 of 4. Collect snapshots of parametric configurations.

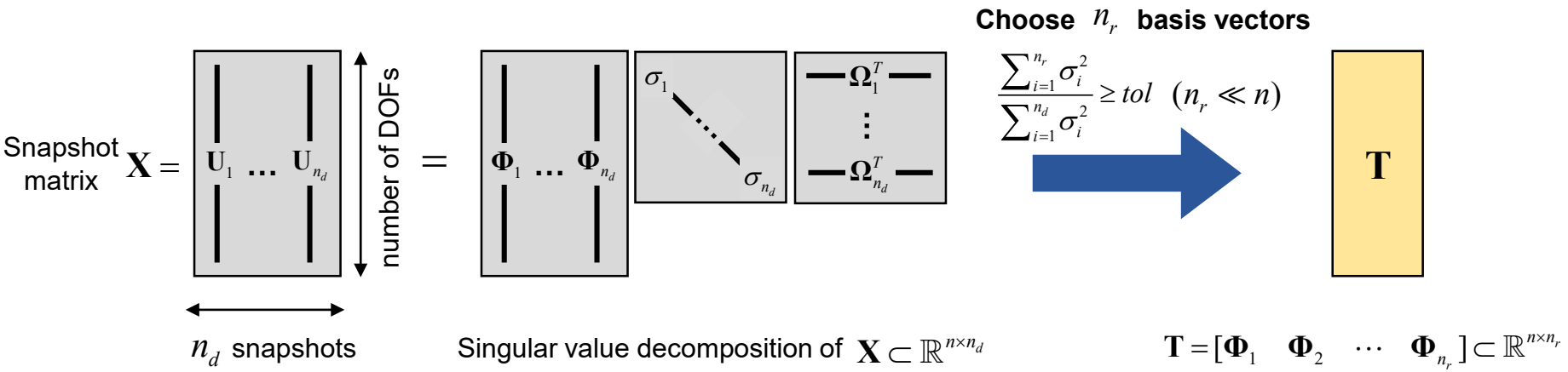
- (implicit) static analysis $^{t}\mathbf{K}\mathbf{U} = {}^{t+\Delta t}\mathbf{R} - {}^{t}\mathbf{F}$
- Implicit dynamic analysis $\mathbf{M}\,{}^{t+\Delta t}\ddot{\mathbf{U}} + {}^{t}\mathbf{K}\mathbf{U} = {}^{t+\Delta t}\mathbf{R} - {}^{t}\mathbf{F}$
- Explicit dynamic analysis $\mathbf{M}\,{}^{t}\ddot{\mathbf{U}} = {}^{t}\mathbf{R} - {}^{t}\mathbf{F}$

$\Leftrightarrow$ **Solve** $\mathbf{N}(\mathbf{U}) = \mathbf{0}$

$\mathbf{M} \in \mathbb{R}^{n\times n}$: time-independent mass matrix $^{t}\mathbf{F} \in \mathbb{R}^{n}$: internal force vector (nonlinear term)

$^{t}\mathbf{R} \in \mathbb{R}^{n}$: external load vector at time *t* $^{t}\mathbf{K} \in \mathbb{R}^{n\times n}$: stiffness matrix (nonlinear term)

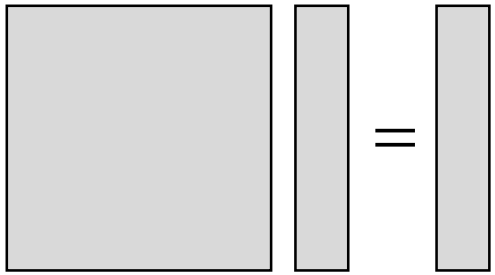
Step 2 of 4. Compute the low-dimensional basis vectors.



Singular value decomposition of $\mathbf{X} \subset \mathbb{R}^{n\times n_d}$

$\mathbf{T} = [\boldsymbol{\Phi}_1 \quad \boldsymbol{\Phi}_2 \quad \cdots \quad \boldsymbol{\Phi}_{n_r}] \subset \mathbb{R}^{n\times n_r}$

Step 3 of 4. Generate the reduced model.

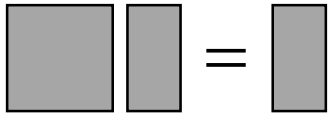


$$\mathbf{N}(\mathbf{U}) = \mathbf{0}$$

$n \times n$ original model

$$\mathbf{U} \approx \mathbf{U}_a = \mathbf{T}\bar{\mathbf{U}}$$

reduction



$$\mathbf{T}^T\mathbf{N}(\mathbf{U}_a) = \mathbf{0}$$

$n_r \times n_r$ reduced model

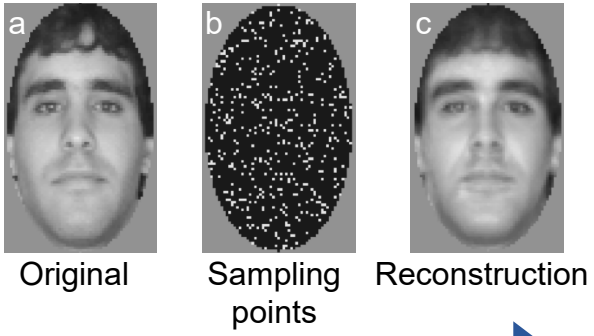
Step 4 of 4. Solve the reduced model $\mathbf{T}^T\mathbf{N}(\mathbf{U}_a) = \mathbf{0}$.

Computing the nonlinear terms depends on the original dimension

1. Lift $\bar{\mathbf{U}} \subset \mathbb{R}^{n_r}$ back to the original high-dimensional space $\mathbf{U}_a \subset \mathbb{R}^n$.
2. Evaluate nonlinear terms such as ${}^t\mathbf{K} \subset \mathbb{R}^{n \times n}$ and ${}^t\mathbf{F} \subset \mathbb{R}^n$.
3. Left-multiply by $\mathbf{T}^T$ as $\mathbf{T}^T\mathbf{N}(\mathbf{U}_a) = \mathbf{0}$.

Related studies

- **Everson and Sirovich (1995)**
  - A first sparse sampling scheme with POD modes called gappy POD
  - Random sampling points



$$\mathbf{F} \approx \boldsymbol{\Xi}\,\overline{\mathbf{F}}$$

$\mathbf{F} \in \mathbb{R}^{n}$: Face vector

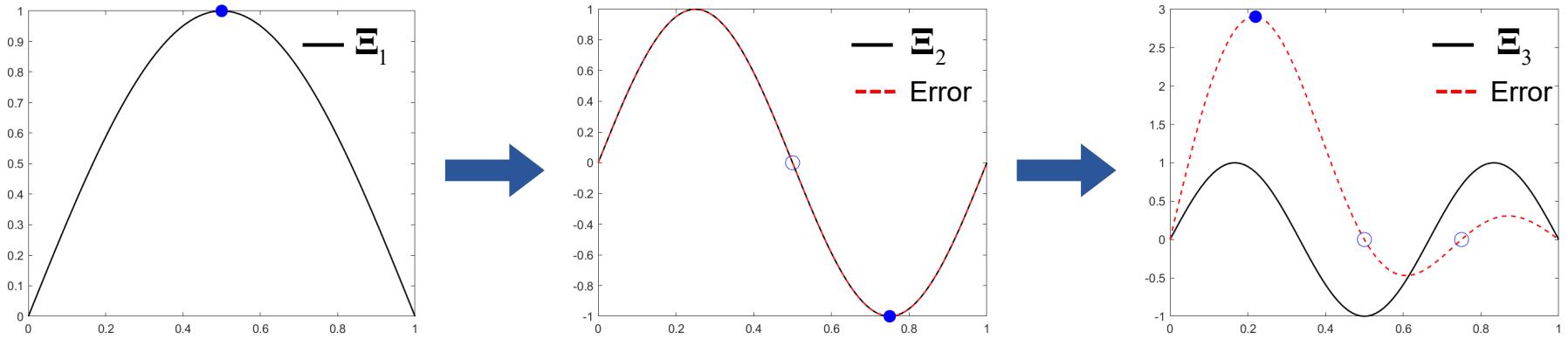
$\boldsymbol{\Xi} \in \mathbb{R}^{n \times n_n}$: POD basis vectors

$\overline{\mathbf{F}} \in \mathbb{R}^{n_n}$: POD coefficient vector

$\mathbf{P}^T \subset \mathbb{R}^{n_s \times n}$: Selection matrix

$$\mathbf{P}^T\mathbf{F} = \mathbf{P}^T\boldsymbol{\Xi}\overline{\mathbf{F}} \Rightarrow \overline{\mathbf{F}} \approx (\mathbf{P}^T\boldsymbol{\Xi})^{+}\mathbf{P}^T\mathbf{F} \Rightarrow \mathbf{F} \approx \boldsymbol{\Xi}(\mathbf{P}^T\boldsymbol{\Xi})^{+}\mathbf{P}^T\mathbf{F}$$

- **Chaturantabut and Sorensen (2010)**
  - Nonlinear model reduction vis discrete empirical interpolation method (DEIM)
  - Sampling point selection algorithm with a greedy approach
  - Instability in certain situations



Related studies

- **Peherstorfer et al. (2014)**
  - Localized variant of the DEIM (LDEIM)
  - Partitioned snapshots

- **Drmac and Gugercin (2016)**
  - DEIM using QR factorization with column pivoting (QDEIM)
  - Sharper error bound for the DEIM projection error
  - Instability in certain situations

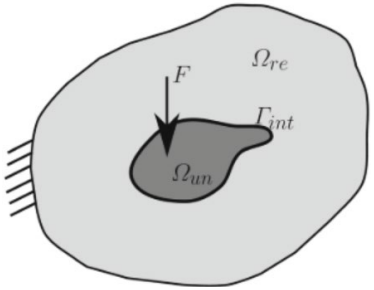
$$\Xi^T \begin{bmatrix} \mathbf{P}_1^T & \mathbf{P}_2^T & \dots & \mathbf{P}_n^T \end{bmatrix} = \mathbf{Q}\,\mathbf{R} \quad \Rightarrow \quad \mathbf{P} = [\mathbf{P}_1^T \quad \mathbf{P}_2^T \quad \cdots \quad \mathbf{P}_{n_n}^T]^T$$

Pivoting matrix

Selection matrix

- **Peherstorfer et al. (2020)**
  - QDEIM with a deterministic oversampling algorithm (GappyPOD+E)
  - Additional sampling points that minimize $\| (\mathbf{P}^T\Xi)^+ \|_2$ in $\mathbf{F} \approx \Xi(\mathbf{P}^T\Xi)^+\mathbf{P}^T\mathbf{F}$

- **Radermacher and Reese (2014) / Corigliano et al. (2015)**
  - Selective POD method by adaptive substructuring.
  - Limited application of local nonlinear problems

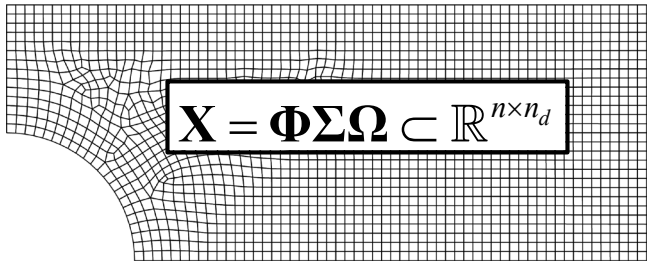


- **Baiges et al. (2019)**
  - A coarse mesh based reduced-order modeling via artificial neural networks

Proposed nonlinear model reduction

Step 1 of 4. Compute a low-dimensional basis.

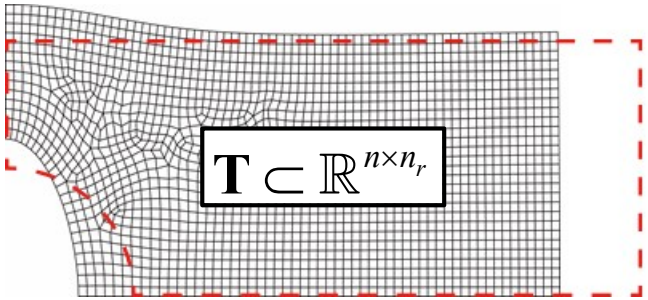
1. Extract the POD basis vectors from the original model.

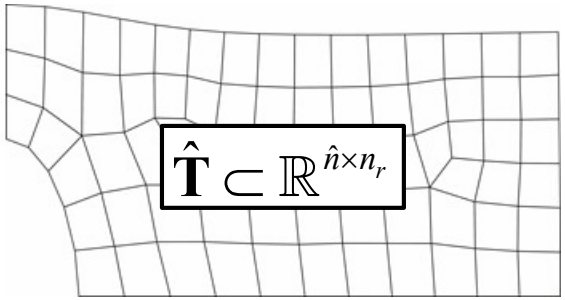


$$\mathbf{T} = [\mathbf{\Phi}_1 \quad \mathbf{\Phi}_2 \quad \cdots \quad \mathbf{\Phi}_{n_r}] \subset \mathbb{R}^{n \times n_r}$$

POD basis vectors for the original model

2. Compute the coarsened POD basis vectors using finite element interpolation.





POD basis vectors for the original model

Coarsened POD basis vectors

Finite element interpolation for a *q*-node element

$$\hat{\boldsymbol{\varphi}}_i = \sum_{j=1}^{q} h_j(\xi_1, \xi_2, \xi_3)\boldsymbol{\varphi}_j$$

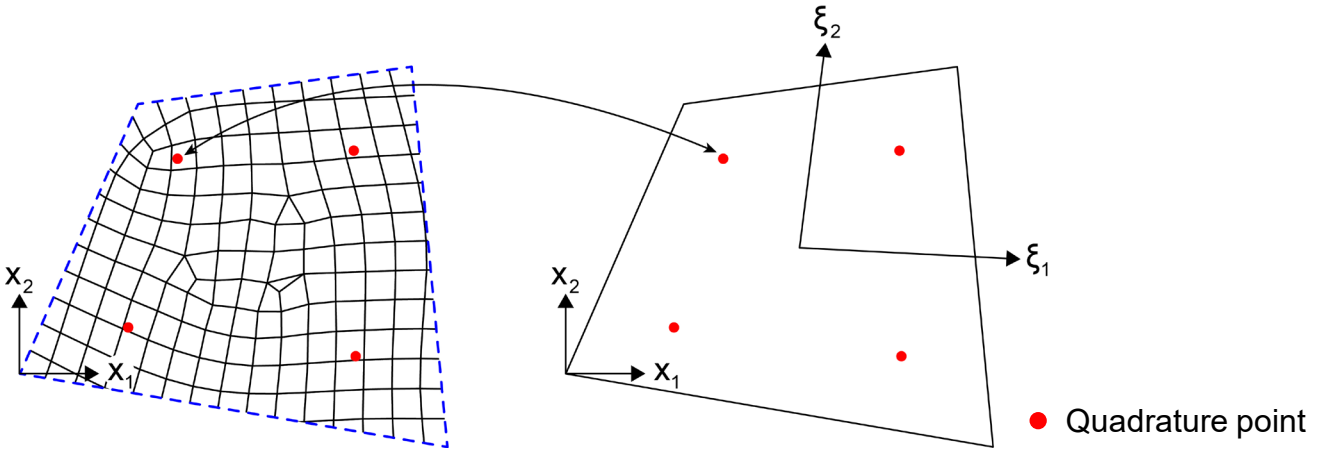
$h_j(\xi_1, \xi_2, \xi_3)$: shape functions with the natural coordinates

$\boldsymbol{\varphi}_j$: corresponding nodal value of $\mathbf{T}$

$\hat{\boldsymbol{\varphi}}_i$: *i*th nodal value of $\hat{\mathbf{T}}$

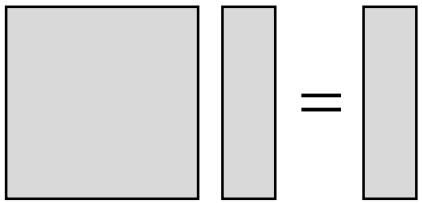
Proposed nonlinear model reduction

Step 2 of 4. Find the map between the coarse mesh model and the original model.



Fine elements Coarse element

Step 3 of 4. Reduce the coarse mesh model using the coarsened POD bases $\hat{\mathbf{T}}$.

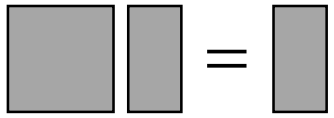


$$\hat{\mathbf{N}}(\hat{\mathbf{U}}) = \mathbf{0}$$

$\hat{n} \times \hat{n}$ coarse mesh model

$$\hat{\mathbf{U}} \approx \hat{\mathbf{U}}_a = \hat{\mathbf{T}}\tilde{\mathbf{U}}$$

reduction



$$\hat{\mathbf{T}}^T \hat{\mathbf{N}}(\mathbf{U}_a, \hat{\mathbf{U}}_a) = \mathbf{0}$$

$n_r \times n_r$ reduced system

$\mathbf{U}_a$: Approximate solution for the original model

$\hat{\mathbf{U}}_a$: Approximate solution for the coarse mesh model

Proposed nonlinear model reduction

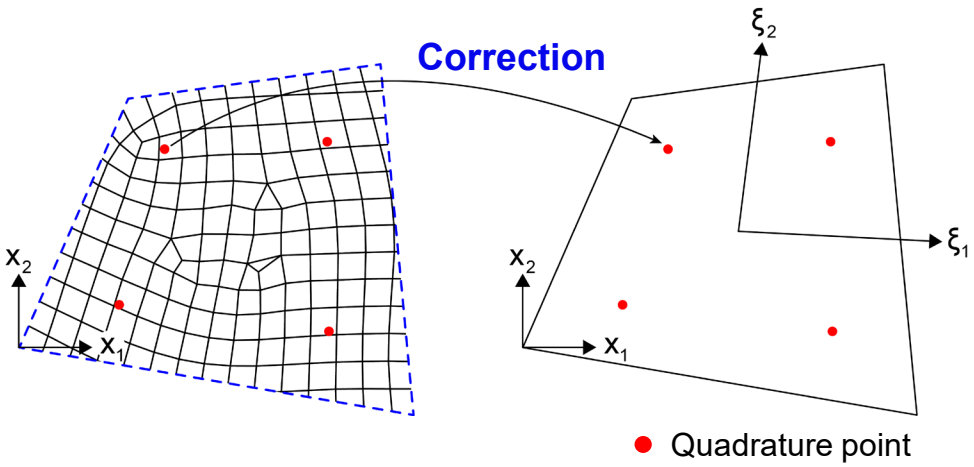
Step 4 of 4. Solve the reduced model.

1. Calculate the approximate solution for the original model by assuming $\tilde{\mathbf{U}} \approx \bar{\mathbf{U}}$.

$$\tilde{\mathbf{U}} \approx \bar{\mathbf{U}} \longrightarrow \mathbf{U} \approx \mathbf{T}\tilde{\mathbf{U}}$$

$\tilde{\mathbf{U}}$: Reduced solution for the coarse mesh model	$\mathbf{T}$: POD basis vectors for the original model
$\bar{\mathbf{U}}$: Reduced solution for the original model	$\mathbf{U}$: Solution for the original model

2. Correct the coarse mesh model.



For the *m*th element,

$${}^{t}\hat{\mathbf{F}}^{(m)} = \int_{{}^{0}\hat{V}^{(m)}} ({}^{t}\hat{\mathbf{B}}_{ij}^{(m)})^{T}\, {}^{t}S_{ij}^{(m)}\, d\, {}^{0}\hat{V}^{(m)}$$

$${}^{t}\hat{\mathbf{K}}^{(m)} = \int_{{}^{0}\hat{V}^{(m)}} ({}^{t}\hat{\mathbf{B}}_{ij}^{(m)})^{T}\, C_{ijkl}^{(m)}\, {}^{t}_{0}\hat{\mathbf{B}}_{kl}^{(m)} + {}^{t}\hat{\mathbf{N}}_{ij}^{(m)}\, {}^{t}S_{ij}^{(m)}\, d\, {}^{0}\hat{V}^{(m)}$$

: Quantities computed on the original model

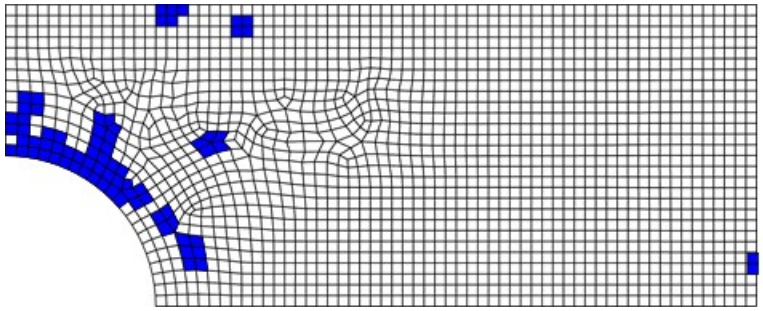
Proposed nonlinear model reduction

Discrete empirical interpolation

1. $\mathbf{T}$ and $\boldsymbol{\Xi}$ are generated.

2. $\mathbf{T}^T\mathbf{N}(\mathbf{U}_a)=\mathbf{0}$ is solved.

3. Nonlinear terms are evaluated on some elements of the original model.



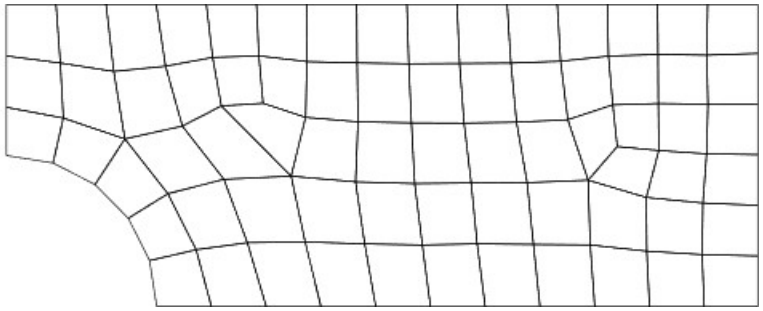
4. Nonlinear terms are approximated.

Coarse mesh projection

1. $\mathbf{T}$ and $\hat{\mathbf{T}}$ are generated.

2. $\hat{\mathbf{T}}^T\hat{\mathbf{N}}(\mathbf{U}_a,\hat{\mathbf{U}}_a)=\mathbf{0}$ is solved.

3. Nonlinear terms are evaluated on the coarse mesh model.



4. Coarse mesh model is corrected using the approximate solution.

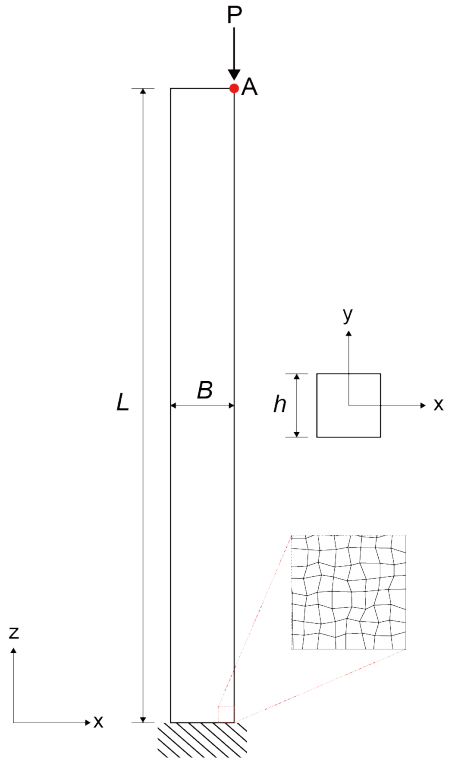
$\mathbf{T}$: POD basis vectors for the displacement

$\boldsymbol{\Xi}$: POD basis vectors for the internal force

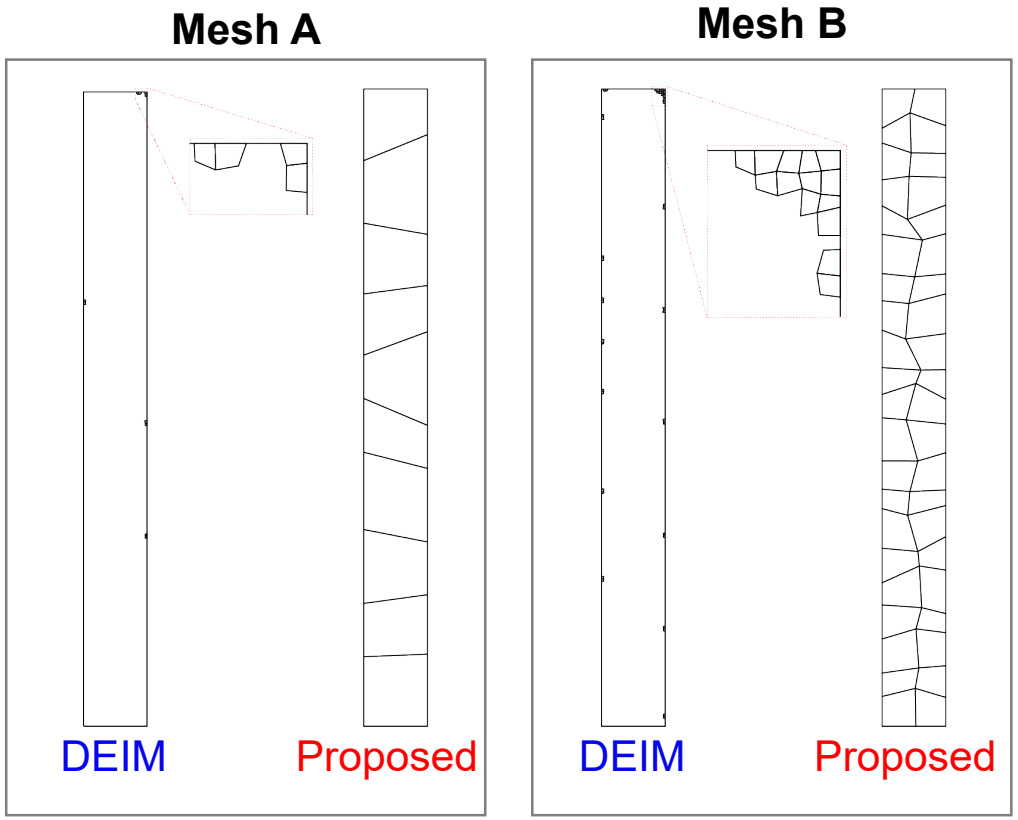
$\hat{\mathbf{T}}$: Coarsened POD basis vectors

Numerical example (1)

- **2D column problem (geometrically nonlinear static analysis)**



Original model: 10240 elements

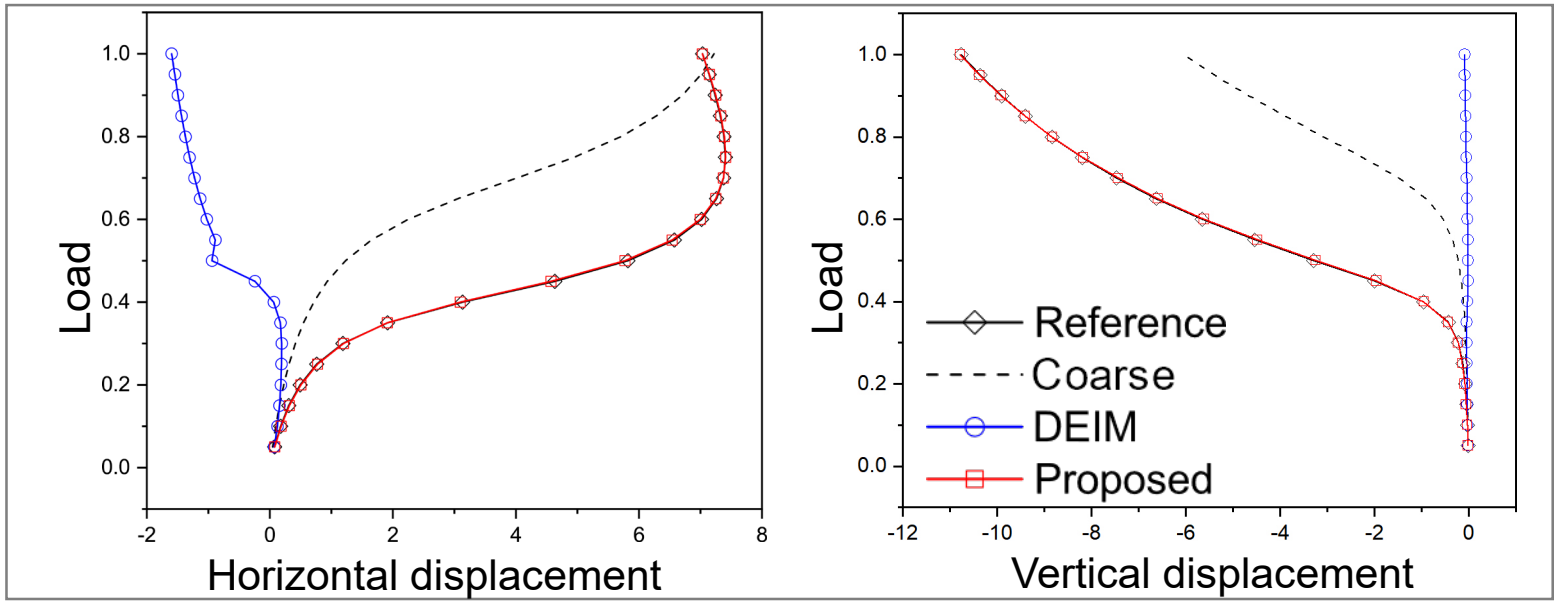


- **Number of snapshots:** 84
- **DOFs:** 21120 DOFs → 5 DOFs (i.e. 5 POD basis vectors are used.)
- **Number of elements used for evaluating nonlinear terms:** Mesh A: 10
  Mesh B: 42

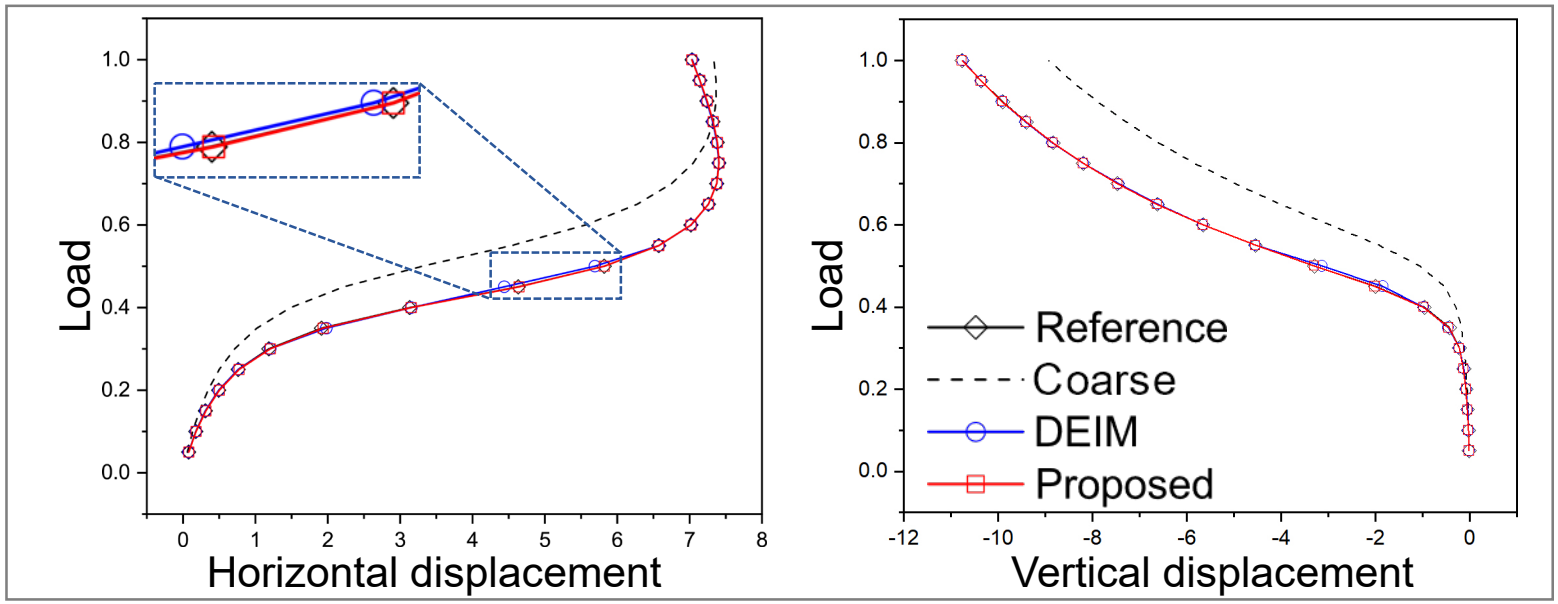
Numerical example (1)

- **Load-displacement curves (evaluation parameter = snapshot parameter)**

Mesh A
(10 elements)

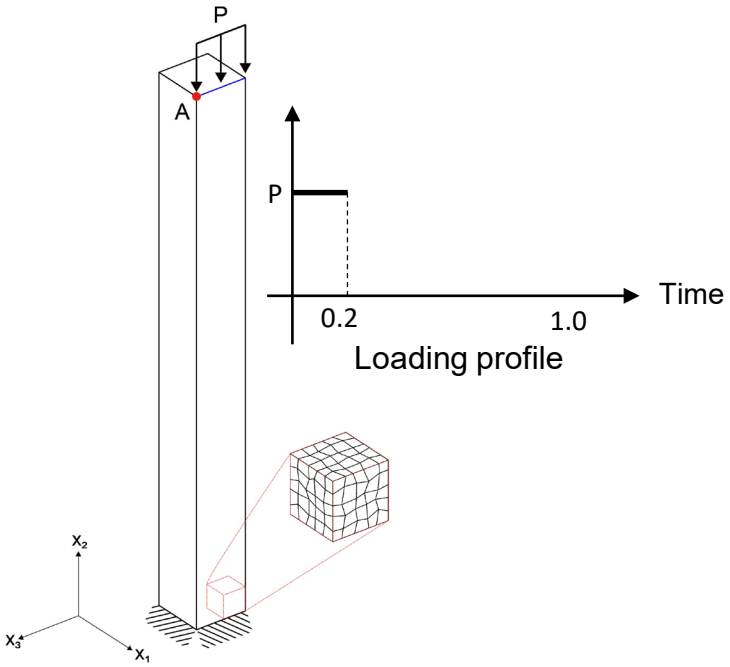


Mesh B
(42 elements)

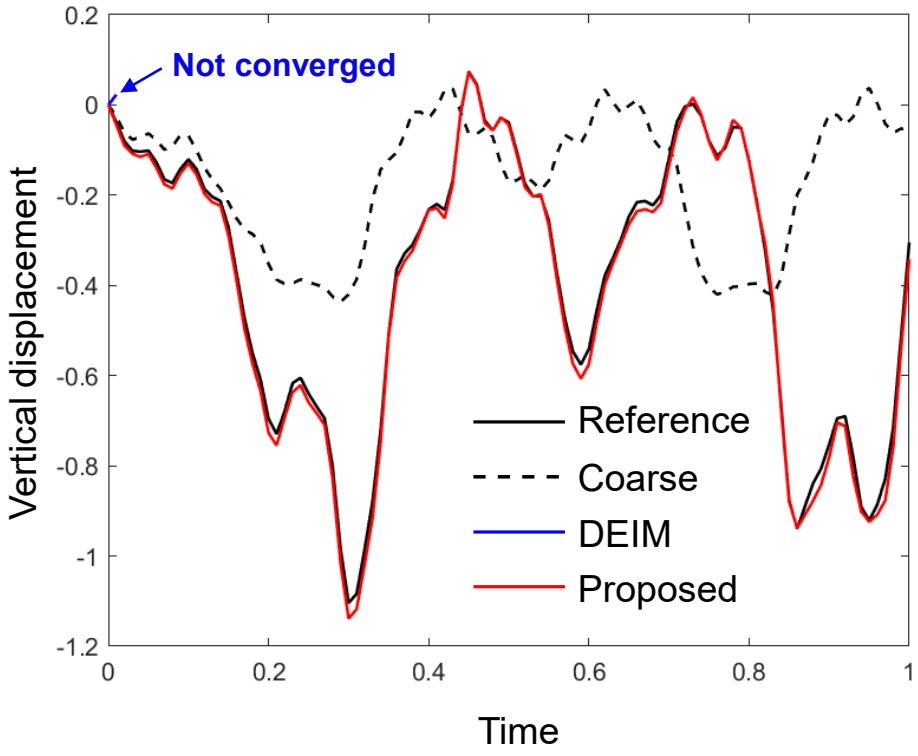


Numerical example (2)

- **3D column problem (geometrically nonlinear dynamic analysis)**



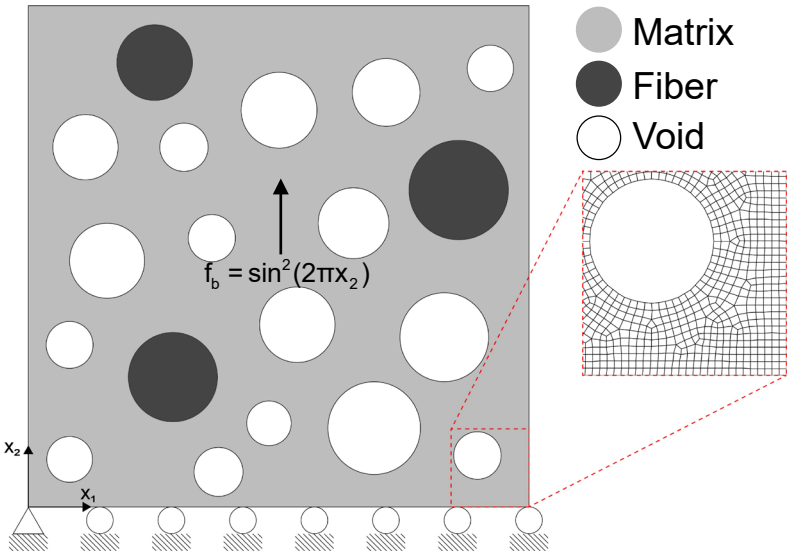
Original model: 8019 elements



- **Number of snapshots:** 710 (2 cases of P = 0.25 & P = 1)
- **DOFs:** 29700 DOFs → 22 DOFs (i.e. 22 POD basis vectors are used.)
- **Number of elements used for evaluating nonlinear terms:** 48
- **Evaluation:** Case of P = 0.625

Numerical example (3)

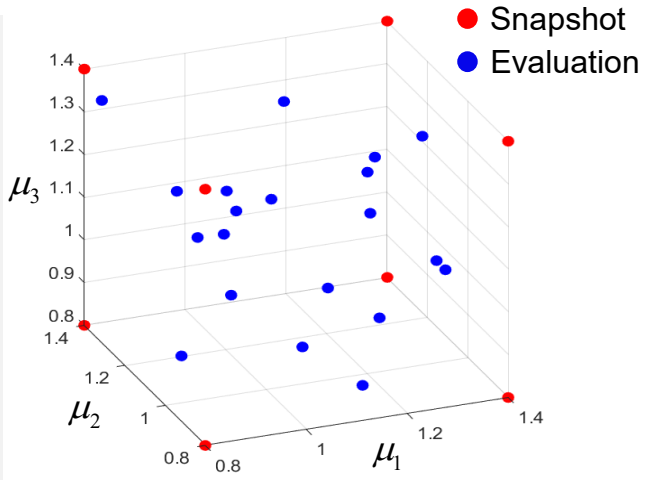
- **Heterogeneous structure problem (elastic-plastic static analysis)**



Original model: 25825 elements

- **Input parameters**
  - Young's modulus E
  - Initial yield stress σ_{yv}
  - Hardening modulus H
- **Material property (isotropic linear hardening)**
  - Matrix: $E = 70\,\mu_1 GPa$, $\sigma_{yv} = 70\,\mu_2 MPa$, $H = 14\,\mu_3 GPa$
  - Fiber: $E = 200\,\mu_1 GPa$, $\sigma_{yv} = 200\,\mu_2 MPa$, $H = 50\,\mu_3 GPa$

- **Number of snapshots:** 553 (8 cases; red dots)
- **DOFs:** 53323 DOFs → 37 DOFs
- **Number of elements used**
  - Mesh A: 826 (GappyPOD+E) & 805 (proposed)
  - Mesh B: 1644 (GappyPOD+E) & 1629 (proposed)
- **Evaluation:** 20 random cases (blue dots)

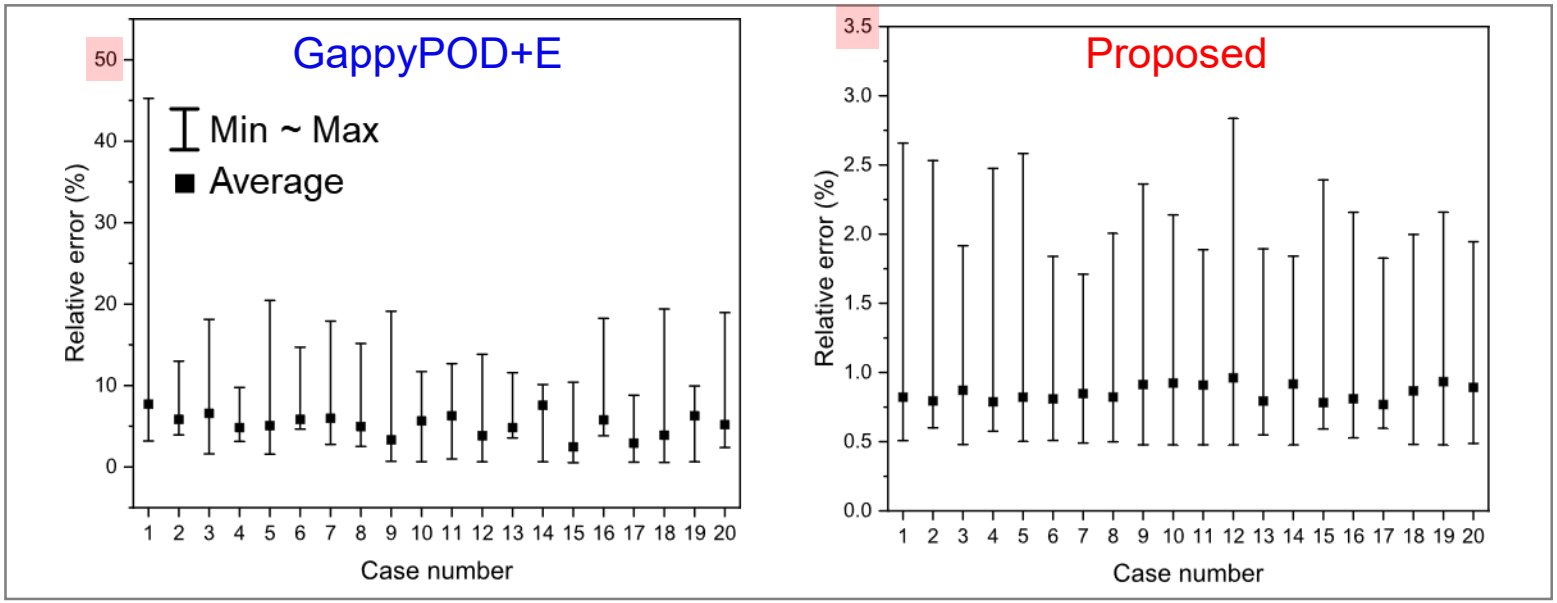


Parameter space

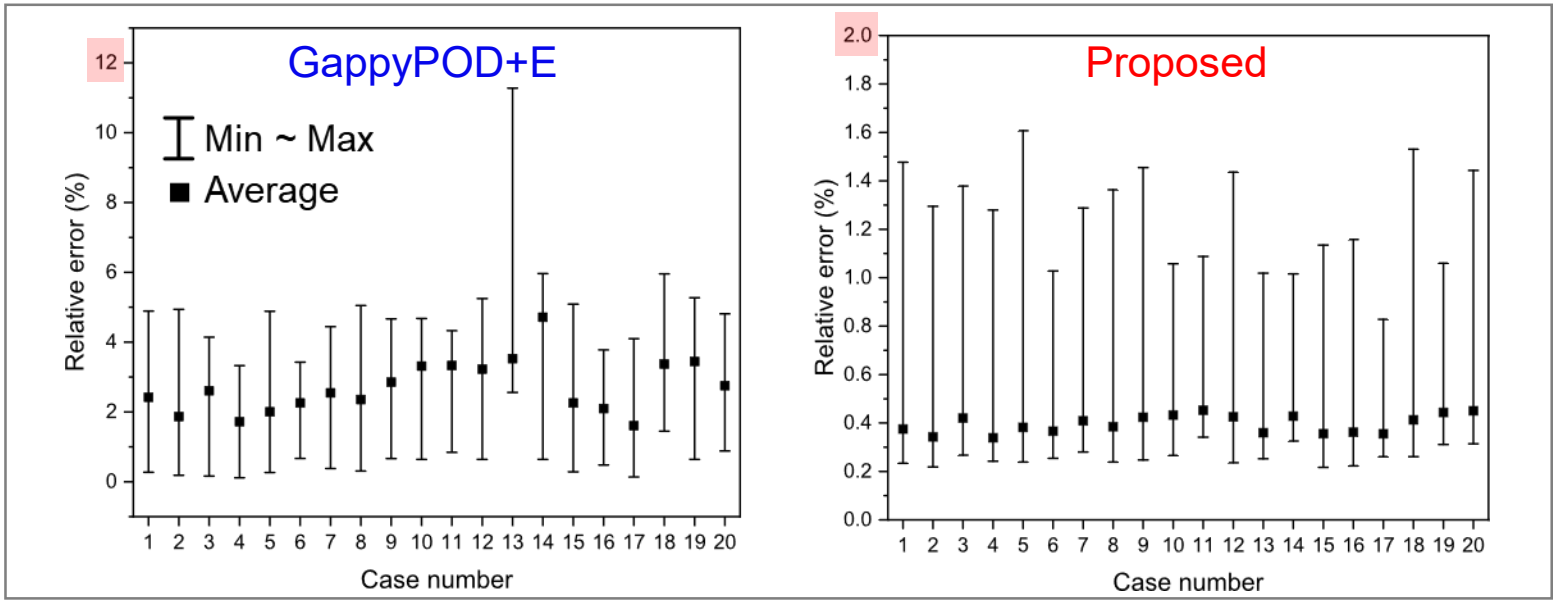
Numerical example (3)

- **Relative displacement error** $\left\| {}^{t}\mathbf{U} - {}^{t}\mathbf{U}_{approximation} \right\|_2 \Big/ \left\| {}^{t}\mathbf{U} \right\|_2$

Mesh A



Mesh B



Closure (topic 2)

1. The **coarse mesh projection** method for nonlinear model reduction has been proposed.

2. A structure is modeled by a coarse mesh model, where its nodes do not need to coincide with those of the original model.

3. Nonlinear terms are only computed on the quadrature points of the coarse mesh model.

4. The proposed method provided **more accurate solution than the DEIM and GappyPOD+E**.

4. Conclusions & future works

Conclusions

1. Effective methods were developed in the projection procedure of linear and nonlinear model reduction.

2. For linear model reduction, a load balancing algorithm for the parallel AMLS (PAMLS) method has been proposed.

3. The load balancing algorithm significantly improved the efficiency of the previous PAMLS method without mesh repartitioning.

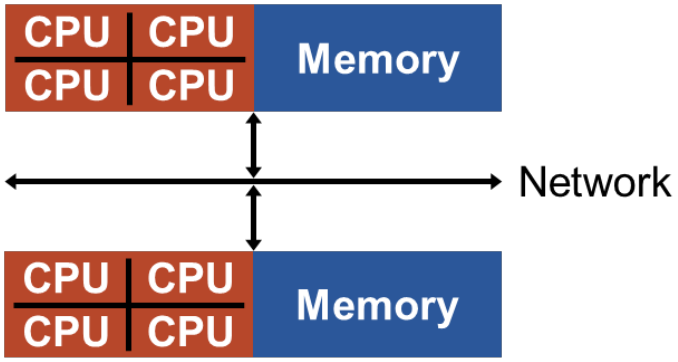
4. For nonlinear model reduction, the coarse mesh projection method has been proposed.

5. The proposed method provided more accurate solutions than the DEIM and GappyPOD+E.

Future works

- **Linear model reduction (topic 1)**

- Distributed memory system

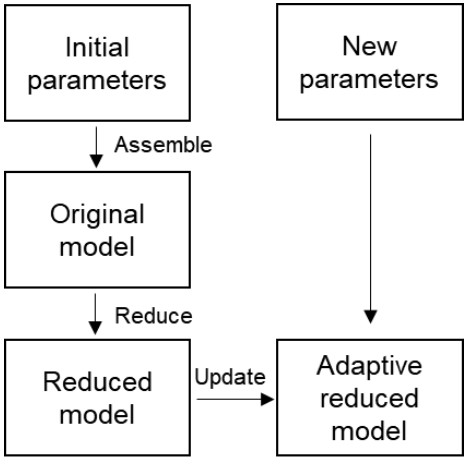


- Mesh partitioning algorithm

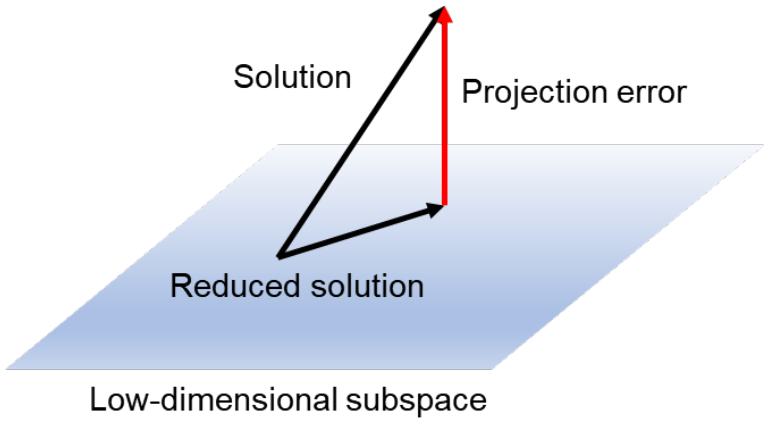
https://fesom.de/technology/libraries/metis/

- **Nonlinear model reduction (topic 2)**

- Online adaptive model reduction



- Error bound & estimation



감사합니다.