대형 유한요소 모델의 동적 해석을 위한 모델축소기법에 관한 연구

On the model reduction methods for dynamic analysis of large finite element models

Seung-Hwan, Boo

2016.05.18

KAIST

Department of Mechanical Engineering

Contents

1. Introduction

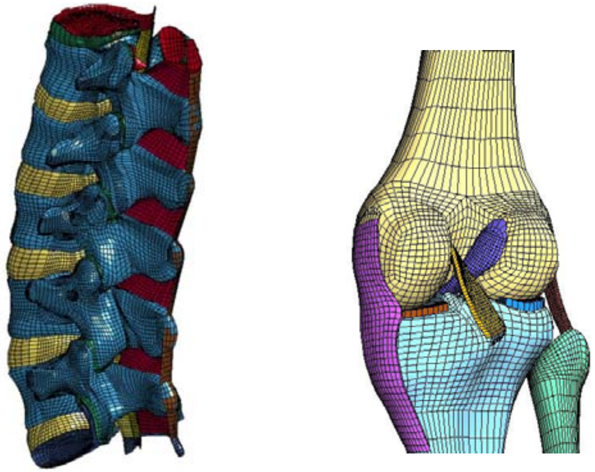
FEM in engineering fields

Offshore Engineering

Aerospace Engineering

Earthquake Engineering

Impact Engineering



BIO Engineering

Today, FEM is closely related with the <u>human safety</u> and <u>health</u> in real life !

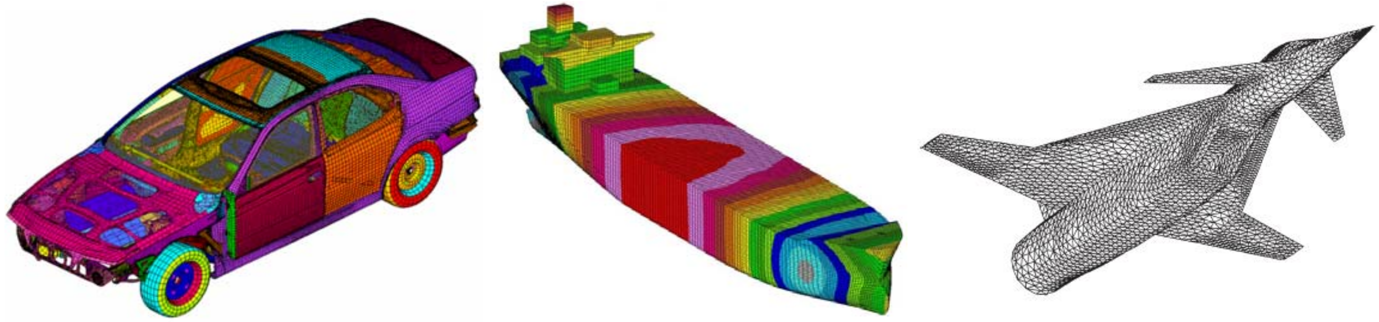
Motivations of model reduction

<u>Finite element model</u>

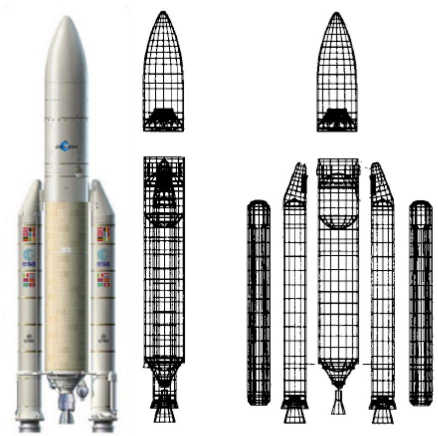
Complexity ↑

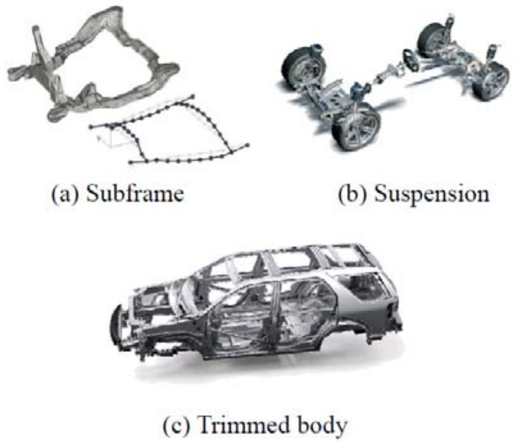
Model size (M_g, K_g) ↑

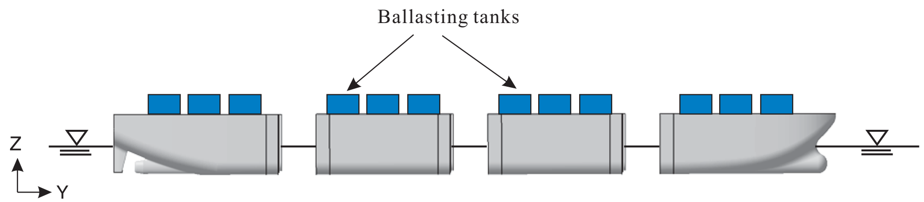
Computing time ↑



Reducing the computational cost has been an important issue.





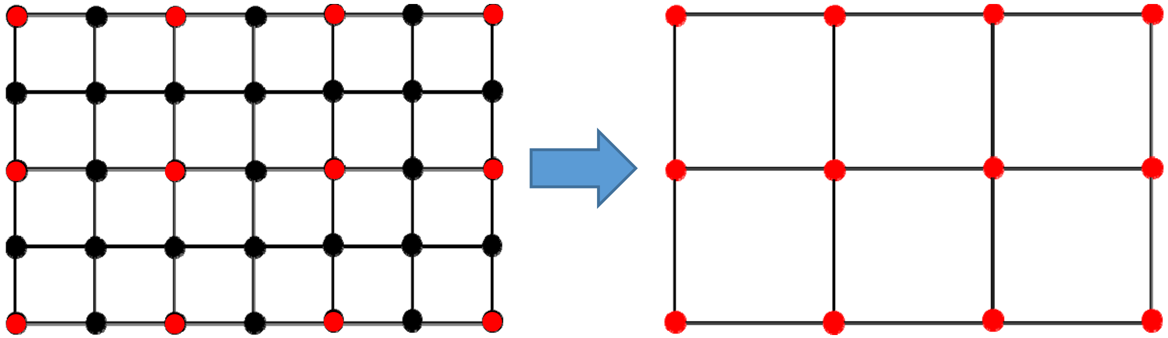


Separate into many pieces and reduce every pieces: "Divide and Conquer" paradigm

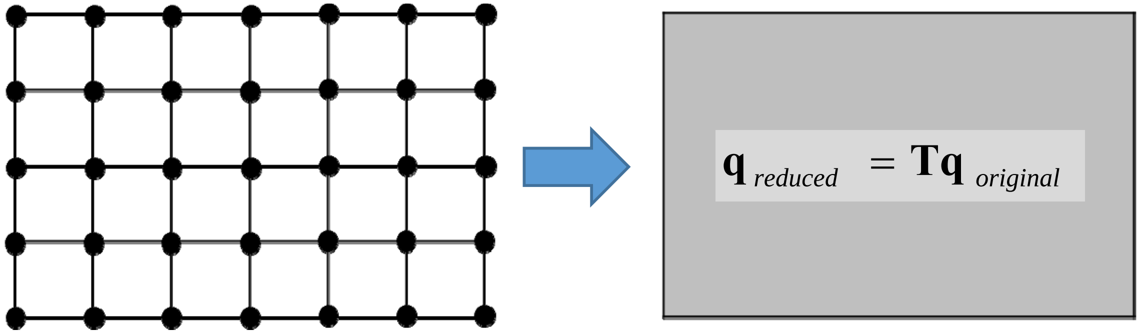
Model reduction methods have been studied for more than 50 years.
Still important research field.

Model reduction concepts

❖ **There are 2 ways to make the reduced model.**



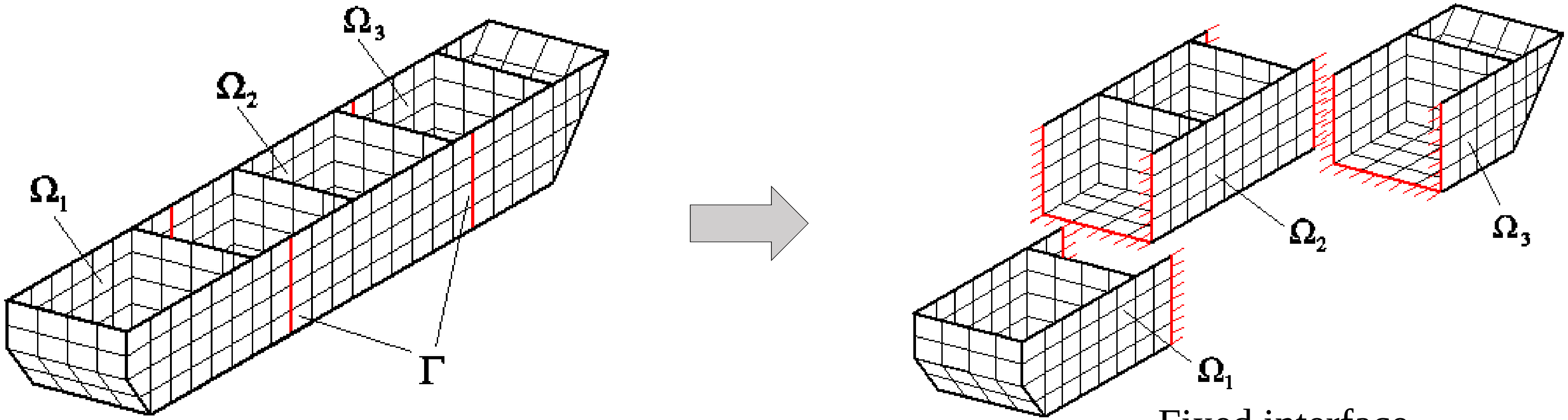
1. DOFs based reduction methods:
Ex) Guyan reduction, IRS method, Super-element etc..



2. Modes based reduction methods:
Ex) CB method, AMLS method etc..

The Craig-Bampton method

Step 1. Partitioning and interface handling



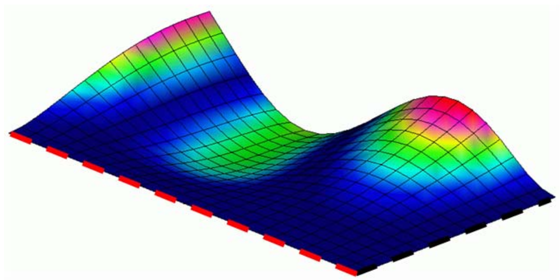
Partitioned structure

Fixed interface
(e.g. CB method, AMLS method)

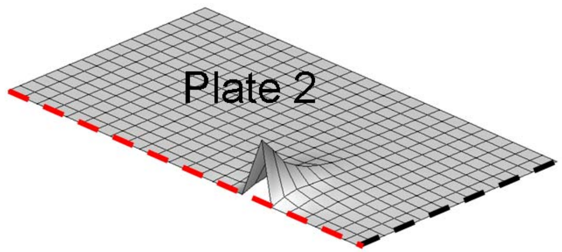
Step 2. Construct transformation matrix

$$\mathbf{T}_0 = [\boldsymbol{\Phi} \quad \boldsymbol{\Psi}],$$

$\boldsymbol{\Phi}$: Fixed interface normal mode



$\boldsymbol{\Psi}$: Interface constraint mode



The Craig-Bampton method

✓ Fixed interface normal modes

$$\mathbf{K}^{(i)}\mathbf{\Phi}^{(i)} = \mathbf{\Lambda}^{(i)}\mathbf{M}^{(i)}\mathbf{\Phi}^{(i)}, \quad \mathbf{\Phi}^{(i)} = [\mathbf{\Phi}_i^d \quad \mathbf{\Phi}_i^r]$$

s : substructural
b : interface boundary
d : dominant
r : residual

$$\Rightarrow \quad \mathbf{\Phi} = \begin{bmatrix} \mathbf{\Phi}_s \\ \mathbf{0}_b \end{bmatrix}, \quad \mathbf{\Phi}_s = [\mathbf{\Phi}_s^d \quad \mathbf{\Phi}_s^r]$$

Step 3. Reduced transformation matrix

$$\mathbf{T}_0 = \left[\begin{array}{cc:c} \mathbf{\Phi}_s^d & \mathbf{\Phi}_s^r & \mathbf{\Psi}_c \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_b \end{array}\right] \quad \Rightarrow \quad \overline{\mathbf{T}}_0 = \left[\begin{array}{c:c} \mathbf{\Phi}_s^d & \mathbf{\Psi}_c \\ \mathbf{0} & \mathbf{I}_b \end{array}\right]$$

Truncated :
Large amount of global modes

: Reduced transformation matrix

[Note]

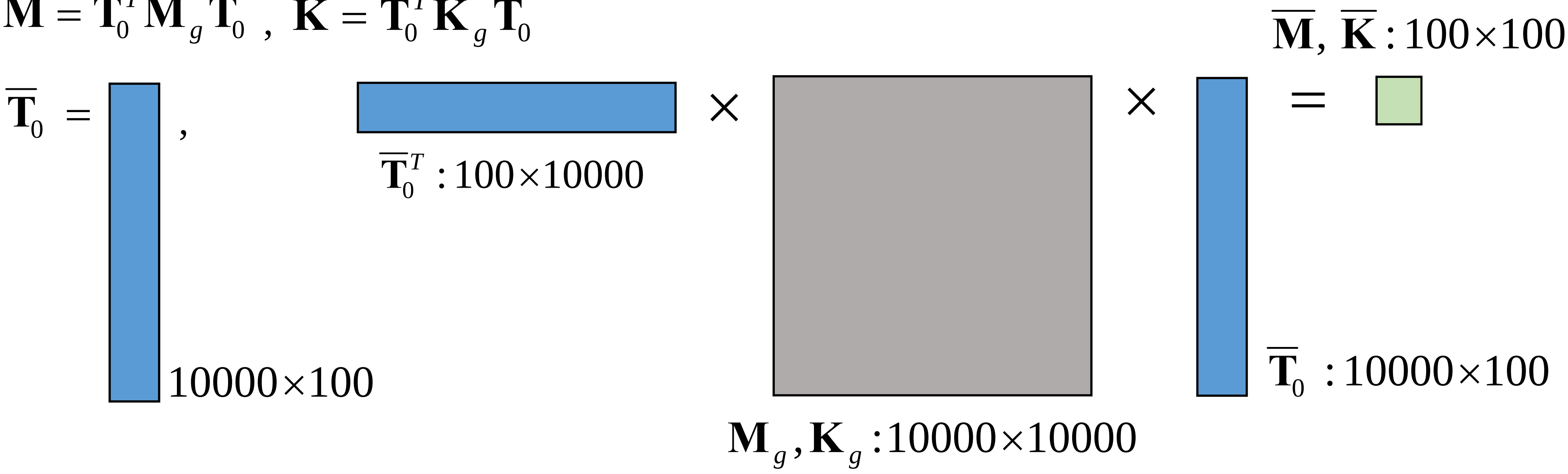
More accurate method → Depend on transformation matrix.

The Craig-Bampton method

Step 4. **Rayleigh-Ritz procedure**

$\overline{\mathbf{M}}, \overline{\mathbf{K}}$ is much smaller than $\mathbf{M}_g, \mathbf{K}_g$.

$$\overline{\mathbf{M}} = \overline{\mathbf{T}}_0^T \mathbf{M}_g \overline{\mathbf{T}}_0 \ , \ \overline{\mathbf{K}} = \overline{\mathbf{T}}_0^T \mathbf{K}_g \overline{\mathbf{T}}_0$$



Step 5. Reduced eigenvalue problem

$$\overline{\mathbf{K}}(\overline{\boldsymbol{\varphi}})_i = \bar{\lambda}_i \overline{\mathbf{M}}(\overline{\boldsymbol{\varphi}})_i \Rightarrow \textbf{Approximated eigenvalue } \bar{\lambda}_i$$

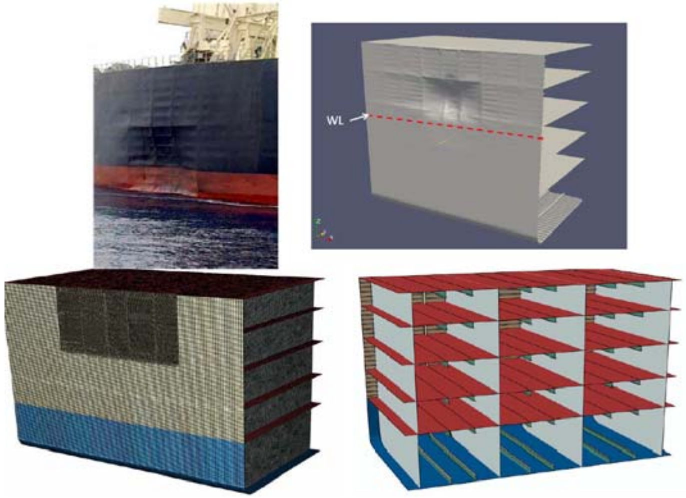
Step 6. Back transformation procedure

$$(\boldsymbol{\varphi}_g)_i \approx (\overline{\boldsymbol{\varphi}}_g)_i = \overline{\mathbf{T}}_0 (\overline{\boldsymbol{\varphi}}_p)_i$$

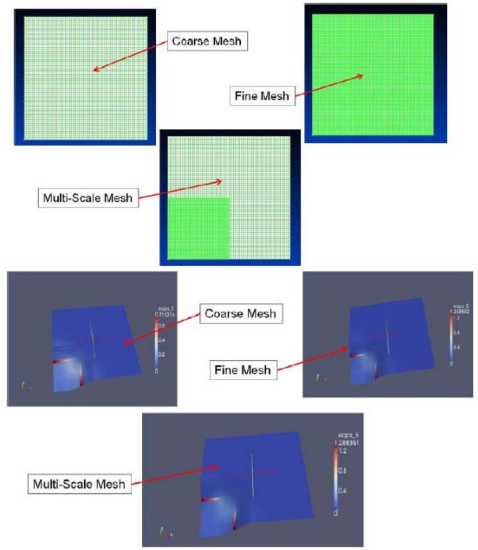
$\overline{\mathbf{T}}_0$ is a key factor in model reduction.

Our Aims → Accurate & Fast.

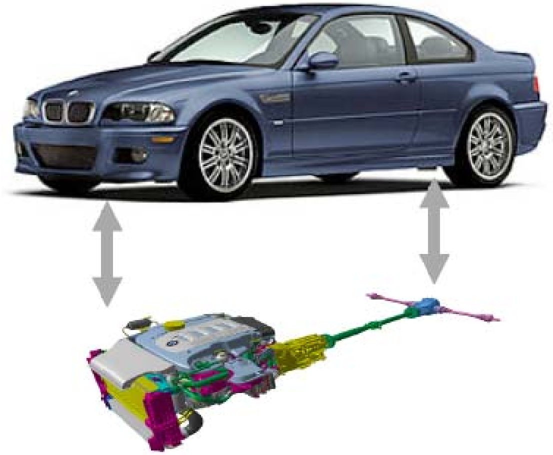
Applications

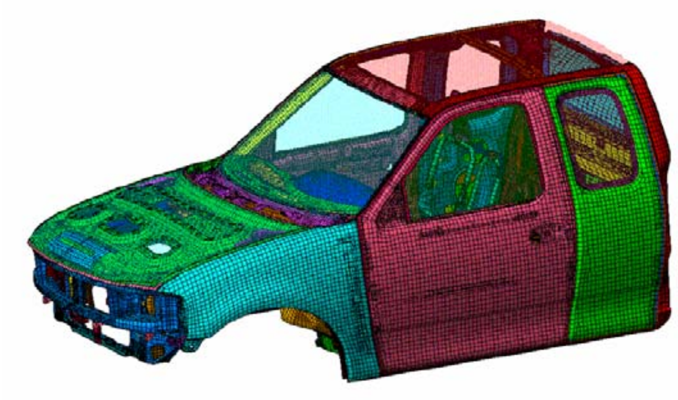


Shock response & damage prediction

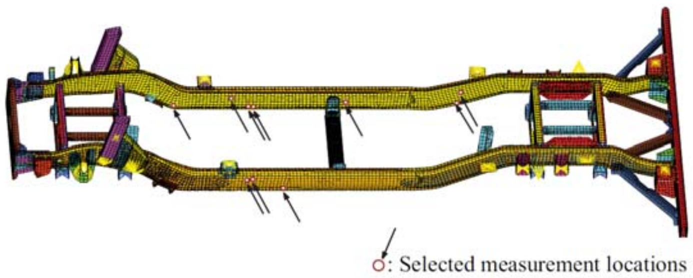


Multi-scale model analysis

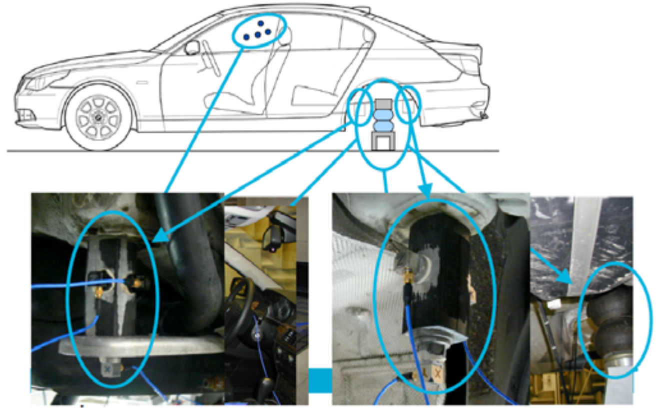




Structural dynamic analysis
and “Eigenvalue solver”
(Especially AMLS method)

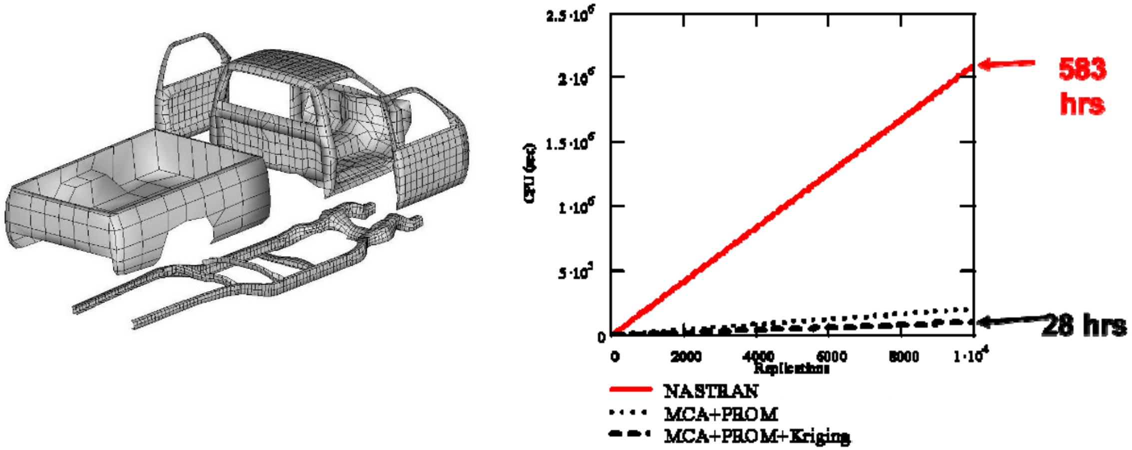


Health monitoring
and sensor positioning

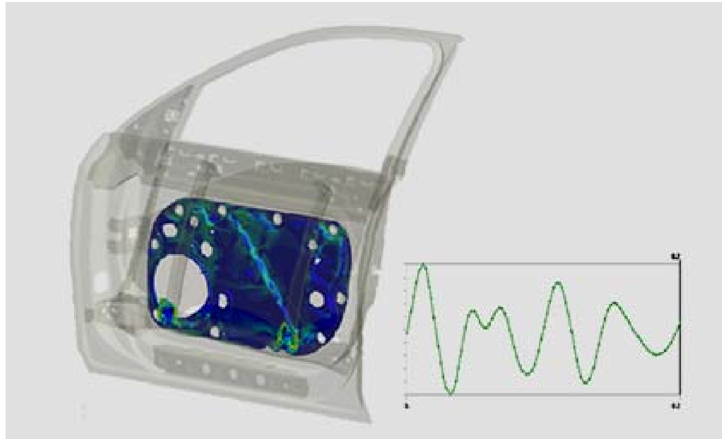


Experimental dynamic analysis

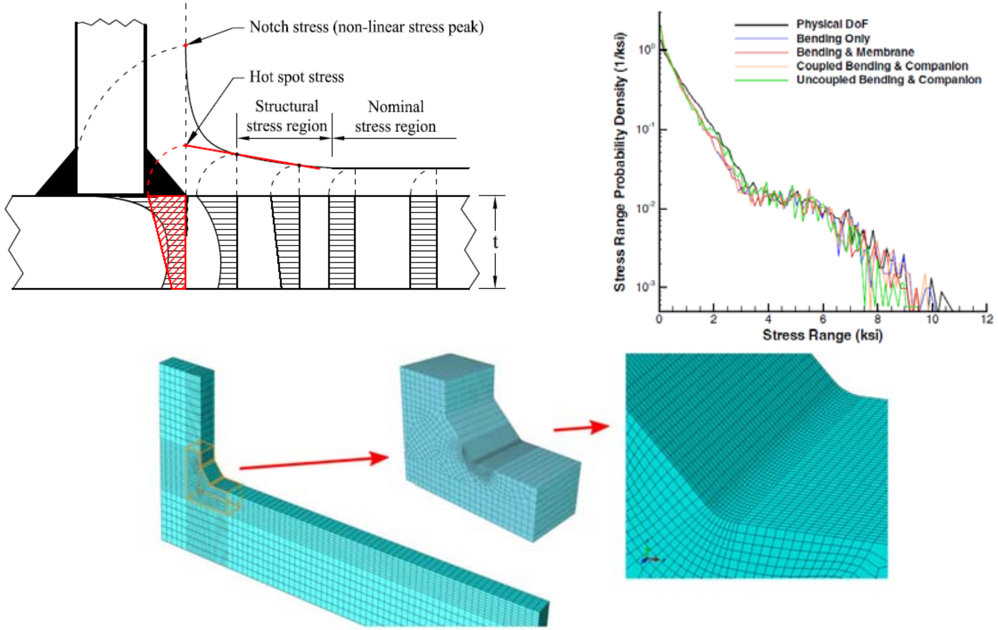
Applications



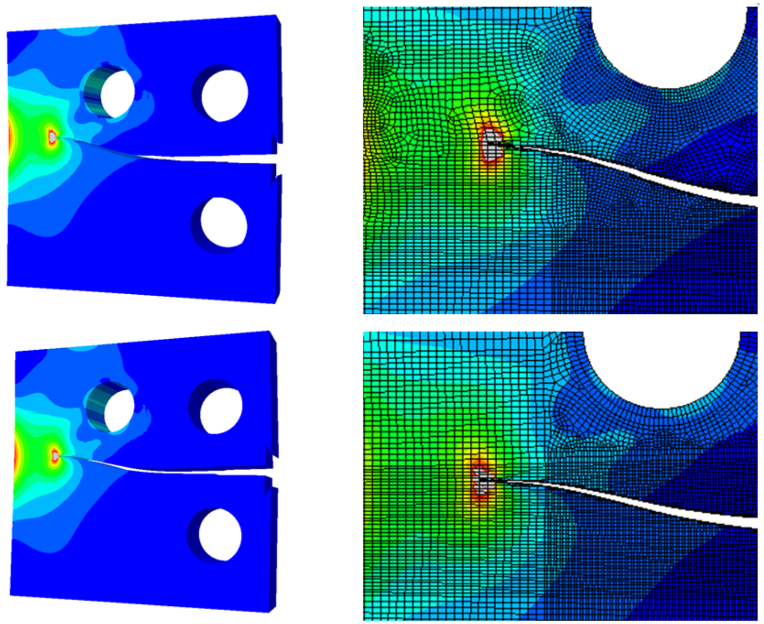
Design optimization through re-analysis



Transient analysis
(Alternative for implicit / explicit solvers)



Spectral fatigue analysis



Crack propagation analysis

2. History & Issues

History of the model reduction method

1965

- Hurty

: A pioneer of CMS method

- Use '*component modes*' and '*synthesis*'.
- Fixed interface CMS
- Rigid body +Constraint

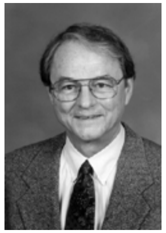
+ Fixed interface normal modes

1968

- Craig-Bampton (CB) method
- Modified Hurt's approach.

:Constraint + Fixed interface normal modes

- Simple & Accurate
- Most popular



Craig

1969

- Goldman

: Free interface normal modes

1971

- MacNeal

: Hybrid method

- *Fixed-Free* interface normal modes

MacNeal

1975

- Rubin

: Extended MacNeal's approach (Approximating *Free* interface normal modes)

- Considering the residual effects

1976

- Craig-Chang

: Modified Rubin's idea (*Free* interface normal modes)

- Considering residual flexibility (Accuracy ↑)

1984

- Ookuma & Nagamatsu

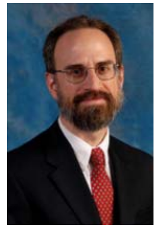
: Developed *multiple* CMS

- Multiple partitioning (Efficiency ↑)

1987

- J.K. Bennighof
- '*Component mode iteration*' method

:Using subspace iteration method at the substructure level



Bennighof

1988

- Craig & Hale

: Used Block Krylov Ritz vectors instead of component normal modes

- Less computational effort

1960s 1970s 1980s

[1960s ~ 1980s]

✓ Various interface handling techniques. (Fixed → Free → Fixed + Free)

✓ To calculate more accurate substructural normal modes. (SIM, Block Krylov)

History of the model reduction method

1990
- El-Sayed & Hsiung

: Employing parallel processing
(Efficiency ↑)

1992
- Singh & Suarez

: A new ***Free* interface** CMS
- Using higher order modal combination technique
(Accuracy ↑)

1996
- Parametric model reduction
- Model updating


E. Balmes

1992
- CMS + Guyan reduction
- Substructuring for solution efficiency


Bouhaddi

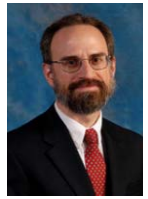
1997
- Greg Hulbert
- Using the quasi-static mode instead of constraint mode


G. Hulbert

2002
- Dual CB method
- Assembly using interface force
- Experimental substructuring


D. Rixen

2004
- J.K. Bennighof
- AMLS method
- Automated multi-level substructuring
- Computer-aid formulation: Extremely fast, eigensolver


Bennighof

2005
- C. Yang & NG Esmond
- ‘*Priori*’ error bound for AMLS method (Only for single level)

2006
- H. Voss : Modified Yang’s approach
- ‘*Priori*’ error bound for CB/AMLS methods
- Check the reliability of the reduced model


H. Voss

1990s

2000s

[1990s ~ 2000s]

✓ First try for ‘Parallel’ computing.

✓ New interface handling techniques.

✓ More accurate reduced model.

[2010s ~]

✓ Expansion to Experiment.

✓ New paradigm: Eigensolver, Interest for ‘Efficiency’

✓ Studies on “Error estimator”: Interest for ‘Reliability’

Primal issues for half a century

❖ **We can identify that, there have been "<u>Three Primal Issues</u>" in the research field.**

Issue 1. Solution "<u>Reliability</u>"

❖ Error bound (= Error estimation): 1. For **<u>CB</u>** method 2. For **<u>AMLS</u>** method

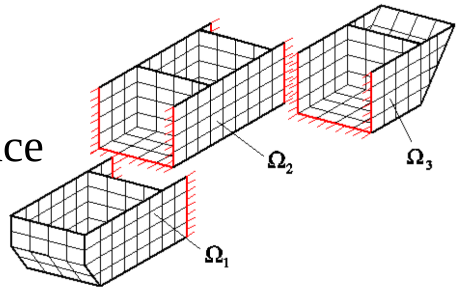
➡ **Topic 1 : Error estimators for the CB/AMLS methods**

Issue 2. Solution "<u>Accuracy</u>"

❖ Interface handling techniques

1. **<u>Fixed</u>** interface 2. **<u>Free</u>** interface

3. **<u>Hybrid</u>** interface



❖ Substructural modes calculation

1. **<u>SIM</u>** 2. **<u>Block Krylov Ritz vectors</u>**

3. **<u>Quasi-static mode</u>**

➡ **Topic 2 : Enhanced AMLS method**

Issue 3. Solution "<u>Efficiency</u>"

❖ Improving solution efficiency

1. **<u>Parallel</u>** techniques 2. **<u>Automated multi-level</u>** substructuring

3. Using **<u>sparsity pattern</u>**

➡ **Topic 3 : Algebraic dynamic condensation method**

3. Research topics

Topic 1-1. Simplified error estimator for the CB method

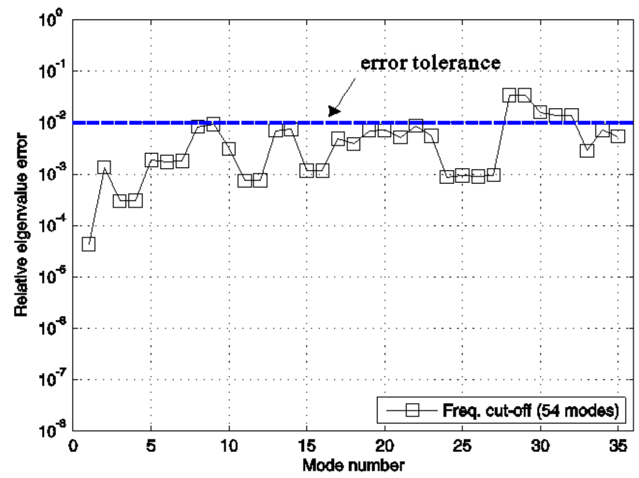
Motivation

❖ **To verify the <u>solution reliability</u> of the reduced model,**

Generally, we use "**Relative eigenvalue errors**"

$$\xi_i = \frac{\bar{\lambda}_i - \lambda_i}{\lambda_i} = \frac{\bar{\lambda}_i}{\lambda_i} - 1$$

However, to obtain the exact eigenvalue λ_i, we should solve the **global eigenvalue problem** $\mathbf{K}_g(\boldsymbol{\varphi}_g)_i = \lambda_i \mathbf{M}_g(\boldsymbol{\varphi}_g)_i$.



< Relative eigenvalue errors >

Since 2000s…

<u>Finite element model</u>

Complexity ↑

Model size ($\mathbf{M}_g$, $\mathbf{K}_g$) ↑

$\mathbf{K}_g(\boldsymbol{\varphi}_g)_i = \lambda_i \mathbf{M}_g(\boldsymbol{\varphi}_g)_i$ **: Requires large computational cost.**

✓ **What about 1,000,000 DOFs or more DOFs problem ?**

→ It is not easy to get λ_i .

❖ **For the reduced model of the large FE model: Interest for "<u>Solution Reliability</u> ↑"**

Previous error estimation method

❖ **Several meaningful error estimators were developed.**

1. Yang *et al*. (2005), "*priori*" error bound for AMLS method

$$\rho(\omega) = \frac{\lambda_1}{\omega - \lambda_1} < \tau$$: Only for single level substructuring

: ω is the approximated eigenvalue, λ_1 is the smallest eigenvalue under consideration

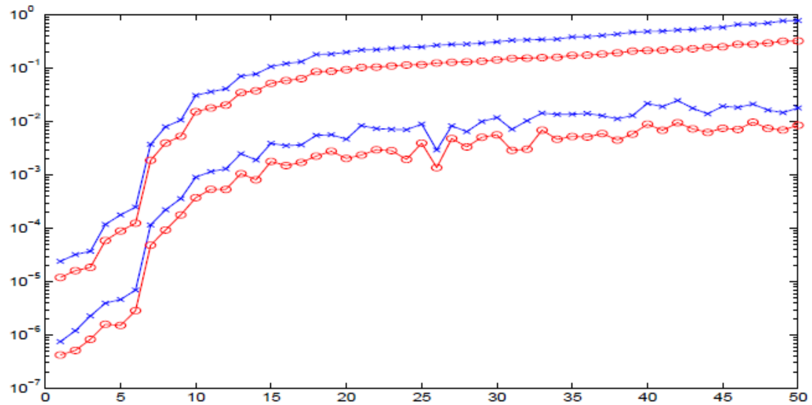
τ is the given tolerance

2. Elssel and Voss (2006), "*priori*" error bound for CB and AMLS methods

CB: $\hat{\mu}_i = \dfrac{\bar{\lambda}_i}{\left|\lambda_r - \bar{\lambda}_i\right|}$, **AMLS:** $\hat{\mu}_i = \prod_{k=0}^{p}\left(1 + \dfrac{\bar{\lambda}_i}{\lambda_k - \bar{\lambda}_i}\right) - 1$

: λ_r is the smallest residual eigenvalue of substructures.

✓ **Scalar operation: Very fast. But, poor accuracy.**



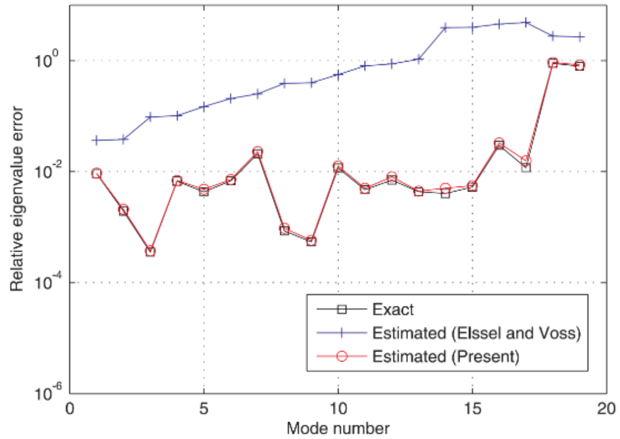
Previous error estimation method

3. Kim and Lee (2015), "*posteriori*" error estimator for the CB method

✓ Derive the "**enhanced transformation matrix**" using residual flexibility matrix $\mathbf{F}_{rs}$.

$$\overline{\mathbf{T}}_1 \quad \text{with} \quad \overline{\mathbf{T}}_1 = \overline{\mathbf{T}}_0 + \overline{\mathbf{T}}_a$$

$$\mu_i = 2(\overline{\boldsymbol{\varphi}}_p)_i^T \overline{\mathbf{T}}_0^T \left[\mathbf{M}_g - \frac{1}{\overline{\lambda}_i} \mathbf{K}_g \right] \overline{\mathbf{T}}_a (\overline{\boldsymbol{\varphi}}_p)_i + (\overline{\boldsymbol{\varphi}}_p)_i^T \overline{\mathbf{T}}_a^T \left[\mathbf{M}_g - \frac{1}{\overline{\lambda}_i} \mathbf{K}_g \right] \overline{\mathbf{T}}_a (\overline{\boldsymbol{\varphi}}_p)_i$$



✓ **Very accurate.**
✓ **Global matrix operations, complicated.**

❖ What is the drawback / limitation?

✓ Yang, Elssel and Voss's : Accuracy ↓ .

✓ Kim and Lee's : Accurate, but efficiency ↓ .

Global matrix computing, Requiring large memory

→ Cannot apply to large FE models (over 100,000 DOFs).

Design focus

1. **Requirements** for a new error estimator ?

- ✓ **Accuracy & Efficiency**. Adoptable for real engineering problems.

2. **Key ideas** ?

- ✓ **Way to see in a part**: **Submatrix level** formulation and computing.
- ✓ Using matrix property: **Sparsity** (**Compact computing**),
  Symmetry (**Half calculation**), **Orthogonality** (**Give zero matrix**).

3. What is the "**Strengths**", which is expected ?

- ✓ Formulations would be expressed in submatrix form: **Computationally efficient**.
- ✓ Then, the **substructural error contributions** for global mode can be calculated.

Derivation procedure

1. Original formulation of error estimator

$$\eta_i = 2(\overline{\boldsymbol{\varphi}}_p)_i^T \overline{\mathbf{T}}_0^{\mathrm{T}} \left[\mathbf{M}_g - \frac{1}{\overline{\lambda}_i} \mathbf{K}_g \right] \overline{\mathbf{T}}_r (\overline{\boldsymbol{\varphi}}_p)_i + (\overline{\boldsymbol{\varphi}}_p)_i^T \overline{\mathbf{T}}_r^T \left[\mathbf{M}_g - \frac{1}{\overline{\lambda}_i} \mathbf{K}_g \right] \overline{\mathbf{T}}_r (\overline{\boldsymbol{\varphi}}_p)_i$$

1 2 3 4

$$\Rightarrow \quad \eta_i = \overline{\boldsymbol{\varphi}}_i^T [\, 2\,\overline{\lambda}_i \overline{\mathbf{T}}_0^T \mathbf{M}_g \mathbf{T}_a - 2\,\overline{\mathbf{T}}_0^T \mathbf{K}_g \mathbf{T}_a + \overline{\lambda}_i^2 \mathbf{T}_a^T \mathbf{M}_g \mathbf{T}_a - \overline{\lambda}_i \mathbf{T}_a^T \mathbf{K}_g \mathbf{T}_a \,] \overline{\boldsymbol{\varphi}}_i$$

✓ **Submatrix form :** $\mathbf{M}_g = \begin{bmatrix} \mathbf{M}_s & \mathbf{M}_c \\ \mathbf{M}_c^{\mathrm{T}} & \mathbf{M}_b \end{bmatrix}$, $\mathbf{K}_g = \begin{bmatrix} \mathbf{K}_s & \mathbf{K}_c \\ \mathbf{K}_c^{\mathrm{T}} & \mathbf{K}_b \end{bmatrix}$, $\overline{\mathbf{T}}_0 = \begin{bmatrix} \mathbf{\Phi}_s^d & \mathbf{\Psi}_c \\ \mathbf{0} & \mathbf{I}_b \end{bmatrix}$, $\mathbf{T}_a = \begin{bmatrix} \mathbf{0} & \mathbf{F}_{rs}\hat{\mathbf{M}}_c \\ \mathbf{0} & \mathbf{0} \end{bmatrix}$

1 $\overline{\mathbf{T}}_0^T \mathbf{M}_g \mathbf{T}_a = \begin{bmatrix} \mathbf{0} & \mathbf{A}_c \\ \mathbf{0} & \mathbf{A}_b \end{bmatrix}$ with $\mathbf{A}_c = \underline{(\mathbf{\Phi}_s^d)^T \mathbf{M}_s \mathbf{F}_{rs}} \hat{\mathbf{M}}_c$, $\mathbf{A}_b = \hat{\mathbf{M}}_c^T \mathbf{F}_{rs} \hat{\mathbf{M}}_c$

"**Orthogonal**" property
$(\mathbf{\Phi}_s^d)^T \mathbf{M}_s \mathbf{F}_{rs} = \mathbf{0}$

$$\Rightarrow \quad \therefore \overline{\mathbf{T}}_0^T \mathbf{M}_g \mathbf{T}_a = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_b \end{bmatrix}$$

✓ **Zero matrices: Give cost saving !**

2 $\overline{\mathbf{T}}_0^T \mathbf{K}_g \mathbf{T}_a = \begin{bmatrix} \mathbf{0} & \mathbf{B}_c \\ \mathbf{0} & \mathbf{B}_b \end{bmatrix}$ with $\mathbf{B}_c = \underline{(\mathbf{\Phi}_s^d)^T \mathbf{K}_s \mathbf{F}_{rs}} \hat{\mathbf{M}}_c$, $\mathbf{B}_b = \underline{(\mathbf{\Psi}_c^T \mathbf{K}_s + \mathbf{K}_c^T)} \mathbf{F}_{rs} \hat{\mathbf{M}}_c$

"**Orthogonal**" property
$(\mathbf{\Phi}_s^d)^T \mathbf{K}_s \mathbf{F}_{rs} = \mathbf{0}$

$$\Rightarrow \quad \therefore \overline{\mathbf{T}}_0^T \mathbf{K}_g \mathbf{T}_a = \mathbf{0}$$

$\mathbf{\Psi}_c = -\mathbf{K}_s^{-1} \mathbf{K}_c$, $\mathbf{\Psi}_c^T \mathbf{K}_s + \mathbf{K}_c^T = \mathbf{0}$

Derivation procedure

1 = 4 $\quad \therefore \overline{\mathbf{T}}_0^T \mathbf{M}_g \mathbf{T}_a = \mathbf{T}_a^T \mathbf{K}_g \mathbf{T}_a$

2. The following equations are obtained

$$\eta_i = \overline{\lambda}_i \overline{\boldsymbol{\varphi}}_i^T \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_b \end{bmatrix} \overline{\boldsymbol{\varphi}}_i + \underline{\overline{\lambda}_i^2 \overline{\boldsymbol{\varphi}}_i^T \mathbf{T}_a^T \mathbf{M}_g \mathbf{T}_a \overline{\boldsymbol{\varphi}}_i}$$

1 3

$$\boxed{\overline{\boldsymbol{\varphi}}_i = \begin{bmatrix} \overline{\boldsymbol{\varphi}}_s \\ \overline{\boldsymbol{\varphi}}_b \end{bmatrix}_i}$$

Higher order term neglected

3. The error estimator is approximated

$$\eta_i \approx \mu_i = \overline{\lambda}_i (\overline{\boldsymbol{\varphi}}_b)_i^T \mathbf{A}_b (\overline{\boldsymbol{\varphi}}_b)_i \quad \text{with} \quad \mathbf{A}_b = \hat{\mathbf{M}}_c^T \mathbf{F}_{rs} \hat{\mathbf{M}}_c.$$

Submatrix form

$$\hat{\mathbf{M}}_c = \begin{bmatrix} \hat{\mathbf{M}}_c^{(1)} \\ \hat{\mathbf{M}}_c^{(2)} \\ \vdots \\ \hat{\mathbf{M}}_c^{(k)} \end{bmatrix}, \quad \mathbf{F}_{rs} = \begin{bmatrix} \mathbf{F}_{rs}^{(1)} & & & \mathbf{0} \\ & \mathbf{F}_{rs}^{(2)} & & \\ & & \ddots & \\ \mathbf{0} & & & \mathbf{F}_{rs}^{(k)} \end{bmatrix}$$

4. A new error estimator

$$\mu_i = \sum_{k=1}^{n} \overline{\lambda}_i (\overline{\boldsymbol{\varphi}}_b)_i^T \hat{\mathbf{M}}_c^{(k)T} \mathbf{F}_{rs}^{(k)} \hat{\mathbf{M}}_c^{(k)} (\overline{\boldsymbol{\varphi}}_b)_i$$

$\mathbf{F}_{rs}^{(k)}$: Symmetric, fully populated matrix.

$$\mathbf{F}_{rs}^{(k)} = \mathbf{F}_d^{(k)} + \mathbf{F}_u^{(k)} + \mathbf{F}_u^{(k)T}$$

Derivation procedure

5. Finally, the simplified error estimator is defined as

$$\boxed{\mu_i = \sum_{k=1}^{n} \mu_i^{(k)}} \quad \text{with} \quad \mu_i^{(k)} = e_1^{(k)} + 2e_2^{(k)},\ e_1^{(k)} = \bar{\lambda}_i (\bar{\boldsymbol{\varphi}}_b)_i^T \hat{\mathbf{M}}_c^{(k)T} \mathbf{F}_d^{(k)} \hat{\mathbf{M}}_c^{(k)} (\bar{\boldsymbol{\varphi}}_b)_i,\ e_2^{(k)} = \bar{\lambda}_i (\bar{\boldsymbol{\varphi}}_b)_i^T \hat{\mathbf{M}}_c^{(k)T} \mathbf{F}_u^{(k)} \hat{\mathbf{M}}_c^{(k)} (\bar{\boldsymbol{\varphi}}_b)_i$$

✓ **Represented by <u>a simple summation of the substructural errors estimated</u>.**

➡ **<u>Attractive feature</u> of the proposed method.**

Previous vs New error estimator: Operation counts

❖ Previous error estimator

$$\eta_i = 2(\overline{\boldsymbol{\varphi}}_p)_i^T \overline{\mathbf{T}}_0^{\mathrm{T}} \left[\mathbf{M}_g - \frac{1}{\overline{\lambda}_i} \mathbf{K}_g \right] \overline{\mathbf{T}}_r (\overline{\boldsymbol{\varphi}}_p)_i + (\overline{\boldsymbol{\varphi}}_p)_i^T \overline{\mathbf{T}}_r^T \left[\mathbf{M}_g - \frac{1}{\overline{\lambda}_i} \mathbf{K}_g \right] \overline{\mathbf{T}}_r (\overline{\boldsymbol{\varphi}}_p)_i \quad \text{with} \quad \overline{\mathbf{T}}_r = \begin{bmatrix} \mathbf{0} & \mathbf{F}_{rs} \hat{\mathbf{M}}_c \\ \mathbf{0} & \mathbf{0} \end{bmatrix}$$

Step	Items	Counts
1	$\mathbf{F}_{rs} = (\mathbf{K}_s)^{-1} - (\boldsymbol{\Phi}_d)(\boldsymbol{\Lambda}_d)^{-1}(\boldsymbol{\Phi}_d)^T$	ns^2
2	$\overline{\mathbf{T}}_r$	$s(2s-1)bn$
3	$\mathbf{M}_g - \frac{1}{\overline{\lambda}_i} \mathbf{K}_g$	$2N^2$
4	η_i	$(2N-1)(N+d)+(2d-1)(N+1)+1$
Total counts	$ns^2 + s(2s-1)bn + \underline{2N^2 + (2N-1)(N+d) + (2d-1)(N+1) + 1}$	

Polynomials of total DOFs, N

Number of total DOFs : N

Number of substructures : n

Number of each substructural DOFs : s

Size of the interface : b

Size of the reduced model : d

Previous vs New error estimator: Operation counts

❖ New error estimator

$$\mu_i = \sum_{k=1}^{n} \mu_i^{(k)} \quad \text{with} \quad \mu_i^{(k)} = e_1^{(k)} + 2e_2^{(k)},\ e_1^{(k)} = \bar{\lambda}_i (\bar{\boldsymbol{\varphi}}_b)_i^T \hat{\mathbf{M}}_c^{(k)T} \mathbf{F}_d^{(k)} \hat{\mathbf{M}}_c^{(k)} (\bar{\boldsymbol{\varphi}}_b)_i,\ e_2^{(k)} = \bar{\lambda}_i (\bar{\boldsymbol{\varphi}}_b)_i^T \hat{\mathbf{M}}_c^{(k)T} \mathbf{F}_u^{(k)} \hat{\mathbf{M}}_c^{(k)} (\bar{\boldsymbol{\varphi}}_b)_i$$

Step	Items	Counts
1	$\mathbf{F}_{rs}^{(k)} = (\mathbf{K}^{(k)})^{-1} - (\boldsymbol{\Phi}_d^{(k)})(\boldsymbol{\Lambda}_d^{(k)})^{-1}(\boldsymbol{\Phi}_d^{(k)})^T$	ns^2
2	$e_1^{(k)}$ and $e_2^{(k)}$	$2n[(s+1)(2b-1)+(b+s)(2s-1)+1]$
3	$\mu_i^{(k)}$	$2n-1$
Total counts	$ns^2 + 2n[(s+1)(2b-1)+(b+s)(2s-1)+1] + 2n-1$ **: Polynomials of “s”**	

Number of total DOFs : N

Number of substructures : n

Number of each substructural DOFs : s

Size of the interface : b

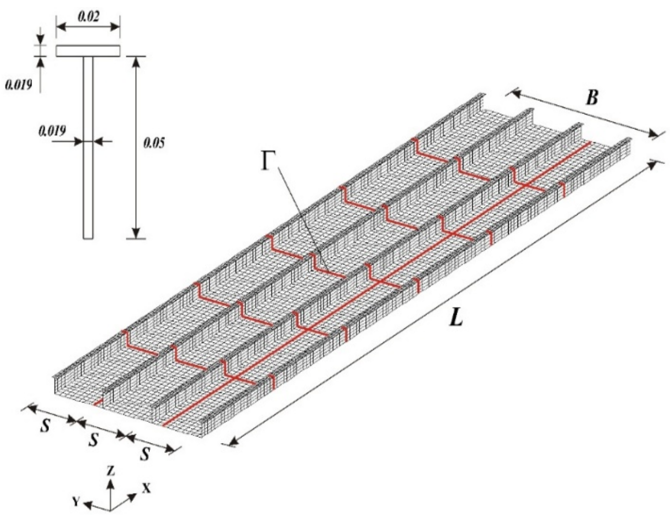
Size of the reduced model : d

Previous vs New error estimator: Operation counts

❖ Operation counts

EEM	Operation counts
Previous	$ns^2 + s(2s-1)bn + 2N^2 + (2N-1)(N+d) + (2d-1)(N+1) + 1$
New	$ns^2 + 2n[(s+1)(2b-1) + (b+s)(2s-1) + 1] + 2n - 1$

✓ For stiffened plate problem



Number of total DOFs, N=52662

Number of substructures : n=18

Number of each substructural DOFs : s=2730

Size of the interface : b=3522

Size of the reduced model : d=3582

Operation counts :

Previous vs New

$9.63\times10^{11} > 1.09\times10^{9}$

➢ **New error estimator only requires 0.22 % operation counts of that of previous error estimator.**

Numerical examples

Computer spec. : Intel core (TM) i7-3770, 3.40 GHz CPU, 32GB RAM.

FE Model: 4-node MITC shell element.

Material: Mild steel. (E=206 GPa, v = 0.3, density = 7,850 kg/m^3)

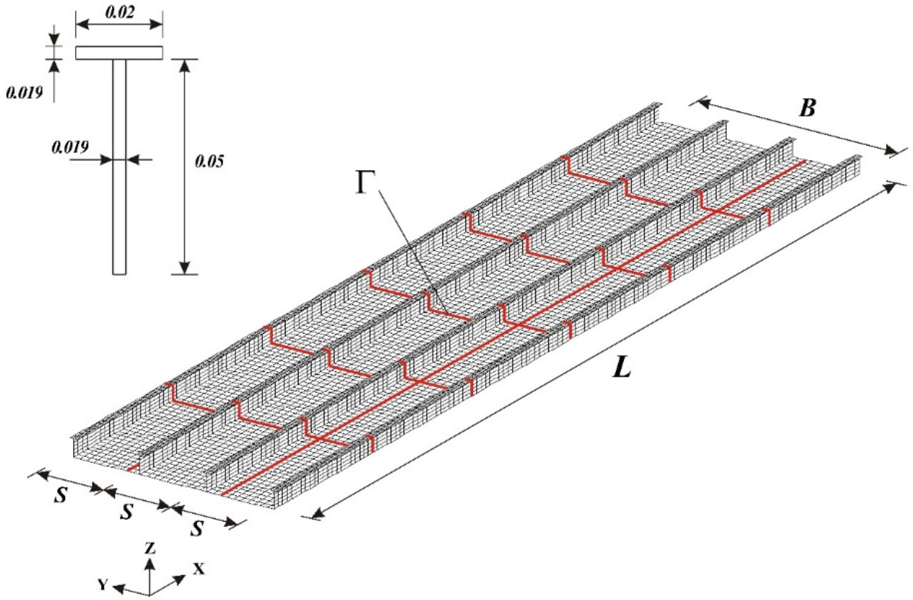
Implementation: Code implementations are done using ***MATLAB***.

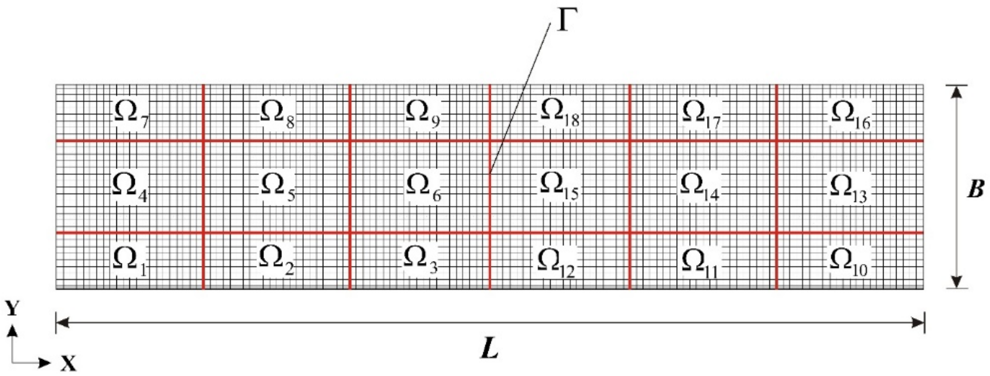
- ✓ Compared with the exact relative eigenvalue errors.
- ✓ Also, compared with the previous error estimation methods

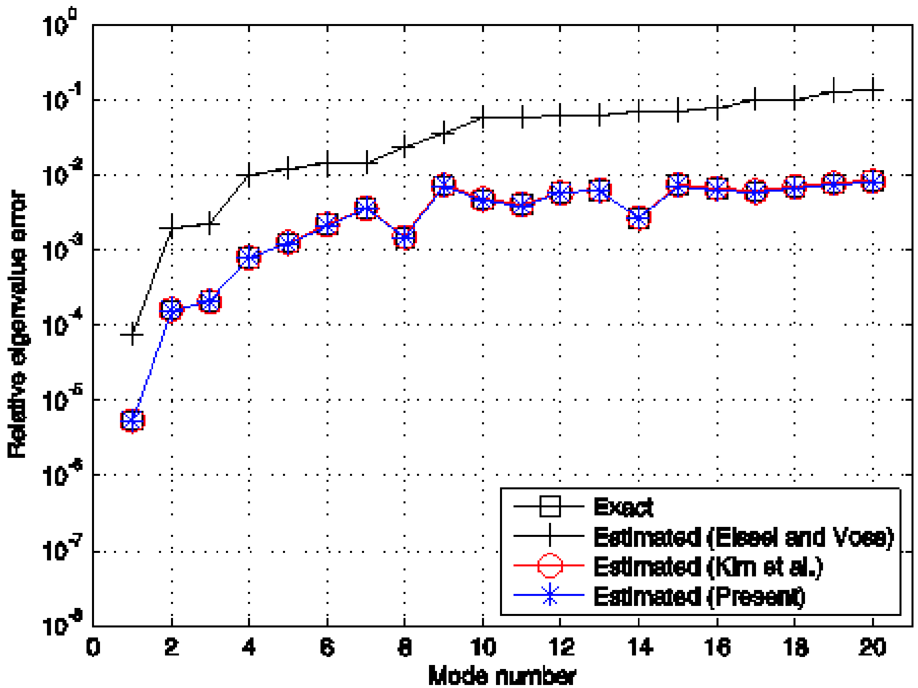
❖ Stiffened Plate structure

Dimension (L=26 m, B=6 m, S=2 m), **18 substructures**. No B.C.

Total DOFs = **52,662** → **Retain modes = 200**. **Reduced system size = 3,582 (6.8%)**.







< Relative eigenvalue errors >

❖ Stiffened Plate structure

➢ Breakdowns of the computational cost

Items		Computation times [sec]	Ratio [%]
CB method		165.24	100.00
Previous error estimator in Eq. (29)	Calculation of the residual flexibility matrix $\mathbf{F}_{rs}$	1.89	1.14
	Calculation of $\mathbf{T}_a$ matrix	41.12	24.89
	Calculation error estimator η_i	16.23	9.82
	Total	59.24	35.85
Present error estimator in Eq. (40)	Calculation of the residual flexibility matrix $\mathbf{F}_{rs}$	1.89	1.14
	Calculation error estimator μ_i	1.87	1.13
	Total	3.76	2.27
Present error estimator considering symmetric partitioning	Calculation of the residual flexibility matrix $\mathbf{F}_{rs}$	0.79	0.48
	Calculation error estimator μ_i	0.40	0.24
	Total	1.19	0.72
Elssel and Voss's error estimator in Eq. (41)		0.000011	0.0000066

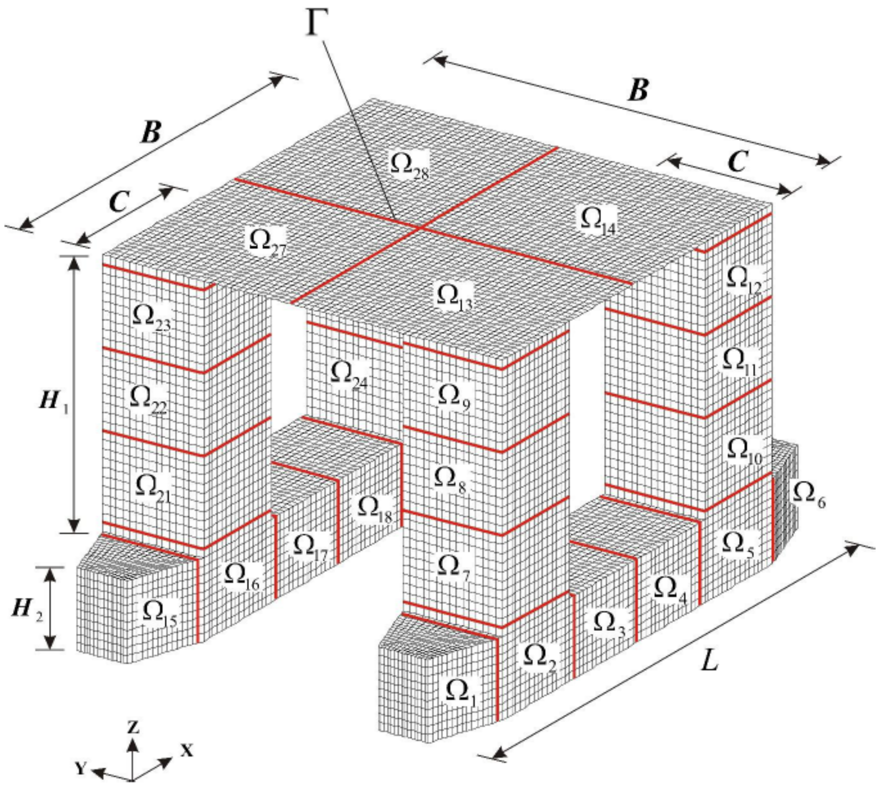
Only requires **2.27 %** of additional computation time.

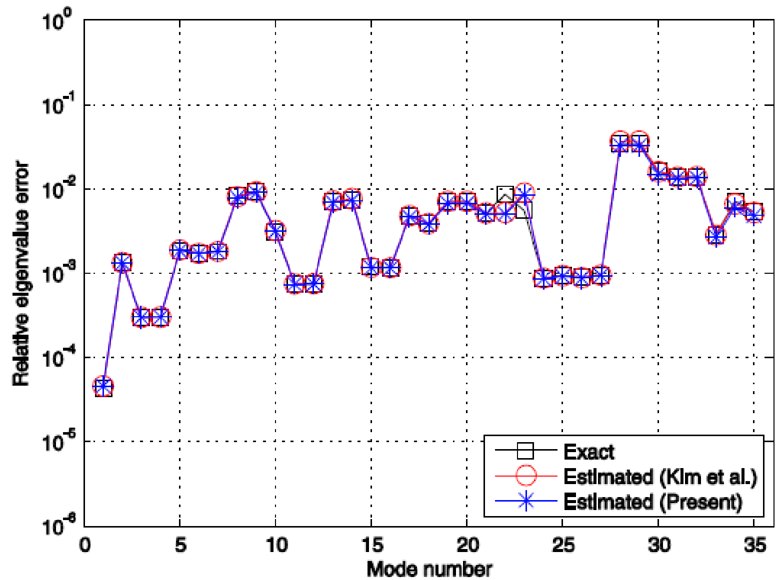
15 times faster

❖ Semi-submerged rig structure

Dimension (L=110 m, B=80 m, C=20 m, H_1=50, H_2=15), **28 substructures**. No B.C.

Total DOFs = **102,504 → Retain modes = 160**. **Reduced system size = 8,806 (8.6 %)**.





Items	Computation times [sec]	Computation times Ratio [%]
CB method	575.77	100.00
Previous error estimator in Eq. (29)	176.69	30.68
Present error estimator in Eq. (40)	11.95	2.07

Original error estimator : Cost is not small.

New error estimator : <u>15 times faster</u>

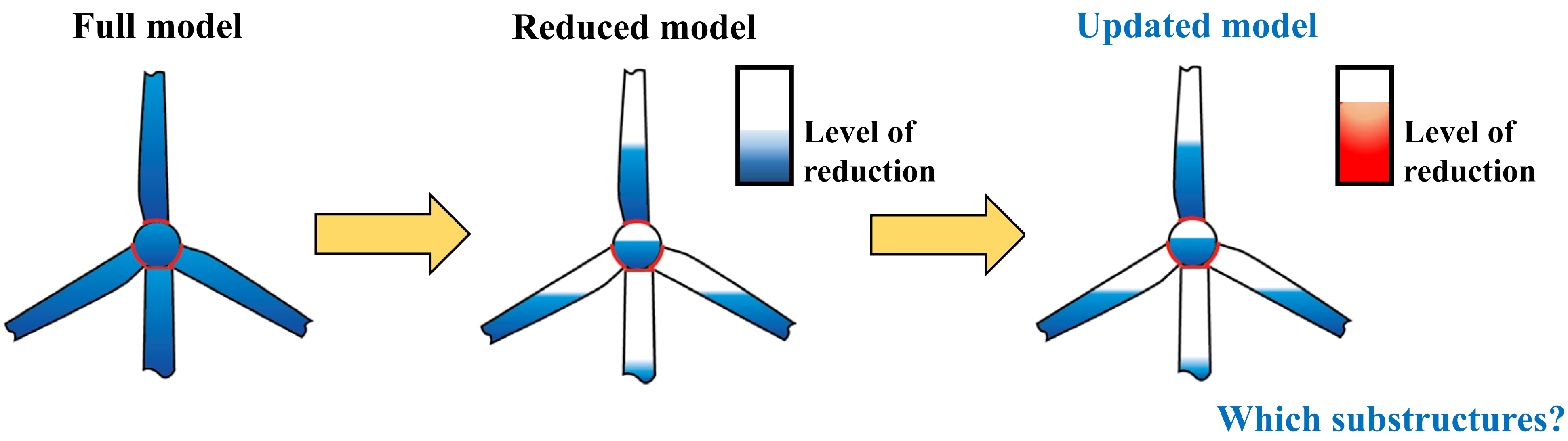
Error control strategy

❖ "**Attractive form**" **of the new error estimator : Simple summations of** $\mu_i^{(k)}$, $\mu_i = \sum_{k=1}^{n} \mu_i^{(k)}$

$\mu_i^{(k)}$: Provides the contribution of the k^{th} substructure to the i^{th} relative eigenvalue error.

➡ **The detailed errors** estimated for a certain substructure.

We can suggest an error control strategy to improve the accuracy of the global modes having relatively large errors.

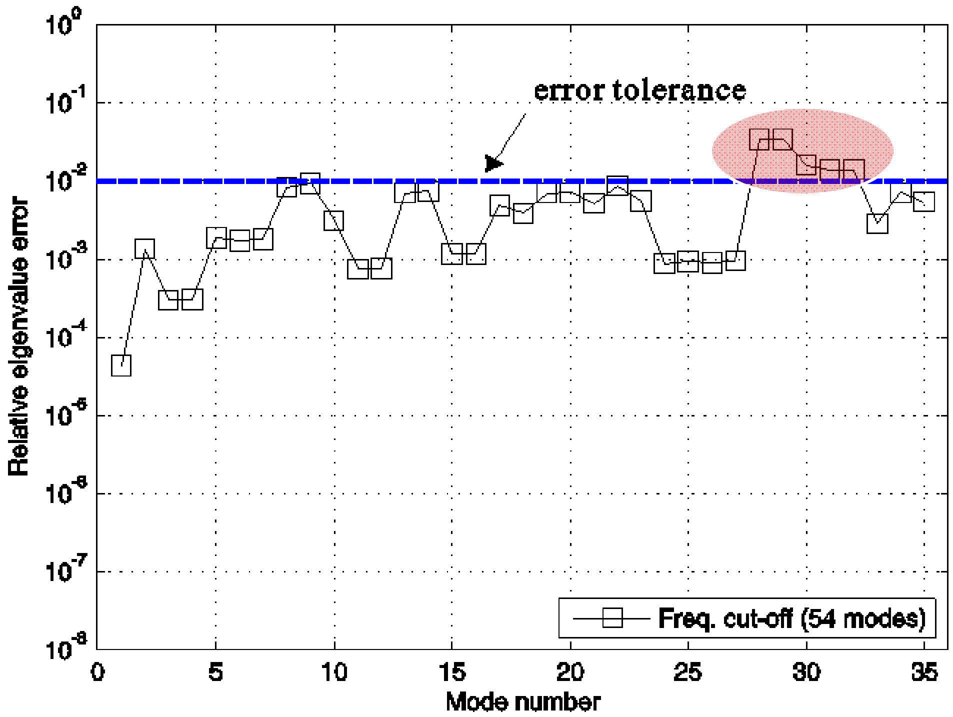


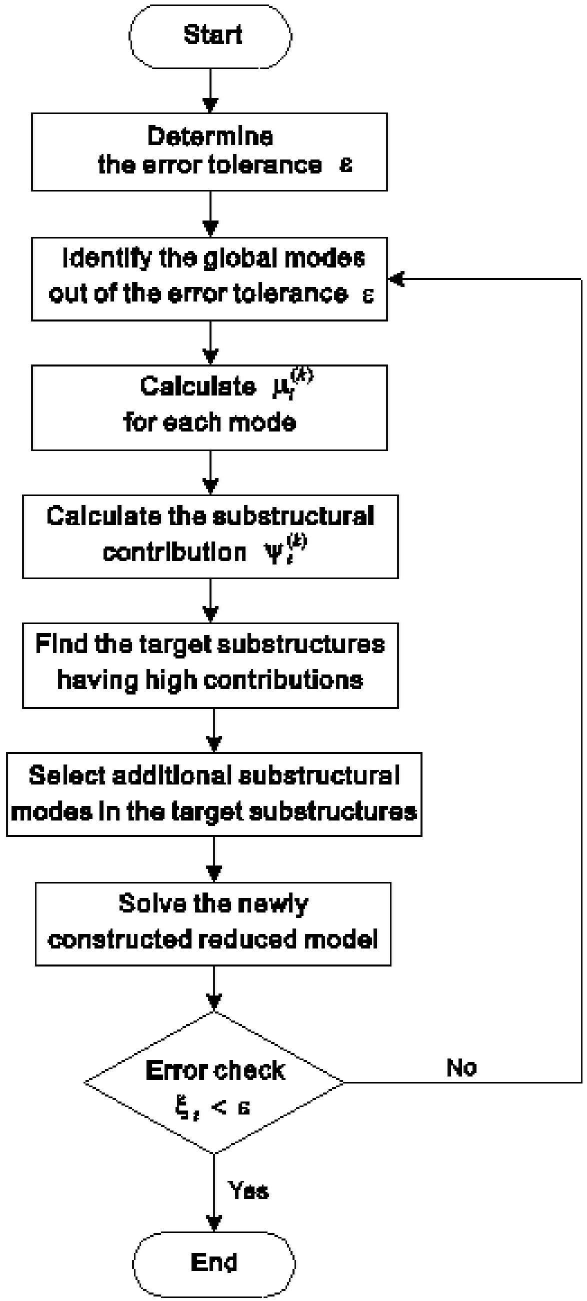
Error control strategy

❖ From $\mu_i^{(k)}$, we can define the "**substructural contribution**"

$$\psi_i^{(k)} = \frac{\mu_i^{(k)}}{\mu_i} \times 100\ [\%] \quad : \textbf{Substructural contribution}$$

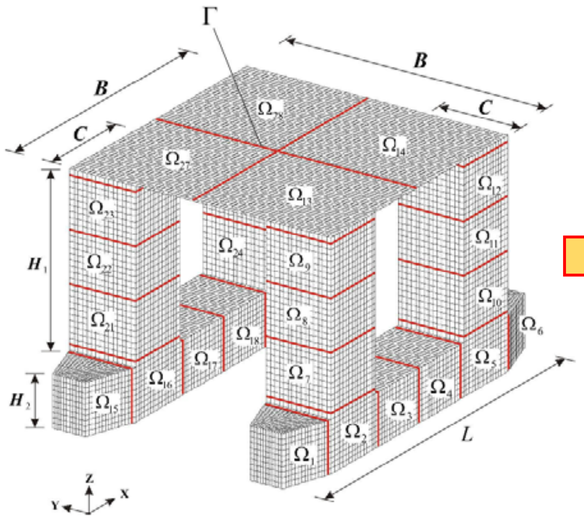
❖ Using $\psi_i^{(k)}$, we can **control the error** out of the tolerance !



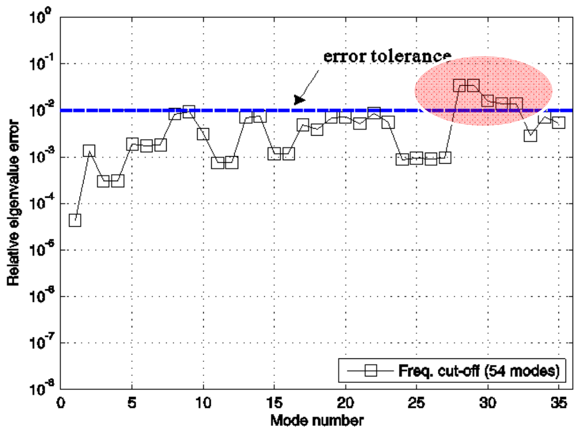


< Flow chart for error control >

Error control strategy



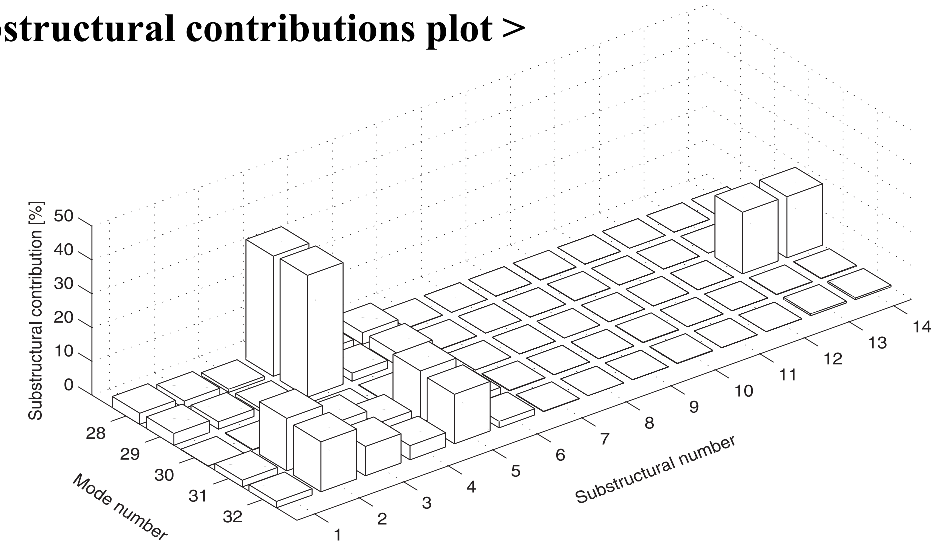
Define the error tolerance



- ✓ Out of the tolerance 10^{-2} : 28^{th} ~32^{th} modes
- ✓ Calculate the estimated substructural error $\mu_i^{(k)}$

Contributions analyze by $\psi_i^{(k)}$

< Substructural contributions plot >



Substructural numbers	Substructural contributions $\psi_i^{(k)}$ (%)				
	$\psi_{28}^{(k)}$	$\psi_{29}^{(k)}$	$\psi_{30}^{(k)}$	$\psi_{31}^{(k)}$	$\psi_{32}^{(k)}$
1, 15	3.41	3.57	0.04	2.05	1.93
2, 16	2.22	2.24	0.16	16.12	15.34
3, 17	0.92	0.93	0.10	9.45	9.00
4, 18	36.84	37.35	0.09	4.61	4.32
5, 19	2.40	2.42	0.16	15.96	15.18
6, 20	3.57	3.73	0.04	2.03	1.91
7, 21	0.10	0.08	0.02	0.16	0.15
8, 22	0.00	0.00	0.00	0.01	0.01
9, 23	0.00	0.00	0.01	0.01	0.01
10, 24	0.12	0.10	0.02	0.15	0.15
11, 25	0.00	0.00	0.00	0.01	0.01
12, 26	0.00	0.00	0.01	0.01	0.01
13, 27	0.00	0.00	19.42	0.14	0.50
14, 28	0.00	0.00	19.27	0.14	0.49
Total	100.00	100.00	100.00	100.00	100.00

- ✓ Define "Target substructures" having high contributions
- ✓ Target substructures : 2 (16), 4 (18), 5 (19), 13 (27), 14 (28)

Error control strategy

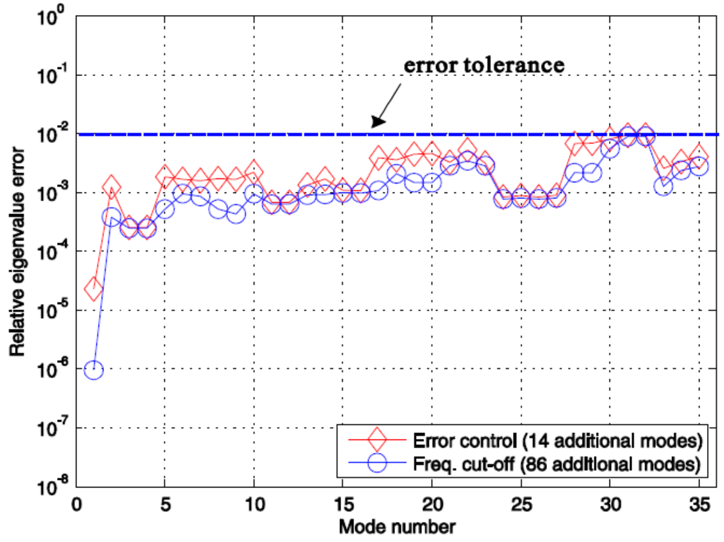
Add substructural modes

- ✓ **Target substructures 2, 5, 16, 19**

  : 2 additional modes

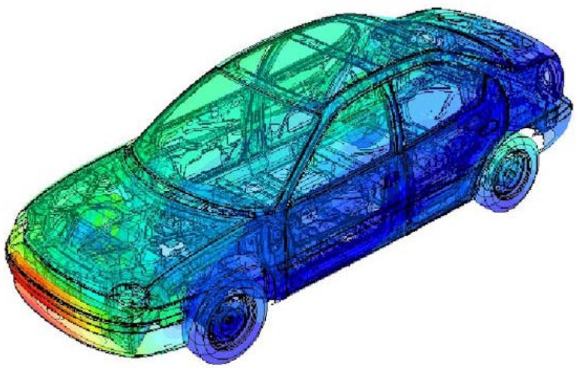
- ✓ **The other target substructures**

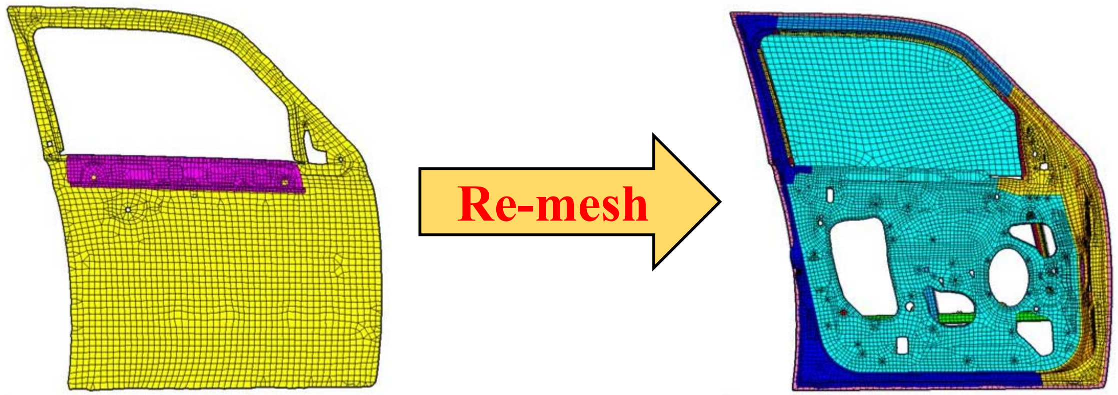
  : 1 additional mode



❖ Also, we can use this error control strategy for "**Model refinement**".

✓ Not adding substructural modes, but '**Re-meshing**' the target substructures crucial for the interested global mode.





Closure

1. We proposed a **simplified error estimator** for the CB method.

2. The estimated relative eigenvalue error is simple to calculate using **summation of the substructural errors estimated**.

3. We proposed **an error control strategy** to improve the accuracy of reduced models efficiently.

4. It would be valuable to develop the **iterative mode selection algorithms** to construct accurate reduced-order models.

Topic 1-2. Error estimator for the AMLS method

Brief introduction of the AMLS method

❖ **Component Mode Synthesis (CMS) methods**

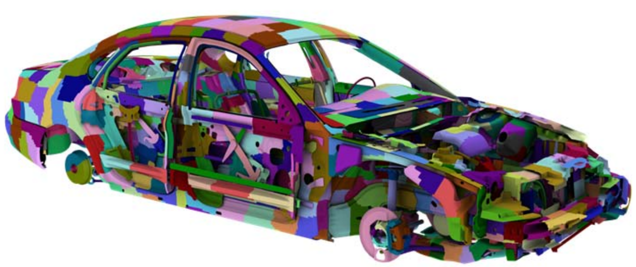
Well known methods. Used for many years in structural engineering.

But, it too suffers from **limitations due to FE model size**.

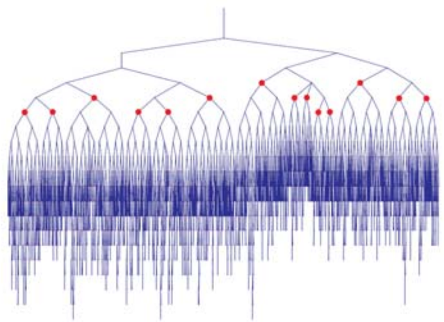
❖ **Automated Multi-Level Substructuring (AMLS) method** (J.K. Bennighof *et al* , *2004*)

✓ **Automatically** partitioning.

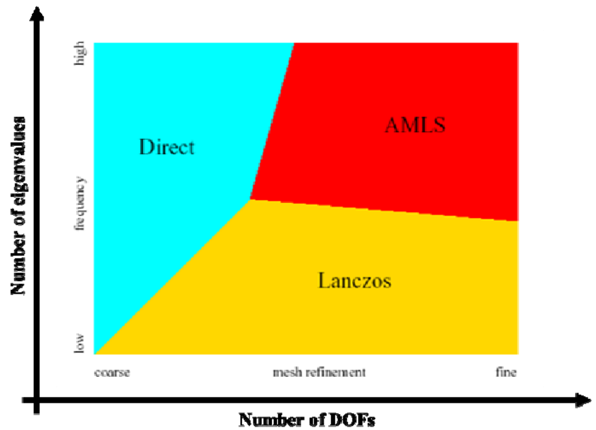
✓ **Recursively transformed** strategies, but much **more complex formulation**.



< Automated substructuring>



< Transformation tree >



✓ Successful in computing **thousands of eigenvalues in a few hours on PC**.

AMLS is a promising **alternative to "Lanczos"** for very large DOFs problem.

Enhanced transformation matrix

❖ To develop the error estimator of the AMLS method

→ Define the "enhanced transformation matrix" of the AMLS method.

1. Original AMLS transformation matrix

$$\mathbf{T}_0 = \mathbf{T}^{(1)}\mathbf{T}^{(2)}\cdots\mathbf{T}^{(N_s)} = \prod_{i=1}^{N_s}\mathbf{T}^{(i)} \Rightarrow \mathbf{T}_0 = \hat{\mathbf{\Psi}}\mathbf{\Phi}, \quad \mathbf{\Phi} = [\mathbf{\Phi}_d \quad \mathbf{\Phi}_r]$$

2. Therefore, the transformation matrix is also divided into two parts, dominant and residual parts.

$$\mathbf{T}_0 = [\mathbf{T}_0^d \quad \mathbf{T}_0^r] \quad \text{with} \quad \underline{\mathbf{T}_0^d = \hat{\mathbf{\Psi}}\mathbf{\Phi}_d}, \; \mathbf{T}_0^r = \hat{\mathbf{\Psi}}\mathbf{\Phi}_r$$

T matrix of AMLS method

3. The global displacement vector is divided into the dominant and residual parts.

$$\mathbf{u}_g = \mathbf{T}_0\boldsymbol{\eta}_p = [\mathbf{T}_0^d \quad \mathbf{T}_0^r]\begin{bmatrix}\boldsymbol{\eta}_p^d \\ \boldsymbol{\eta}_p^r\end{bmatrix} \Rightarrow \mathbf{M}_g\ddot{\mathbf{u}}_g + \mathbf{K}_g\mathbf{u}_g = \mathbf{0} \Rightarrow \left[\begin{array}{c|c}\mathbf{\Lambda}_d - \lambda\mathbf{M}_{dd} & -\lambda\mathbf{M}_{dr} \\ \hline -\lambda\mathbf{M}_{dr}^T & \mathbf{\Lambda}_r - \lambda\mathbf{M}_{rr}\end{array}\right]\begin{bmatrix}\boldsymbol{\eta}_p^d \\ \boldsymbol{\eta}_p^r\end{bmatrix} = \mathbf{0}$$

Exact transformed form.

Enhanced transformation matrix

4. Expanding the second row linear equation.

$$\begin{bmatrix} \mathbf{\Lambda}_d - \lambda \mathbf{M}_{dd} & -\lambda \mathbf{M}_{dr} \\ -\lambda \mathbf{M}_{dr}^T & \mathbf{\Lambda}_r - \lambda \mathbf{M}_{rr} \end{bmatrix} \begin{bmatrix} \boldsymbol{\eta}_p^d \\ \boldsymbol{\eta}_p^r \end{bmatrix} = \mathbf{0} \Rightarrow \boldsymbol{\eta}_p^r = \underline{\lambda(\mathbf{\Lambda}_r - \lambda \mathbf{M}_{rr})^{-1} \mathbf{M}_{dr}^T} \boldsymbol{\eta}_p^d$$

✓ The residual term can be substituted into **only dominant term**.

5. Therefore, $\mathbf{u}_g$ can be described **with only dominant term.**

$$\mathbf{u}_g = \mathbf{T}_0^d \boldsymbol{\eta}_p^d + \mathbf{T}_0^r \underline{\boldsymbol{\eta}_p^r} \Rightarrow \mathbf{u}_g = \mathbf{T}_0^d \underline{\boldsymbol{\eta}_p^d} + \lambda \hat{\mathbf{\Psi}} \hat{\mathbf{F}}_r \hat{\mathbf{\Psi}}^T \mathbf{M}_g \mathbf{T}_0^d \underline{\boldsymbol{\eta}_p^d}$$ **:Exact form**

with $\hat{\mathbf{F}}_r = \mathbf{\Phi}_r (\mathbf{\Lambda}_r - \lambda \mathbf{M}_{rr})^{-1} \mathbf{\Phi}_r^T$ **:Residual flexibility matrix**

✓ We cannot calculate this term in direct. **Need approximation**.

6. For approximation, using $\bar{\lambda}$ instead of λ in $\hat{\mathbf{F}}_r$, the following equation is obtained

$$\Rightarrow \bar{\mathbf{F}}_r = \mathbf{\Phi}_r (\mathbf{\Lambda}_r - \bar{\lambda} \mathbf{M}_{rr})^{-1} \mathbf{\Phi}_r^T$$

Enhanced transformation matrix

7. The residual flexibility matrix $\overline{\mathbf{F}}_r$ is expanded and approximated

$$\overline{\mathbf{F}}_r = \mathbf{\Phi}_r(\mathbf{\Lambda}_r - \overline{\lambda}\mathbf{M}_{rr})^{-1}\mathbf{\Phi}_r^T = \underline{\mathbf{\Phi}_r\mathbf{\Lambda}_r^{-1}\mathbf{\Phi}_r^T} + \overline{\lambda}\mathbf{\Lambda}_r^{-1}\mathbf{M}_{rr}\mathbf{\Lambda}_r^{-1}\mathbf{\Phi}_r^T + \mathrm{O}(\overline{\lambda}^2) + \cdots$$

: Known as *Neuman series* of matrix

Considering…

$$\Rightarrow \quad \mathbf{\Phi}_r\mathbf{\Lambda}_r^{-1}\mathbf{\Phi}_r^T = \hat{\mathbf{K}}^{-1} - (\mathbf{\Phi}_d)(\mathbf{\Lambda}_d)^{-1}(\mathbf{\Phi}_d)^T = \mathbf{F}_{rs}$$

: Residual modes are expressed with the dominants term.

8. The global displacement vector is approximated as

$$\overline{\mathbf{u}}_g = \mathbf{T}_0^d\underline{\mathbf{\eta}_p^d} + \lambda\hat{\mathbf{\Psi}}\mathbf{F}_{rs}\hat{\mathbf{\Psi}}^T\mathbf{M}_g\mathbf{T}_0^d\underline{\mathbf{\eta}_p^d}$$

9. **Enhanced transformation matrix** is defined

$$\overline{\mathbf{u}}_g = \overline{\mathbf{T}}_1\mathbf{\eta}_\mathrm{p}^\mathrm{d} \quad \text{with} \quad \underline{\overline{\mathbf{T}}_1 = \overline{\mathbf{T}}_0 + \lambda\overline{\mathbf{T}}_r}, \quad \overline{\mathbf{T}}_r = \hat{\mathbf{\Psi}}\mathbf{F}_{rs}\hat{\mathbf{\Psi}}^T\mathbf{M}_g\mathbf{T}_0^d$$

: Unknown λ is another issue.

Enhanced transformation: Same size with $\overline{\mathbf{T}}_0$

Error estimator for the AMLS method

1. Mass orthonormality and stiffness orthogonality

$$\frac{1}{\lambda_i}(\boldsymbol{\varphi}_g)_i^T \mathbf{K}_g (\boldsymbol{\varphi}_g)_i = (\boldsymbol{\varphi}_g)_i^T \mathbf{M}_g (\boldsymbol{\varphi}_g)_i \Leftarrow$$

$$(\boldsymbol{\varphi}_g) \approx (\overline{\boldsymbol{\varphi}}_g)_i = \overline{\mathbf{T}}_1 (\overline{\boldsymbol{\varphi}}_p)_i = (\overline{\mathbf{T}}_0 + \lambda \overline{\mathbf{T}}_r)(\overline{\boldsymbol{\varphi}}_p)_i$$

$$\xi_i \approx (\overline{\boldsymbol{\varphi}}_p)_i^T [\underline{\bar{\lambda}_i \overline{\mathbf{T}}_0^T \mathbf{M}_g \overline{\mathbf{T}}_r} + \cancel{\bar{\lambda}_i^2 \overline{\mathbf{T}}_r^T \mathbf{M}_g \overline{\mathbf{T}}_r}](\overline{\boldsymbol{\varphi}}_p)_i \Rightarrow \mu_i = \bar{\lambda}_i (\overline{\boldsymbol{\varphi}}_p)_i^T \mathbf{A} \mathbf{F}_{rs} \mathbf{A}^T (\overline{\boldsymbol{\varphi}}_p)_i$$

$$= \mathbf{A}\mathbf{F}_{rs}\mathbf{A}^T \quad \text{with} \quad \mathbf{A} = \boldsymbol{\Phi}_d^T \widetilde{\mathbf{M}}_g$$

[Note]

Calculated during the recursive transformation procedure of the mass matrix: $\overline{\mathbf{M}} = \boldsymbol{\Phi}_d^T \widetilde{\mathbf{M}}_g \boldsymbol{\Phi}_d$

2. Symmetry of matrix $\mathbf{F}_{rs}$

$$\mathbf{F}_{rs} = \mathbf{F}_d + \mathbf{F}_u + \mathbf{F}_u^T$$

3. Finally, we can define an error estimator

$$\mu_i = \bar{\lambda}_i (\overline{\boldsymbol{\varphi}}_p)_i^T \mathbf{E} (\overline{\boldsymbol{\varphi}}_p)_i \quad \text{with} \quad \mathbf{E} = \mathbf{E}_1 + \mathbf{E}_2 + \mathbf{E}_2^T \Rightarrow$$

$$\mathbf{E}_1 = \mathbf{A}\mathbf{F}_d\mathbf{A}^T, \quad \mathbf{E}_{i,j}^1 = \sum_{k=1}^{N_S} \mathbf{A}_{i,k} \mathbf{F}_k^d \mathbf{A}_{k,j}^T$$

$$\mathbf{E}_2 = \mathbf{A}\mathbf{F}_u\mathbf{A}^T, \quad \mathbf{E}_{i,j}^2 = \sum_{k=1}^{N_S} \mathbf{A}_{i,k} \mathbf{F}_k^u \mathbf{A}_{k,j}^T$$

$\overline{N}_p \times \overline{N}_p$ matrix

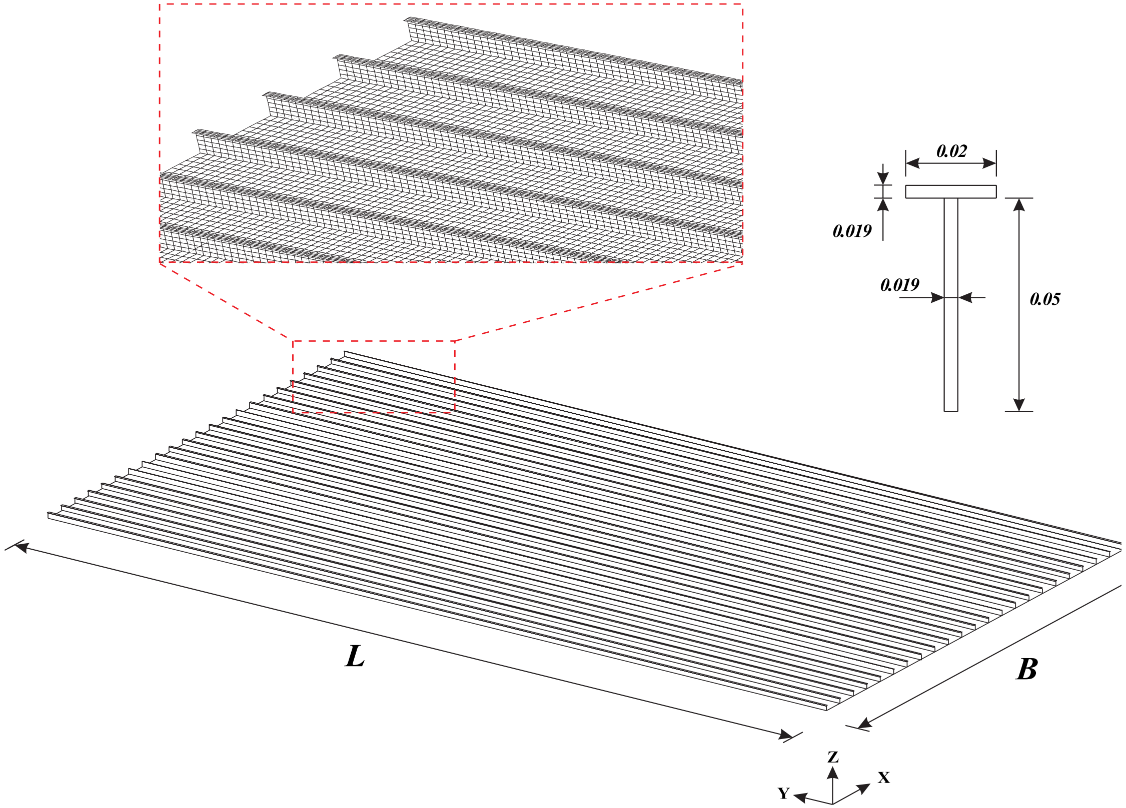
Therefore, we do not need to handle the matrices of global DOF size.

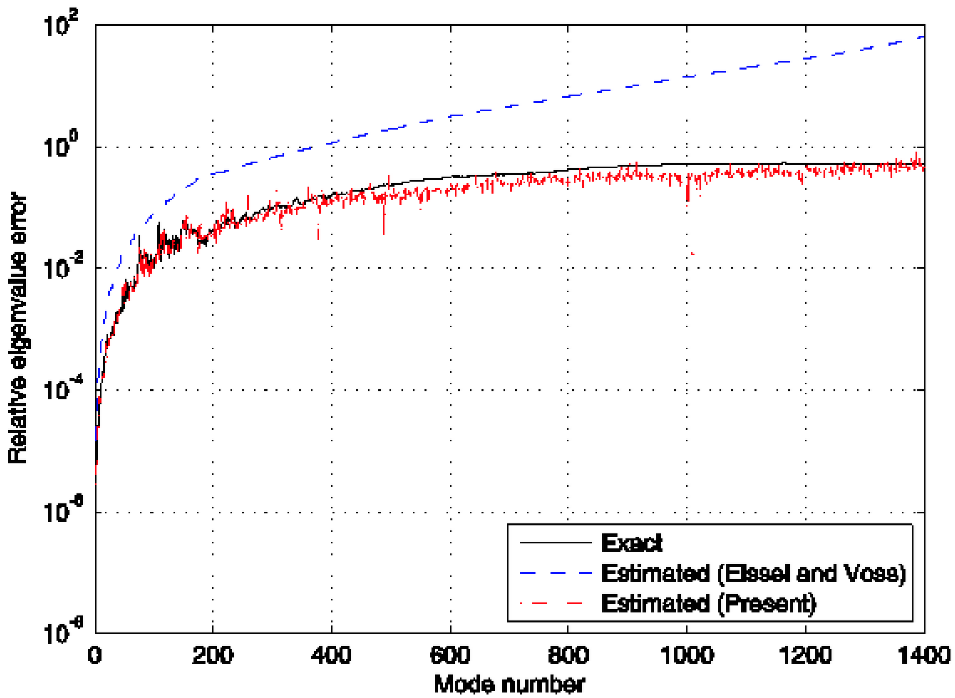
"E" matrix is simply computed with submatrix additions and multiplications.

❖ Stiffened plate structure

Dimension (L=78 m, B=44 m), <u>**1,023**</u> **substructures (Using METIS)**. Free BC is imposed.

Total DOFs = **1,004,088 →** **Retain modes = 2,200**. **Reduced system size = 5,450 (0.5 %)**.





< Relative eigenvalue errors >

❖ Stiffened plate structure

➢ Breakdowns of the computational cost

Items		Related Equations	Computation times [sec]	Ratio [%]
AMLS	Transformation procedures	13	3355.02	96.74
	Solution of the reduced eigenvalue problem	14	113.21	3.26
	Total	-	3468.23	100.00
Error estimation	Calculation of the residual flexibility matrix $\mathbf{F}_{rs}$	36	36.37	1.05
	Construction of $\mathbf{A}$ and $\mathbf{E}$ matrices	48, 51	227.31	6.55
	Calculation error estimator μ_i	50	5.83	0.17
	Total	-	269.51	7.77

The additional computation cost is about 7% compared to that of AMLS method to calculate the relative eigenvalue errors in 1,400 global modes.

❖ **"Lanczos (*eigs* in MATLAB)" : 1,400 global modes (9,500 sec). 1,500 global modes (N/A)**

Closure

1. To develop the error estimation method for the AMLS method, **an enhanced transformation matrix** was newly developed which considered the residual mode effect.

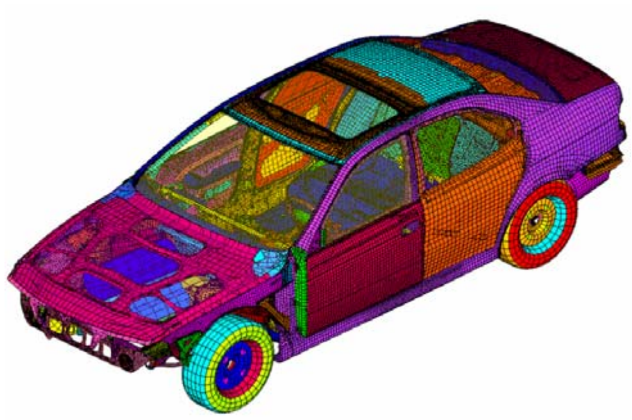
2. Using the enhanced transformation matrix, we proposed **an error estimator for the AMLS method.**

[Note]

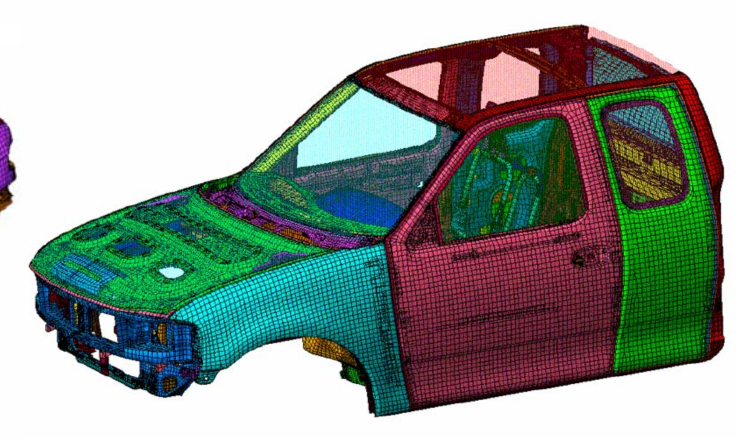
- ❖ **Error estimators** for **CB / AMLS** methods were introduced in the above.
- ❖ The success to develop the error estimator comes from "**Construction of the enhanced transformation matrix**".
- ❖ Therefore, it is possible to develop **the error estimators for any other CMS methods.**

Topic 2. Enhanced AMLS method

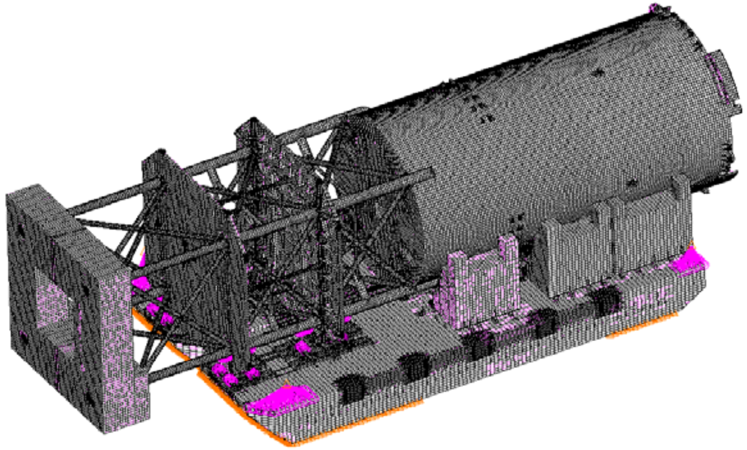
Issue



1,000,000 DOFs



2,000,000 DOFs



3,000,000 DOFs

Demands for more accurate method / large structural analyses:

- ✓ **<u>LANCZOS algorithm has a limitation</u>** to solve large DOFs problem.
- ✓ **<u>AMLS</u>** method was developed **(Alternative of LANCZOS).**
- ✓ Necessary to develop a powerful method **<u>outperforming the AMLS method</u>.**

Enhanced AMLS method

1. The enhanced transformation matrix

$$\mathbf{u}_g \approx \overline{\mathbf{u}}_g = \overline{\mathbf{T}}_1 \boldsymbol{\eta}_p^d \text{ with } \overline{\mathbf{T}}_1 = \overline{\mathbf{T}}_0 + \lambda \overline{\mathbf{T}}_r,\ \overline{\mathbf{T}}_r = \hat{\boldsymbol{\Psi}} \mathbf{F}_{rs} \hat{\boldsymbol{\Psi}}^T \mathbf{M}_g \mathbf{T}_0^d$$

Unknown

How can we handle the unknown λ ?

2. Employing O'Callahan's technique

$$\overline{\mathbf{T}}_0 \Rightarrow \overline{\mathbf{M}}_p \ddot{\overline{\boldsymbol{\eta}}}_p + \overline{\mathbf{K}}_p \overline{\boldsymbol{\eta}}_p = \mathbf{0} \Rightarrow -\lambda \overline{\mathbf{M}}_p \overline{\boldsymbol{\eta}}_p + \overline{\mathbf{K}}_p \overline{\boldsymbol{\eta}}_p = \mathbf{0} \Rightarrow \lambda \overline{\boldsymbol{\eta}}_p = \overline{\mathbf{M}}_p^{-1} \overline{\mathbf{K}}_p \overline{\boldsymbol{\eta}}_p$$

Reduced model of original AMLS method

$$\Rightarrow \overline{\mathbf{T}}_1 = \overline{\mathbf{T}}_0 + \lambda \overline{\mathbf{T}}_r = \overline{\mathbf{T}}_0 + \hat{\boldsymbol{\Psi}} \mathbf{F}_{rs} \hat{\boldsymbol{\Psi}}^T \mathbf{M}_g \mathbf{T}_0^d \overline{\mathbf{M}}_p^{-1} \overline{\mathbf{K}}_p$$

New transformation matrix.

3. Reduced mass and stiffness matrices

$$\tilde{\mathbf{M}}_p = \overline{\mathbf{T}}_1^T \mathbf{M}_g \overline{\mathbf{T}}_1 = \underline{\overline{\mathbf{M}}_p} + \overline{\mathbf{T}}_r^T \mathbf{M}_g \overline{\mathbf{T}}_0 + \overline{\mathbf{T}}_0^T \mathbf{M}_g \overline{\mathbf{T}}_r + \overline{\mathbf{T}}_r^T \mathbf{M}_g \overline{\mathbf{T}}_r$$

$$\tilde{\mathbf{K}}_p = \overline{\mathbf{T}}_1^T \mathbf{K}_g \overline{\mathbf{T}}_1 = \underline{\overline{\mathbf{K}}_p} + \overline{\mathbf{T}}_r^T \mathbf{K}_g \overline{\mathbf{T}}_0 + \overline{\mathbf{T}}_0^T \mathbf{K}_g \overline{\mathbf{T}}_r + \overline{\mathbf{T}}_r^T \mathbf{K}_g \overline{\mathbf{T}}_r$$

✓ **Reduced matrices of original AMLS method**

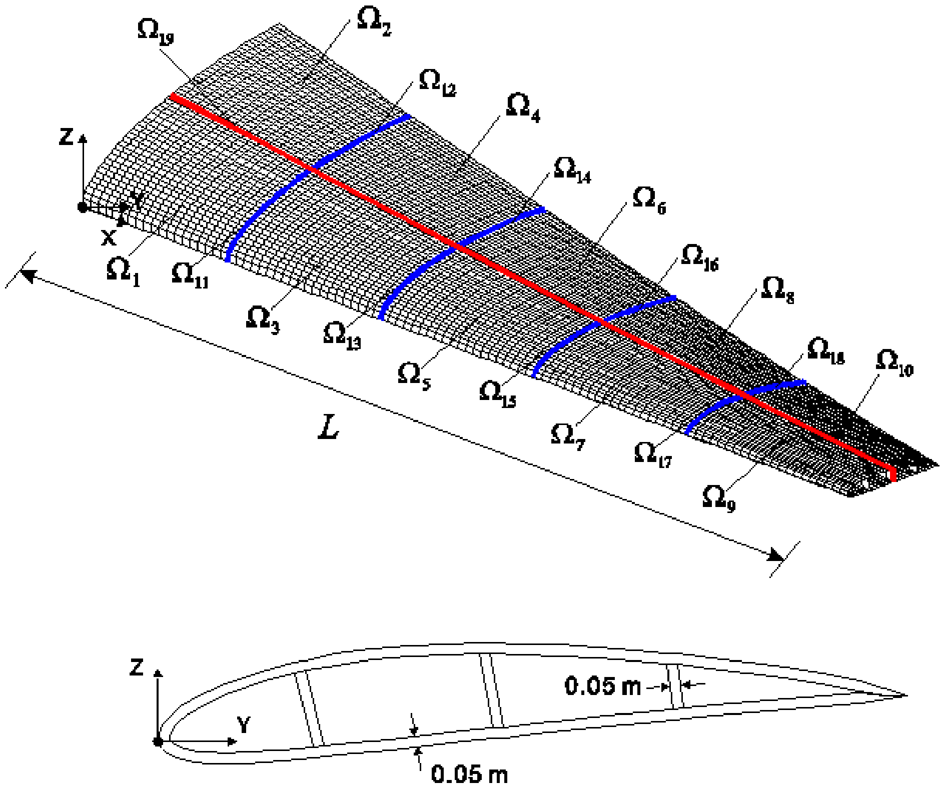
4. Reduced eigenvalue problem

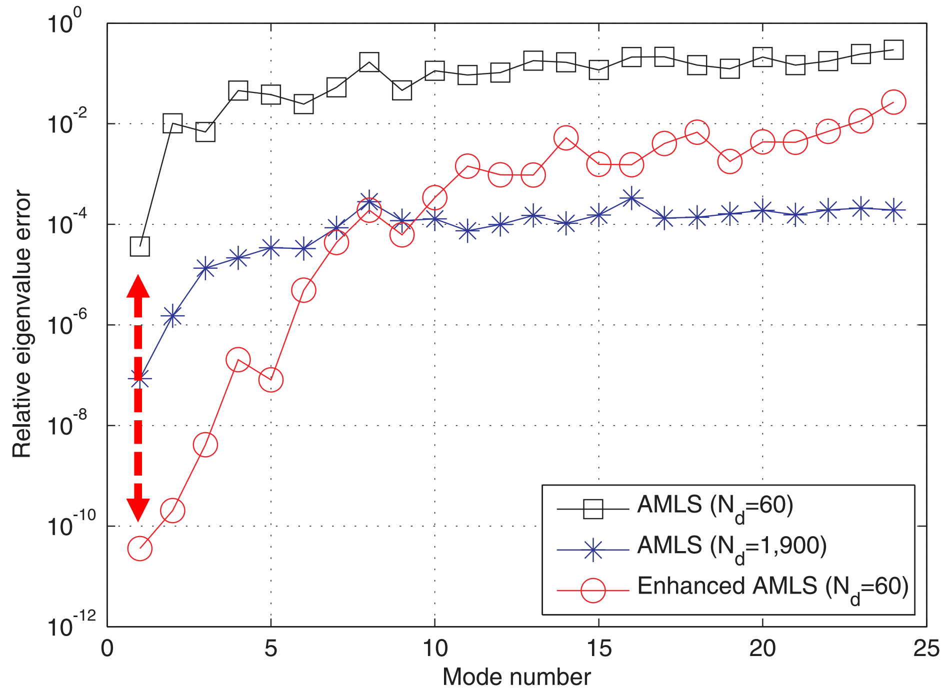
$$\tilde{\mathbf{K}}_p (\overline{\boldsymbol{\varphi}})_i = \overline{\lambda}_i \tilde{\mathbf{M}}_p (\overline{\boldsymbol{\varphi}})_i \Rightarrow (\boldsymbol{\varphi}_g)_i \approx (\overline{\boldsymbol{\varphi}}_g)_i = \overline{\mathbf{T}}_1 (\overline{\boldsymbol{\varphi}}_p)_i$$

❖ Turbine blade structure

Dimension (L=35 m, Thickness=0.05 m), **28 substructures**. **Fixed BC at $x = 0$.**

Total DOFs 51,308 → Retain modes 60. **Reduced system size 1,260 (2.45 %)**.





Accurate 10^5 times for 1^{st} mode.

Closure

1. The **enhanced transformation matrix** considering the residual mode effect is derived.

2. Using the enhanced transformation matrix, **the enhanced AMLS method** was presented.

❖ However, **there are limitations** to use **as an eigenvalue solver**.

- ✓ Does not have capacity to solve large DOFs problems **over 50,000 DOFs**.
- ✓ **Very complicated formulation** of the enhanced transformation matrix

$$\overline{\mathbf{T}}_1 = \overline{\mathbf{T}}_0 + \underline{\hat{\mathbf{\Psi}}\,\mathbf{F}_{rs}\,\hat{\mathbf{\Psi}}^T\,\mathbf{M}_g\,\mathbf{T}_0^d}\,\overline{\mathbf{M}}_p^{-1}\overline{\mathbf{K}}_p \Rightarrow$$

  - **Global matrix computation.**
  - **Fully populated matrix.**
  - **Induce lack of memory and huge costs.**

- ✓ Need to improve its **computational efficiency**.

  1) Formulation modification in the **submatrix computing level**.

  2) **An optimized algorithm** for computer programming.

Topic 3. Algebraic dynamic condensation method

Introduction

Improved Reduced System (IRS) method (O'Callahan, *1989*)

1. **Master / Slave DOFs selection, and reduction.**

$$\mathbf{u}_g = \begin{bmatrix} \mathbf{u}_s \\ \mathbf{u}_m \end{bmatrix} \approx \bar{\mathbf{u}}_g = \mathbf{T}_1 \mathbf{u}_m \quad \text{with} \quad \mathbf{T}_1 = \mathbf{T}_G + \mathbf{T}_a \mathbf{H}_G \quad (= \mathbf{T}_G + \lambda \mathbf{T}_a)$$

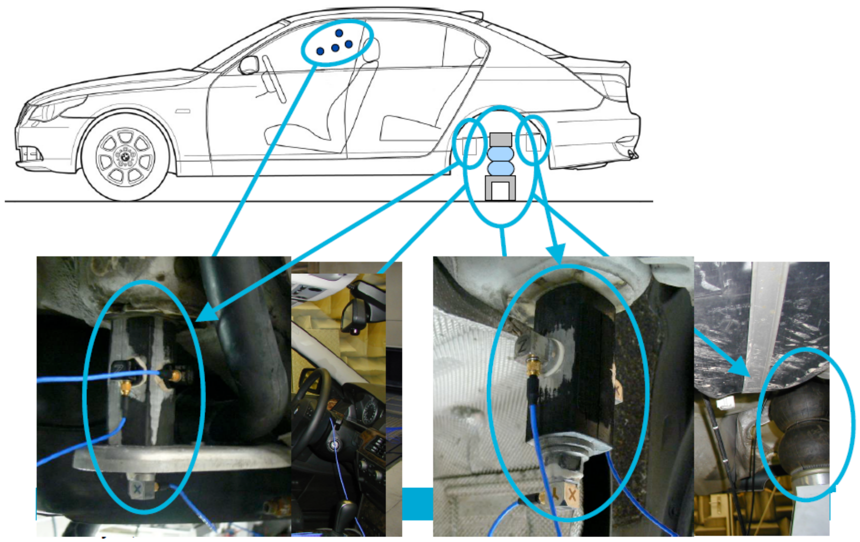
$$\underbrace{\mathbf{T}_G = \begin{bmatrix} -\mathbf{K}_s^{-1}\mathbf{K}_{sm} \\ \mathbf{I}_m \end{bmatrix}}_{\text{Guyan's transformation matrix}}, \quad \mathbf{T}_a = \begin{bmatrix} \mathbf{K}_s^{-1}(\mathbf{M}_{sm} - \mathbf{M}_s \mathbf{K}_s^{-1}\mathbf{K}_{sm}) \\ \mathbf{0} \end{bmatrix}, \quad \mathbf{H}_G = \overline{\mathbf{M}}_G^{-1}\overline{\mathbf{K}}_G$$

2. **Considering inertial effect of slave DOFs**

   → Good accuracy.

3. **Very simple formulation.**

$$\overline{\mathbf{M}}_1 = \mathbf{T}_1^T \mathbf{M}_g \mathbf{T}_1 \ , \quad \overline{\mathbf{K}}_1 = \mathbf{T}_1^T \mathbf{K}_g \mathbf{T}_1$$



4. **No need to sub- eigenvalue analyses.**

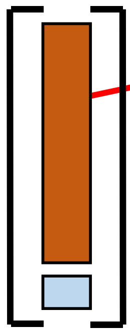
Issue

What is the drawback / limitation?

✓ Cannot apply to large FE model (over 100,000 DOFs).

: IRS transformation matrix

$$\mathbf{T}_1 = \begin{bmatrix} -\mathbf{K}_s^{-1}\mathbf{K}_{sm} + \mathbf{K}_s^{-1}(\mathbf{M}_{sm} - \mathbf{M}_s\mathbf{K}_s^{-1}\mathbf{K}_{sm}) \\ \mathbf{I}_m \end{bmatrix}, \quad \mathbf{T}_1 =$$

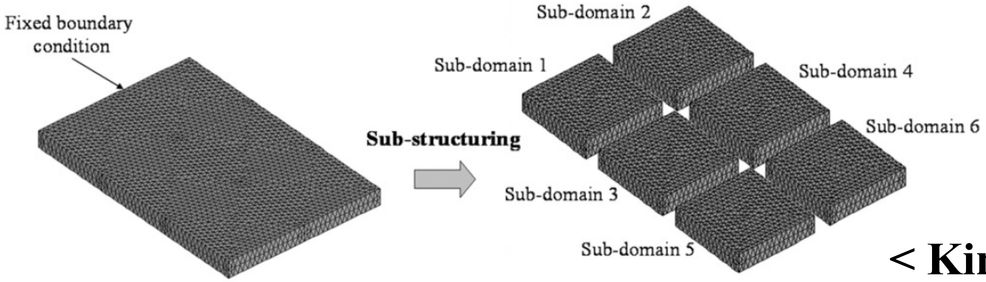


Fully populated part

→ Induce large costs.

Require large memory.

✓ "Domain based partitioning" is not suitable



Substructures: Still large DOFs

→ Large matrix inversion $\mathbf{K}_s^{-1}$

< Kim and Cho, *IJNME*, 2007 >

✓ Rayleigh-Ritz procedure using transformation matrix

$$\overline{\mathbf{M}} = \mathbf{T}_1^T\mathbf{M}_g\mathbf{T}_1, \quad \overline{\mathbf{K}} = \mathbf{T}_1^T\mathbf{K}_g\mathbf{T}_1$$: Expensive global matrix operations.

Design focus

❖ How do we improve the **reduction efficiency** of the IRS method ?

1. Design focus ?

✓ Improving "**computational efficiency**" of the IRS method.

→ **Avoiding** expensive **global matrix operations** and **matrix populations**.

2. Key ideas ?

✓ Employing "**algebraic substructuring**" algorithm.

:Using "**METIS**" adopted in the AMLS method.

✓ **Submatrix** operations.

✓ Transformation **without the global transformation matrix**.

3. What is the "**Strengths**", which is expected ?

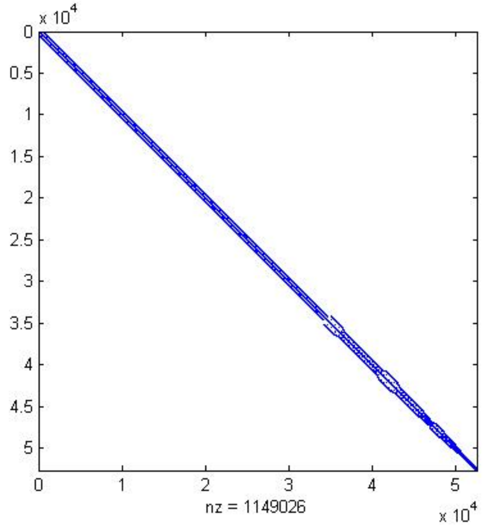
✓ Formulations would be **expressed in submatrix form**: Computationally efficient.

✓ We can **handle** practical engineering problems with **large DOFs in PC**.

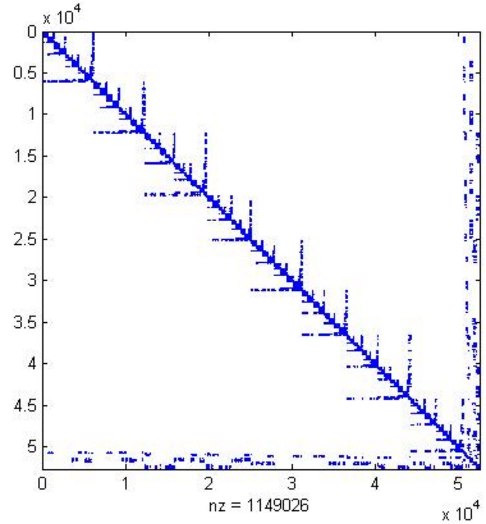
Algebraic dynamic condensation

Step 1. Algebraic substructuring

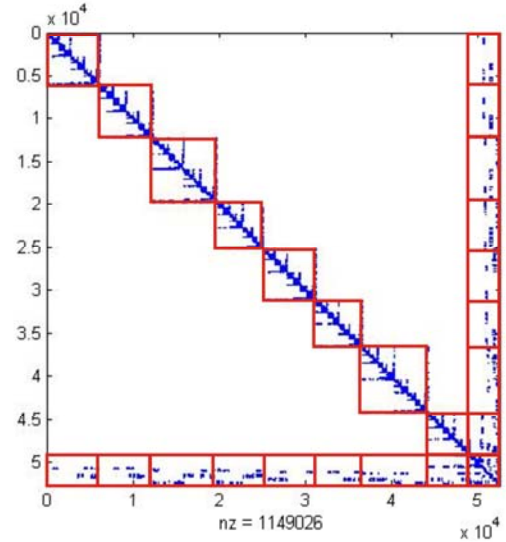
(1) A sparse matrix



(2) Matrix permutation



(3) Partitioned into submatrices



$$\mathbf{M}_g = \left[\begin{array}{ccc|c} \mathbf{M}_1 & & \mathbf{0} & \mathbf{M}_1^c \\ & \ddots & & \vdots \\ \mathbf{0} & & \mathbf{M}_n & \mathbf{M}_n^c \\ \hline (\mathbf{M}_1^c)^T & \cdots & (\mathbf{M}_n^c)^T & \mathbf{M}_b \end{array}\right], \quad \mathbf{K}_g = \left[\begin{array}{ccc|c} \mathbf{K}_1 & & \mathbf{0} & \mathbf{K}_1^c \\ & \ddots & & \vdots \\ \mathbf{0} & & \mathbf{K}_n & \mathbf{K}_n^c \\ \hline (\mathbf{K}_1^c)^T & \cdots & (\mathbf{K}_n^c)^T & \mathbf{K}_b \end{array}\right]$$

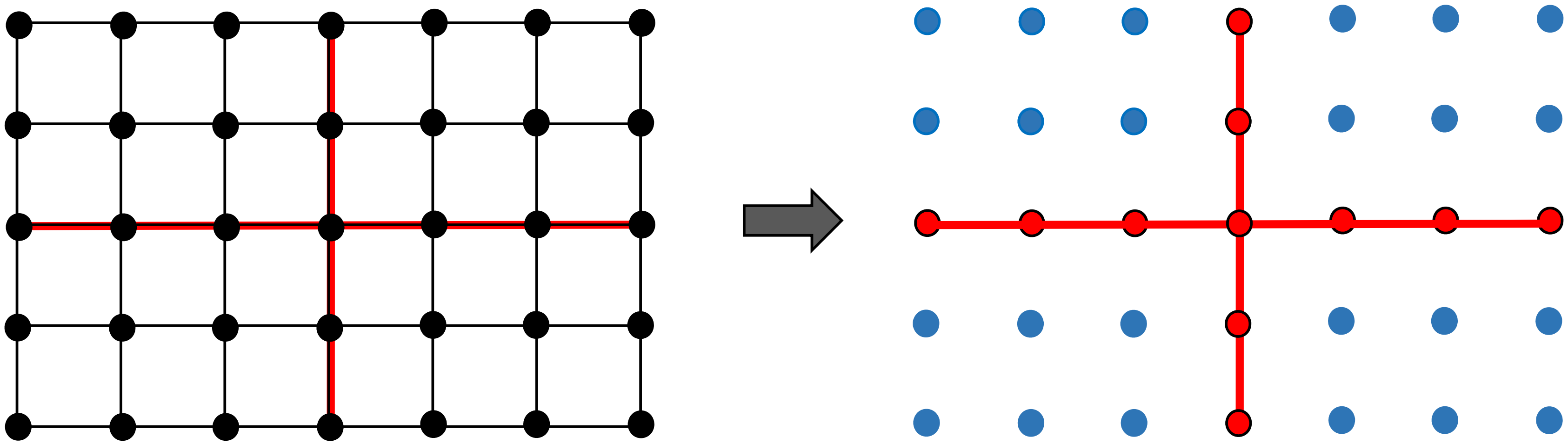
[Note]

Matrix permutation = Renumbering nodes in FE model.

→ Does not alter the physical characteristic of original FE model.

Algebraic dynamic condensation

Step 2. Substructural stiffness condensation



1. Equations of motion

$$\begin{bmatrix} \mathbf{K}_s & \mathbf{K}_c \\ \mathbf{K}_c^T & \mathbf{K}_b \end{bmatrix}\begin{bmatrix} \mathbf{u}_s \\ \mathbf{u}_b \end{bmatrix} = \lambda \begin{bmatrix} \mathbf{M}_s & \mathbf{M}_c \\ \mathbf{M}_c^T & \mathbf{M}_b \end{bmatrix}\begin{bmatrix} \mathbf{u}_s \\ \mathbf{u}_b \end{bmatrix}$$

2. Expanding the 1st row linear equation

$$\mathbf{u}_s = (\mathbf{K}_s - \lambda \mathbf{M}_s)^{-1}(\mathbf{K}_c - \lambda \mathbf{M}_c)\mathbf{u}_b = [\underline{\boldsymbol{\Psi}_c} + \lambda \mathbf{K}_s^{-1}\hat{\mathbf{M}}_c + o(\lambda^2) + \cdots]\,\mathbf{u}_b \,,$$

Considering…

Static condensation matrix

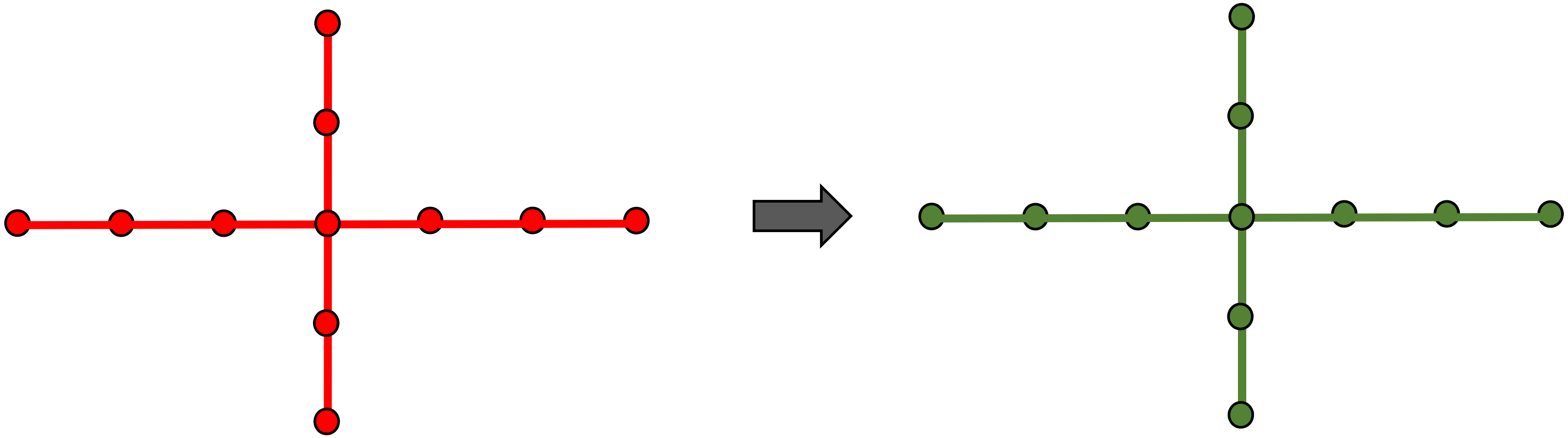
$$\boxed{\mathbf{u}_g \approx \bar{\mathbf{u}}_g = \begin{bmatrix} \bar{\mathbf{u}}_s \\ \mathbf{u}_b \end{bmatrix} = \boldsymbol{\Psi}\mathbf{u}_b \;\; \text{with} \;\; \boldsymbol{\Psi} = \begin{bmatrix} \boldsymbol{\Psi}_c \\ \mathbf{I}_b \end{bmatrix}}$$

3. Static condensation procedure

$$\hat{\mathbf{M}}_b = \boldsymbol{\Psi}^T \mathbf{M}_g \boldsymbol{\Psi} \;,\;\; \hat{\mathbf{K}}_b = \boldsymbol{\Psi}^T \mathbf{K}_g \boldsymbol{\Psi} \;\; \Rightarrow \;\; \boxed{\hat{\mathbf{M}}_b = \mathbf{M}_b + \sum_{i=1}^{n}(\mathbf{M}_i^c)^T \boldsymbol{\Psi}_i^c + \sum_{i=1}^{n}(\boldsymbol{\Psi}_i^c)^T \hat{\mathbf{M}}_i^c, \;\; \hat{\mathbf{K}}_b = \mathbf{K}_b + \sum_{i=1}^{n}(\mathbf{K}_i^c)^T \boldsymbol{\Psi}_i^c}$$

Algebraic dynamic condensation

Step 3. Interface boundary reduction



$N_b \times N_b$ matrix

: Still large size

$\overline{N}_b \times \overline{N}_b$ matrix

$\overline{N}_b$ is the number of the dominant interface normal modes.

1. Eigenvalue problem for interface boundary

$$\hat{\mathbf{K}}_b \mathbf{u}_b = \overline{\lambda}\, \hat{\mathbf{M}}_b \mathbf{u}_b \;\Rightarrow\; \mathbf{u}_b = \mathbf{\Phi}_b \mathbf{q}_b = [\underline{\mathbf{\Phi}_b^d} \;\; \mathbf{\Phi}_b^r] \begin{bmatrix} \mathbf{q}_b^d \\ \mathbf{q}_b^r \end{bmatrix} \;\Rightarrow\; \mathbf{u}_b \approx \overline{\mathbf{u}}_b = \mathbf{\Phi}_b^d \mathbf{q}_b^d$$

Considering…

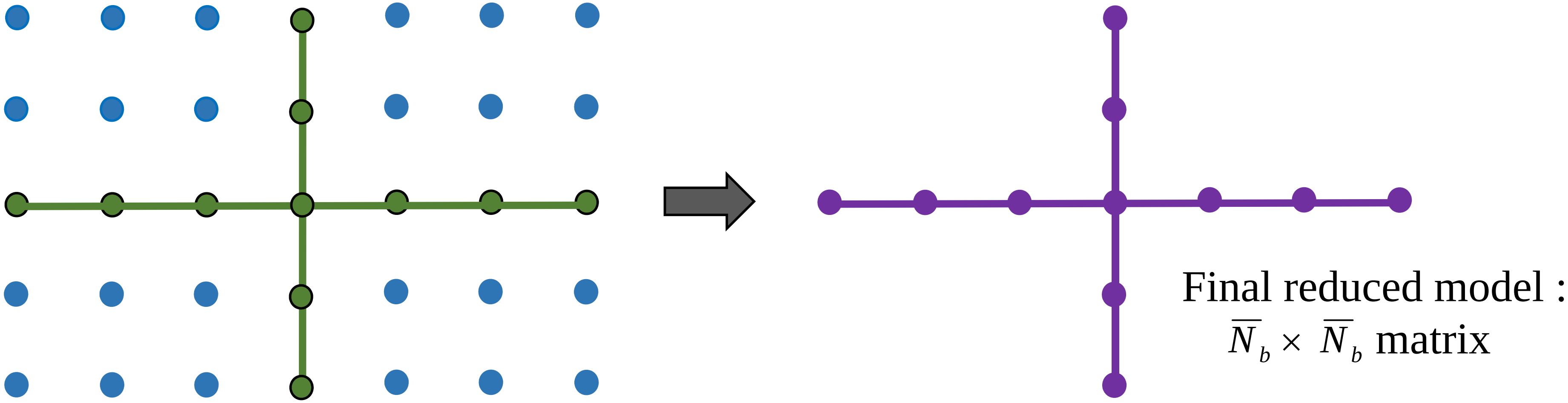
2. Reduce interface boundary

$$\overline{\mathbf{M}}_b = (\mathbf{\Phi}_b^d)^T \hat{\mathbf{M}}_b (\mathbf{\Phi}_b^d) \;, \quad \overline{\mathbf{K}}_b = (\mathbf{\Phi}_b^d)^T \hat{\mathbf{K}}_b (\mathbf{\Phi}_b^d)$$

Algebraic dynamic condensation

Step 4. Substructural inertial effect condensation

Inertial effect



1. Expanded the 1st row linear equation

$$\mathbf{u}_s = (\mathbf{K}_s - \lambda \mathbf{M}_s)^{-1}(\mathbf{K}_c - \lambda \mathbf{M}_c)\mathbf{u}_b = [\underbrace{\boldsymbol{\Psi}_c + \lambda \mathbf{K}_s^{-1}\hat{\mathbf{M}}_c}_{\text{Considering…}} + o(\lambda^2) + \cdots]\,\mathbf{u}_b$$

$$\Rightarrow \quad \bar{\mathbf{u}}_s = (\boldsymbol{\Psi}_c + \lambda \mathbf{K}_s^{-1}\hat{\mathbf{M}}_c)\mathbf{u}_b, \qquad \bar{\mathbf{u}}_b = \boldsymbol{\Phi}_b^d\, \mathbf{q}_b^d$$

Inertial effect condensation matrix

$$\Rightarrow \quad \boxed{\bar{\mathbf{u}}_g = \begin{bmatrix} \bar{\mathbf{u}}_s \\ \bar{\mathbf{u}}_b \end{bmatrix} = \boldsymbol{\Psi}_1 \mathbf{q}_b^d \quad \text{with} \quad \boldsymbol{\Psi}_1 = \bar{\boldsymbol{\Psi}} + \lambda\, \boldsymbol{\Psi}_a}$$

2. Handling unknown λ

$$\hat{\mathbf{K}}_b\,\mathbf{u}_b = \bar{\lambda}\,\hat{\mathbf{M}}_b\,\mathbf{u}_b \quad \Rightarrow \quad \bar{\lambda}\,\mathbf{q}_b^d = \mathbf{H}_b \mathbf{q}_b^d \quad \text{with} \quad \mathbf{H}_b = \bar{\mathbf{M}}_b^{-1}\,\bar{\mathbf{K}}_b$$

Algebraic dynamic condensation

3. Inertial effect condensation procedure

No transformation matrix.

$$\tilde{\mathbf{M}}_b = \mathbf{\Psi}_1^T \mathbf{M}_g \mathbf{\Psi}_1$$

$$\tilde{\mathbf{K}}_b = \mathbf{\Psi}_1^T \mathbf{K}_g \mathbf{\Psi}_1$$

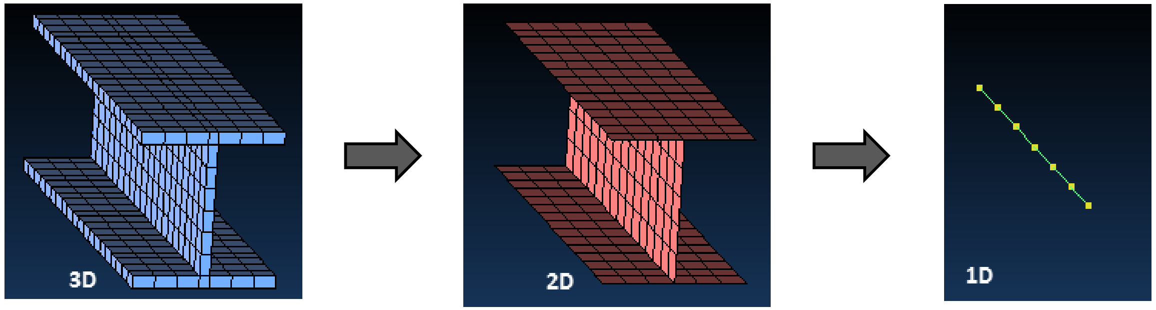
$$\tilde{\mathbf{M}}_b = \overline{\mathbf{M}}_b + \mathbf{R}_1 + \mathbf{R}_1^T + \mathbf{R}_2, \quad \tilde{\mathbf{K}}_b = \overline{\mathbf{K}}_b + \mathbf{H}_b^T \mathbf{R}_1$$

with

$$\mathbf{R}_1 = \sum_{i=1}^{n} (\mathbf{\Phi}_b^d)^T (\hat{\mathbf{M}}_i^c)^T \mathbf{A}_i \mathbf{H}_b \;, \quad \mathbf{R}_2 = \sum_{i=1}^{n} \mathbf{H}_b^T \mathbf{A}_i^T \mathbf{M}_i \mathbf{A}_i \mathbf{H}_b \;, \quad \mathbf{A}_i = \mathbf{K}_i^{-1} \hat{\mathbf{M}}_i^c (\mathbf{\Phi}_b^d)$$

✓ **Reduced system : Simple matrix summation and multiplications at a submatrix level.**

❖ Similar with "FEM degeneration" :



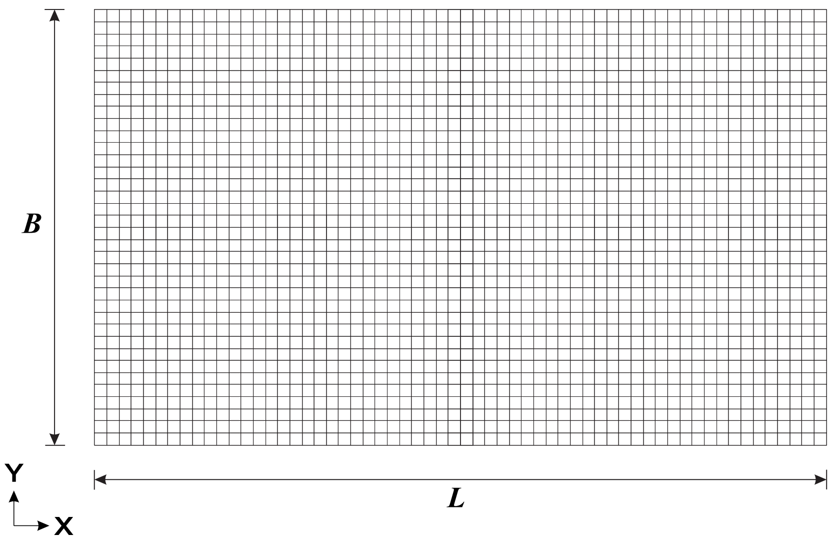
Step 5. Approximated global eigenvectors

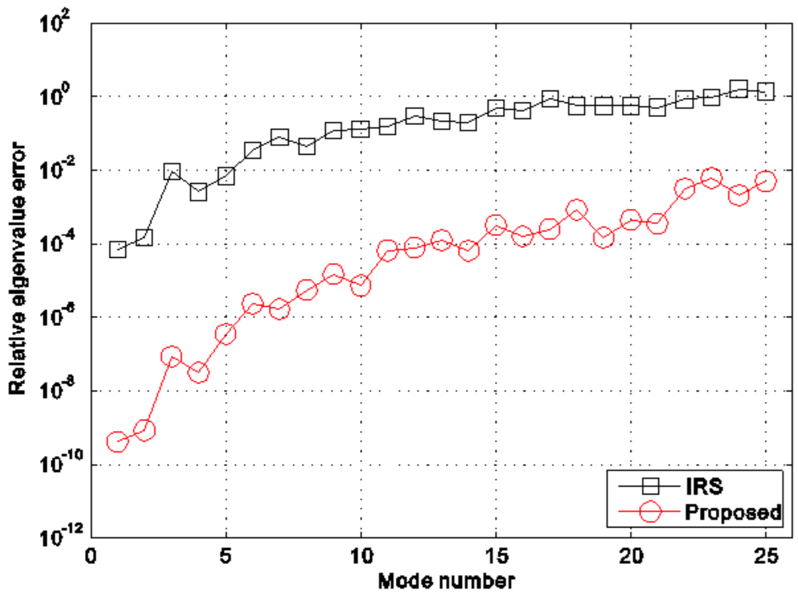
Transformation matrix: $$\mathbf{\Psi}_1 = \begin{bmatrix} \mathbf{\Psi}_1^1 \\ \mathbf{\Psi}_2^1 \\ \vdots \\ \mathbf{\Psi}_n^1 \\ \hline \mathbf{\Phi}_b^d \end{bmatrix} \quad \text{with} \quad \mathbf{\Psi}_i^1 = \mathbf{\Psi}_i^c \mathbf{\Phi}_b^d + \mathbf{K}_i^{-1} \hat{\mathbf{M}}_i^c \mathbf{\Phi}_b^d \mathbf{H}_b \quad \Rightarrow \quad (\overline{\boldsymbol{\varphi}}_g)_i = \mathbf{\Psi}_1 \overline{\boldsymbol{\varphi}}_i$$

❖ Rectangular plate problem

Dimension (L=20 m, B=12 m, Thickness=0.025 m), 16 substructures. No B.C.

Total DOFs = **11,285 → Retain interface modes = 100**. **Reduced system size = 100 (0.88 %)**.





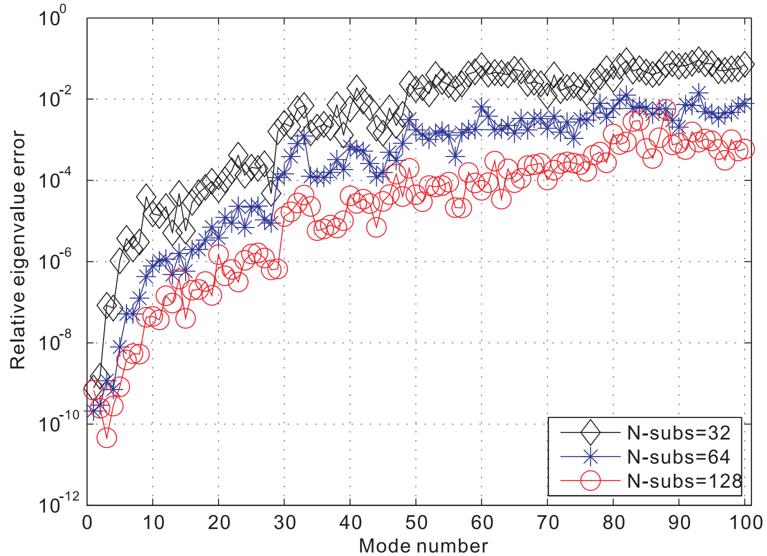
	Items	Related Equations	Computation times [sec]	Ratio [%]
IRS	Transformation procedure	8, 13	205.70	99.95
	Reduced eigenvalue problem	14	0.10	0.05
	Total	-	205.80	100.00
Proposed	Algebraic substructuring	16	0.05	0.02
	Calculation of $\hat{\mathbf{M}}_b$ and $\hat{\mathbf{K}}_b$ matrices	25	0.61	0.30
	Eigenvalue problem for the interface boundary	22	0.49	0.24
	Calculation of $\overline{\mathbf{M}}_b$, $\overline{\mathbf{K}}_b$ and $\mathbf{H}_b$ matrices	28, 33	0.03	0.01
	Calculation of $\tilde{\mathbf{M}}_b$ and $\tilde{\mathbf{K}}_b$ matrices	36	0.70	0.34
	Reduced eigenvalue problem	37	0.10	0.05
	Total	-	1.98	0.96

Compared to the computation time required for the IRS method: 100 times faster

❖ Truss structure

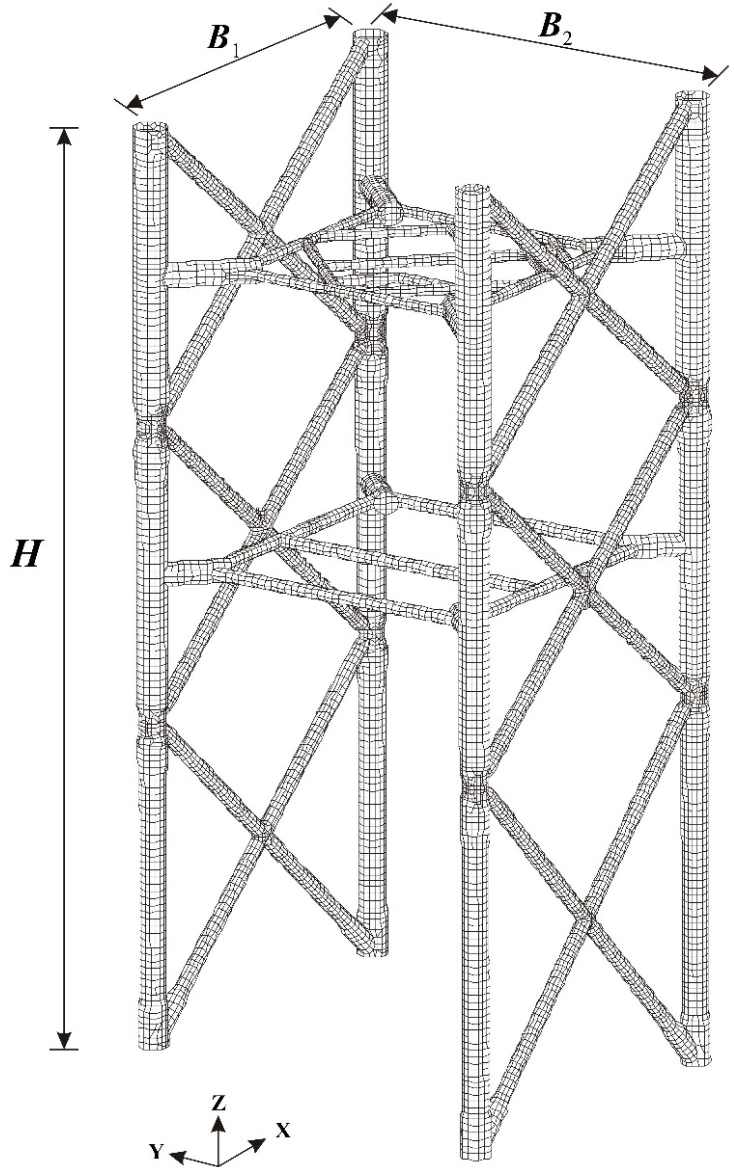
Dimension (L=98 m, B_1=45 m, B_2=48 m), No B.C.

Total DOFs = **155,766 → Retain interface modes = 150**. **Reduced system size = 150 (0.09 %)**.



Items	Number of substructures	Computation times [sec]
IIRS	-	N/A
Proposed	32	339.5
	64	168.6
	128	101.9

Faster



✓ **As the number of substructures increases,**

both accuracy and computation cost improve.

Closure

1. We develop a new efficient reduced-order modeling technique, named **algebraic dynamic condensation**.

2. The formulation can be simply **expressed using multiplications and summations of submatrices**, and thus it presents **excellent computational efficiency**.

3. The proposed method also provides **better accuracy** than the IRS method, and it can **handle relatively large FE models**, for which the IRS method fails to work.

4. Future works

Future works

Issue 1. Solution "Reliability"

- ✓ Error estimator applicable to the CB method **using the interface reduction technique**.
- ✓ **Iterative mode selection algorithms** to construct accurate reduced-order models.
- ✓ **More simplified error estimator** considering FE models with more than several millions of DOFs.

Issue 2. Solution "Accuracy"

- ✓ **A new eigenvalue solver** using the enhanced AMLS method

  → Increasing computational efficiency. Optimized algorithm for implementation.

Issue 3. Solution "Efficiency"

- ✓ **An effective iterative scheme** could also be developed.
- ✓ Employing the **multi-level algebraic substructuring** to solve FE models with more than several millions of DOFs.

Q & A