It is time to develop analysis tools based on boundary integral equation

Jianming Zhang

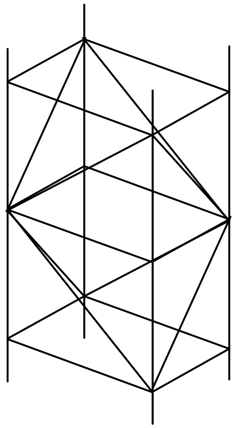
Nov. 17-19 2010

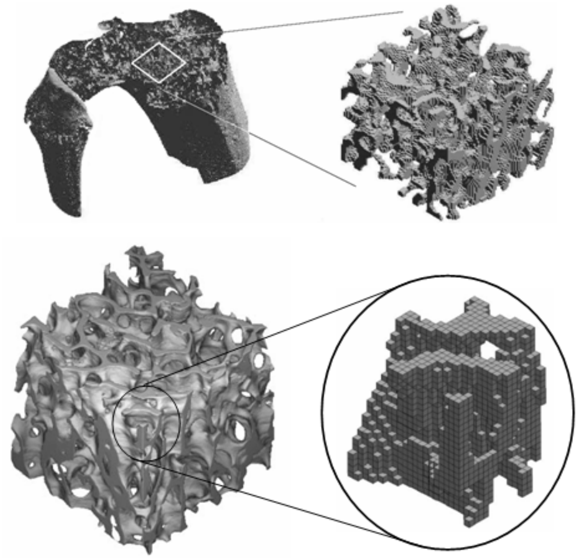
Outline

- Difficulties in FEM
- Review of BIE
- Boundary Face Method
- Numerical results by BFM
- Hierarchical Matrix and ACA
- Numerical results by ACA
- Software development
- Conclusions

Difficulties in FEM

- Continuous parametric model and Discrete model.
  - High quality meshing demanding considerable effort or skill
  - Interaction between CAE and CAD



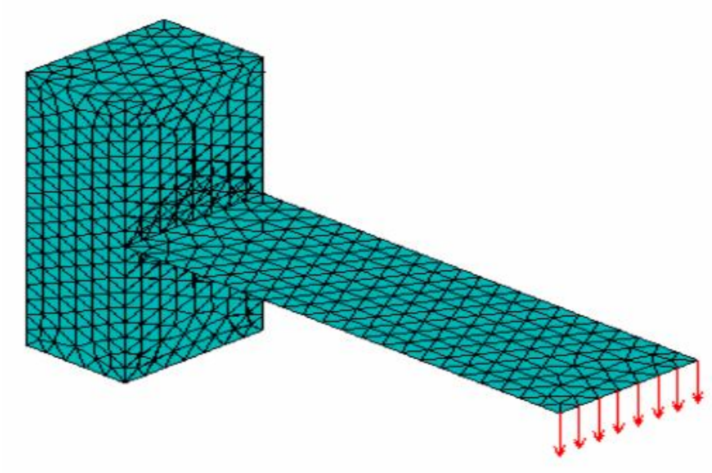


- Derivable trial function necessary in the weak form
  - Stiffer computational model
  - Contradiction between conforming and nonconforming elements

Difficulties in FEM（2）

- Many kinds of abstract element based on priori assumptions

  - Element performance relies on its shape. Small features are omitted due to connectivity and aspect ratio



  - Accuracy for stresses is of one order lower than displacements

  - New assumptions are required for connecting different kinds of elements, unable to capture local stress

  - Sound and solid training in FEM, rich skills and experiences are a must for a successful user. Analyst and designer are always not the same person

Desirable features of analysis tools for industrial daily design:

- Automatic meshing for complicated structures with complex geometry
- Complete solid modeling to capture local stress concentration
- Seamless interaction with CAD packages
- Fast computation ability to solve large scale problems

Complete solid stress analysis——
Boundary Face mehtod

Review of BIE

- 2D potential problem

$$\nabla^2 u = 0, \quad \forall x \in \Omega$$
$$u = \bar{u}, \quad \forall x \in \Gamma_u$$
$$\frac{\partial u}{\partial n} \equiv q = \bar{q}, \quad \forall x \in \Gamma_q$$

- The equivalent weak form

$$\int_\Omega v\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) dxdy + \int_{\Gamma_q} \bar{v}\left(k\frac{\partial u}{\partial n} - \bar{q}\right) d\Gamma = 0$$

- Once integration by part, FEM formulation

$$\int_\Omega \left(\frac{\partial v}{\partial x}\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\frac{\partial u}{\partial y}\right) dxdy - \int_{\Gamma_q} v\bar{q}\, d\Gamma = 0$$

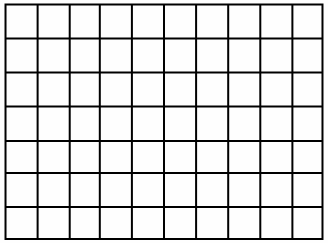
- Twice integration by part, BIE formulation

$$\int_\Omega u\nabla^2 v\, d\Omega - \oint_\Gamma \left(\frac{\partial v}{\partial n}u - v\frac{\partial u}{\partial n}\right) d\Gamma = 0$$

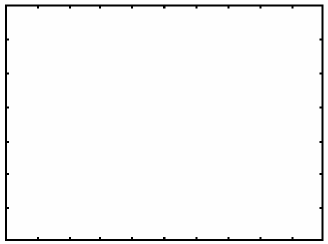
- Contradiction between conforming and nonconforming elements
- Locking problems: membrane locking, volumetric locking, shear locking etc.
- Reduced integration and hourglass modes
- Accuracy of fluxes is one order lower than that of potential

Advantages of boundary formulations:

- Easy mesh generation and modification

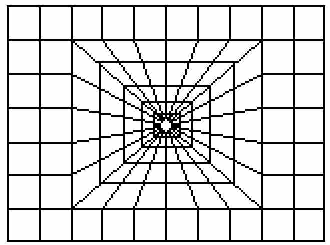


Domain type

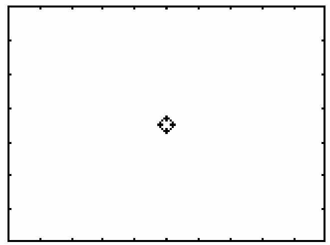


Boundary type

Adding a hole



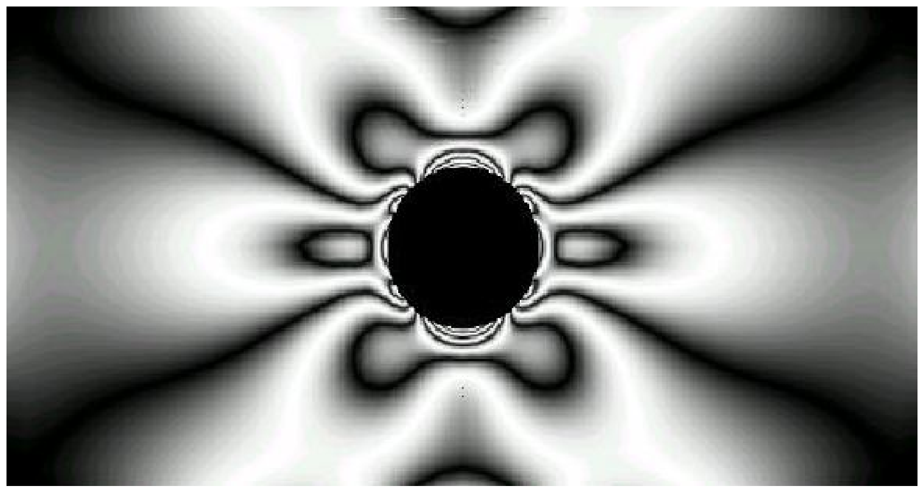
Domain type



Boundary type

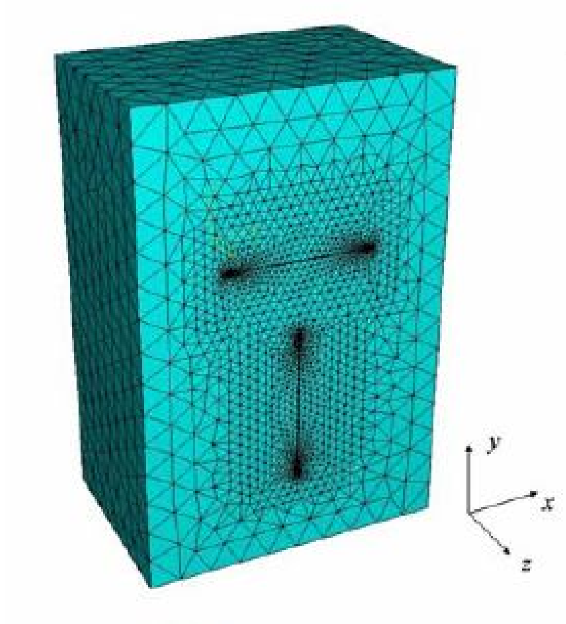
Potential to make direct use of a body's parametric representation through Brep data of CAD packages

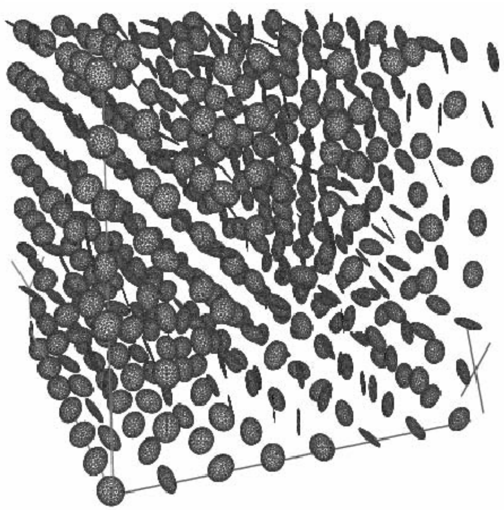
- High accuracy for local stress concentration

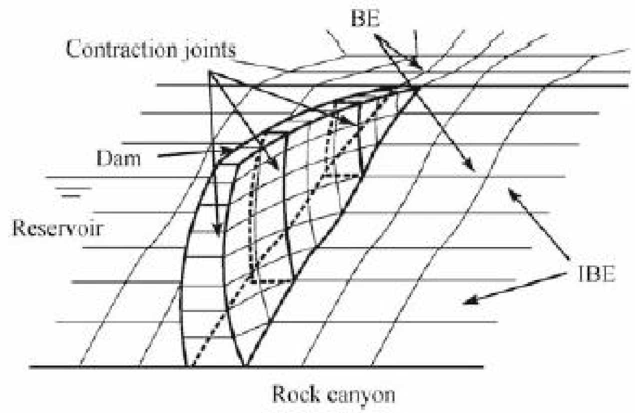


Advantages of boundary formulations (2)

- Suitable for solving singular problems

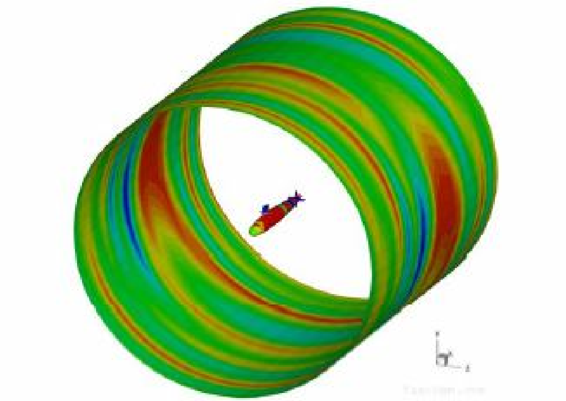






Complete coupled system of dam-canyon-reservoir

- Suitable for solving problems involving infinite domains



Acoustic fields from a Skipjack submarine model
(250,220 elements for a radiation model, *ka* = 38.4, solved in 54 min. *)

Advantages of boundary formulations (3)

- Natural way for imposing boundary conditions

（1）Boundary conditions are expressed by density functions. No abstract point load.

（2）For Robin condition, $\left.\left(\frac{\partial u}{\partial x}+\alpha u\right)\right|_S=\beta$

$$\int_\Omega v\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)dxdy+\int_{\Gamma_R}\overline{v}\left(k\frac{\partial u}{\partial n}+\alpha u-\overline{q}\right)d\Gamma=0$$

$$\int_\Omega\left(\frac{\partial v}{\partial x}\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\frac{\partial u}{\partial y}\right)dxdy-\int_{\Gamma_R}v\alpha u d\Gamma=\int_{\Gamma_R}v\overline{q}d\Gamma$$

$$\mathbf{Hu}-\mathbf{Gq}=\mathbf{0} \quad \Longrightarrow \quad \mathbf{Ax}=\mathbf{b}$$

Disadvantages of boundary formulations

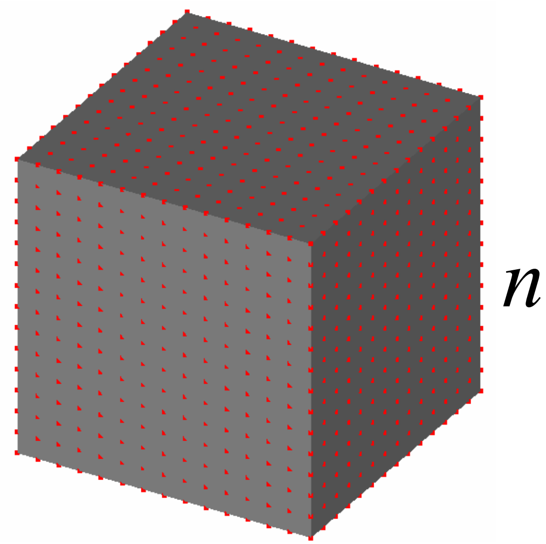
- Dense and unsymmetrical coefficient matrices
  - Memory complexity $O(N^2)$
  - CPU complexity Direct solver： $O(N^3)$
    Iterative solver： $O(N^2)$
- Requirement of fundamental solution
  - Applicable to linear problems only
- Singular and nearly singular integration involves complex mathematical operations

Breakthroughs in BEM techniques

- **Fast algorithms**
  - Fast Multipole Method

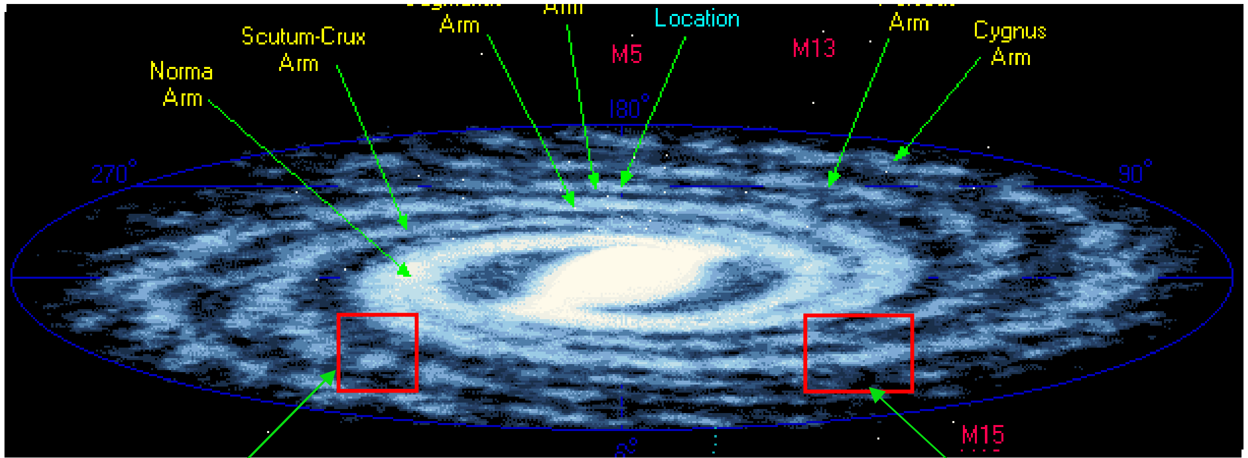
    Memory complexity: $O(N)$; CPU complexity: $O(N)$
  - Hierarchical Matrix and Adaptive Cross Approximation (ACA)

    Memory complexity: $O(N\log N)$; CPU complexity: $O(N\log^2 N)$



	Domain type methods (FEM, EFG, MLPG)	Boundary type methods (BEM, HdBNM)	Boundary type with linear complexity
Total degrees of freedom	$O(n^3)$	$O(n^2)$	$O(n^2)$
Memory requirement	$O(n^3)$	$O(n^4)$	$O(n^2)$
Time complexity	$O(n^3)$	$O(n^4)$	$O(n^2)$

Breakthroughs in BEM techniques --Fast algorithms



B_{local} Galaxy: 2×10^{11} stars B_{far}

$$F_i = \sum_{j=1}^{N} G m_i m_j / r_{ij}^2$$

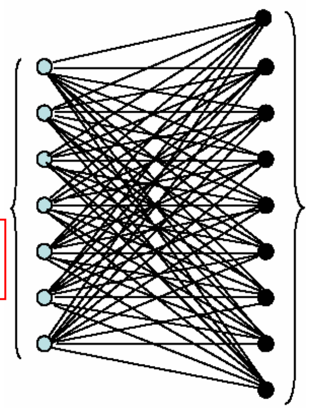
$$6\times(2\times10^{11})^2 \approx 2.4\times10^{23}$$

$$t = 7.6\times10^{6}\,\text{year}\ (10^{8}\,Flops)$$

FMM:

$$t = 3.3\ \text{Hours}\ (10^{8}\,Flops)$$

N B_{local}

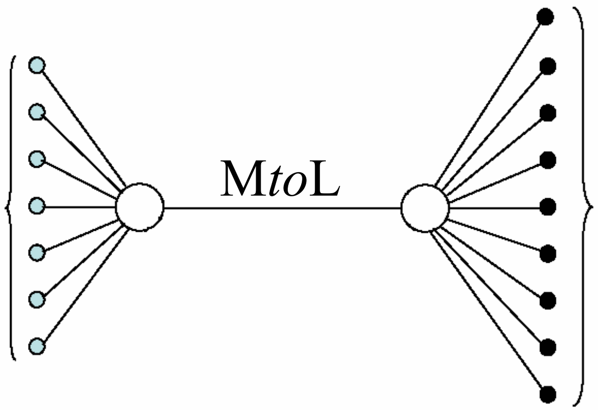


M B_{far}

Straightforward

Total number of operations O(NM)

N B_{local}



M B_{far}

Multipole expansion

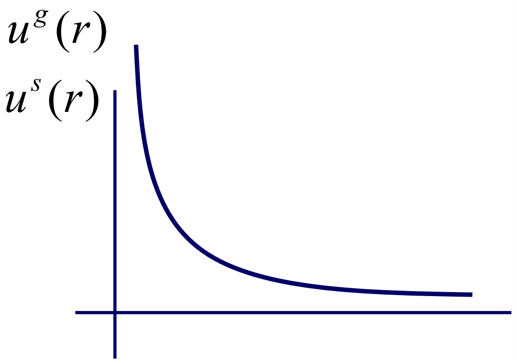
Total number of operations O(N+M)

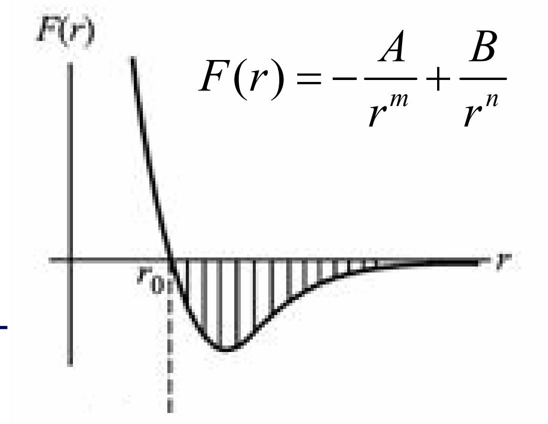
Breakthroughs in BEM techniques --Fast algorithms

$$u^g = \frac{Gm_1m_2}{r}$$

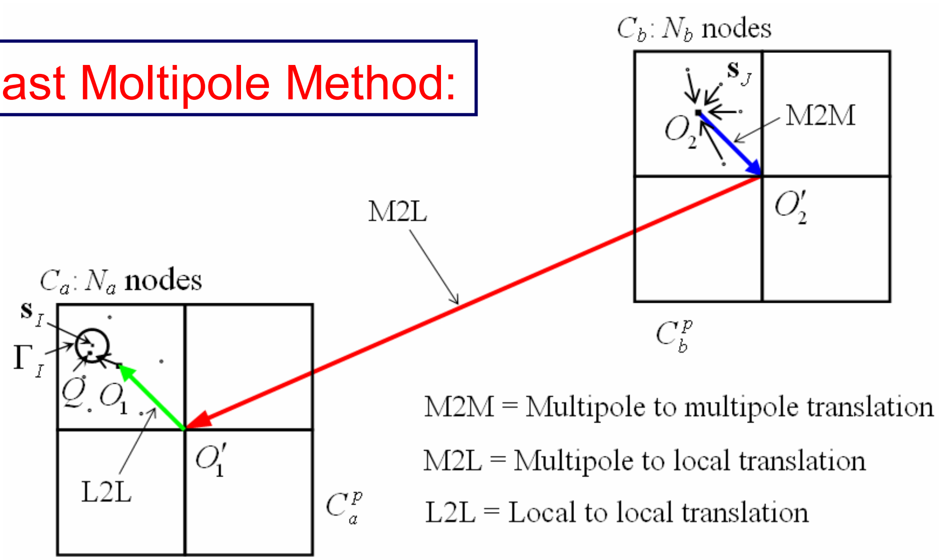
$$u^s(r) = \frac{1}{4\pi}\frac{1}{r(P,Q)}$$

$$u^s_{ij}(r) = \frac{1}{16\pi(1-v)Gr}\left[(3-4v)\delta_{ij} + r_{,i}r_{,j}\right]$$





Fast Moltipole Method:



$$M_{n'}^{m'}(Q_2') = \sum_{n=0}^{\infty}\sum_{m=-n}^{n} R_n^m(\overrightarrow{O_2'O_2})M_{n-n'}^{m-m'}(Q_2)$$

$$L_n^m(O_1') = \sum_{n'=0}^{\infty}\sum_{m'=-n'}^{n'} (-1)^n \overline{S_{n+n'}^{m+m'}}(\overrightarrow{O_1'O_2'})M_{n'}^{m'}(Q_2')$$

$$L_{n'}^{m'}(O_1) = \sum_{n=0}^{\infty}\sum_{m=-n}^{n} R_{n-n'}^{m-m'}(\overrightarrow{O_1'O_1})L_n^m(Q_1')$$

Breakthroughs in BEM techniques (2)

Breakthroughs in BEM techniques:

- **Dual reciprocity method**

$$\nabla^2 u = b \approx \sum_{i=1}^{N+L} \alpha_i \varphi_i(r) = \sum_{i=1}^{N+L} \alpha_i (\nabla^2 \phi_i(r))$$

$$c_i u_i + \int_\Gamma q^* u d\Gamma - \int_\Gamma u^* q d\Gamma = \sum_{k=1}^{N+L} \alpha_k (c_i \phi_{ik} + \int_\Gamma q^* \phi_k d\Gamma - \int_\Gamma u^* \frac{\partial \phi_k}{\partial n} d\Gamma)$$

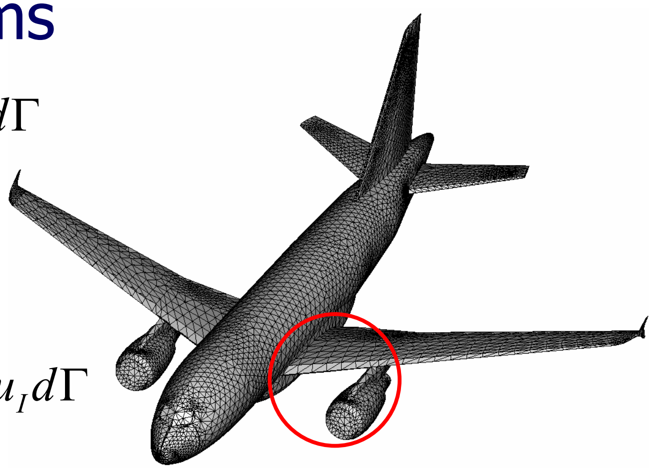
- **Accurate algorithm for nearly singular integration**

Therefore, an analysis tool for BVP that performs far better than existing techniques is achievable by using boundary integral equation formulations

Boundary Face Method

- The self-regular BIE for potential problems

$$0=\int_{\Gamma}(u(\mathbf{s})-u(\mathbf{y}))q^{s}(\mathbf{s},\mathbf{y})d\Gamma-\int_{\Gamma}q(\mathbf{s})u^{s}(\mathbf{s},\mathbf{y})d\Gamma$$



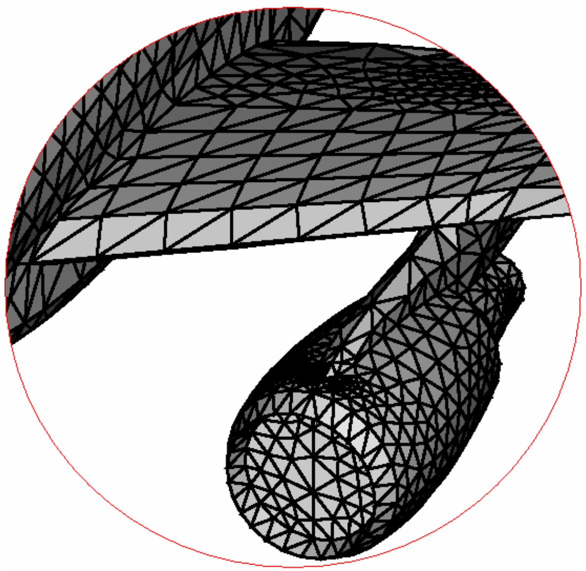
- The discretized form by elements

$$0=\sum_{j=1}^{N_e}\int_{\Gamma_j}u^{s}(\mathbf{s},\mathbf{y})\sum_{I=1}^{N_p}\Phi_I(\mathbf{s})q_I d\Gamma-\sum_{j=1}^{N_e}\int_{\Gamma_j}q^{s}(\mathbf{s},\mathbf{y})\sum_{I=1}^{N_p}(\Phi_I(\mathbf{s})-\Phi_I(\mathbf{y}))u_I d\Gamma$$

In standard BEM,

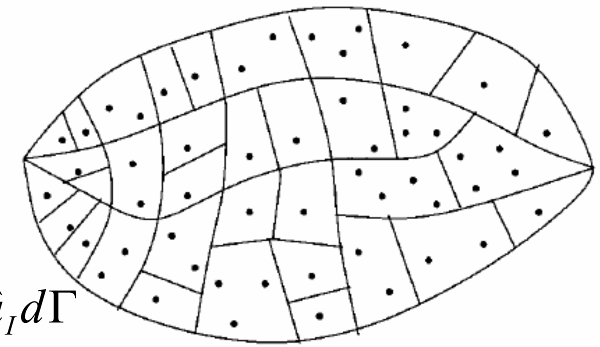
elements are used to

- **facilitate boundary integration**
- **interpolate Boundary variables**
- **approximate the geometry**



Boundary Face Method (2)

- The discretized form of BIE in BFM

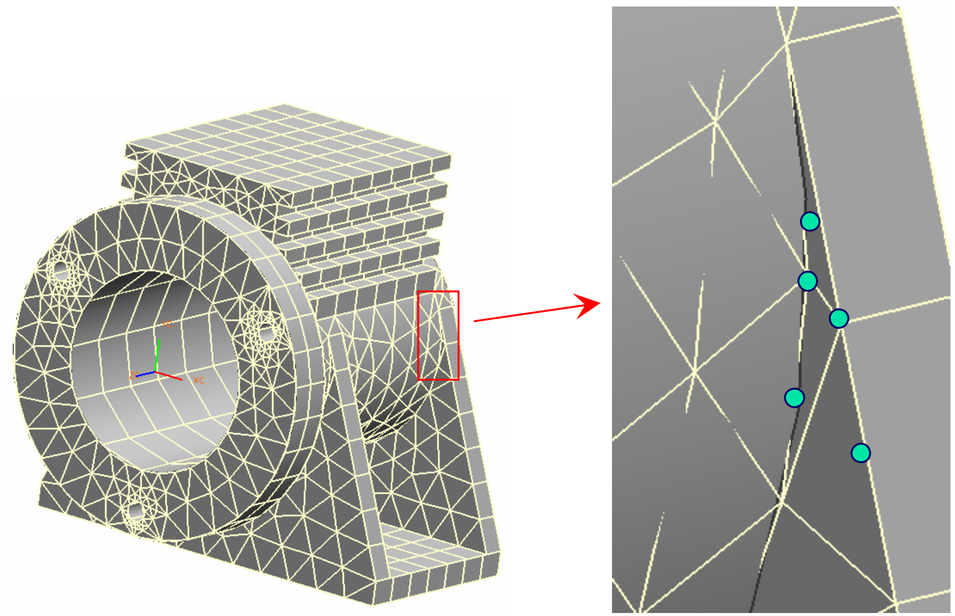


$$0=\sum_{j=1}^{N_c}\int_{\Gamma_j} u^s(\mathbf{s},\mathbf{y})\sum_{I=1}^{N}\Phi_I(\mathbf{s})\hat{q}_I\,d\Gamma-\sum_{j=1}^{N_c}\int_{\Gamma_j} q^s(\mathbf{s},\mathbf{y})\sum_{I=1}^{N}(\Phi_I(\mathbf{s})-\Phi_I(\mathbf{y}))\hat{u}_I\,d\Gamma$$

In the BFM,

elements are used to

- **facilitate boundary integration, only**
- **Shape functions are separated from the elements**
- **The exact geometry is kept**



Boundary Face Method (3)

- Element class in C++

```
class CElement
{
public:
    CFace       *pFace;
    CShape      *pShape;
    ...
    virtual int getRegularPatches(CRectPatch*, int*, CTrglPatch*);
    virtual int getSingularPatches(CIntpPt*, int*, CAnglPatch*,
                                   int*, CRectPatch*, int*, CTrglPatch*);
    int getShape(int num, T2POINT *vTpt, double *vShp);
    virtual int getIntGeoData(T2POINT*, D3POINT*, D3NORMAL*, double*);
    ...
};
```

Boundary Face Method (4)

- Shape function class in C++

```
class CShape
{
public:
    CShapeFun *pShapeFun;
    double *dShp;    //pointing to the array in CShapeFun
    double *dShpDt1;//pointing to the array in CShapeFun
    double *dShpDt2;//pointing to the array in CShapeFun
    ...
    virtual int getIntpPtPointers(CLocation ***vIntpPt, double **vShape);
    virtual int getShape(T2POINT pt, double **vShape, CLocation ***vIntpPt,
                    int *nIntpPt){return 0;}
    virtual int getShapeDerivatives(T2POINT pt, double **vShpDt1,
                double **vShpDt2, CLocation ***vIntpPt, int *nIntpPt);
    ...
};
```

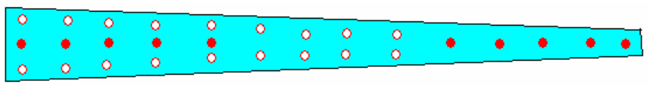
$$\Phi_I(\mathbf{s}) = \sum_{j=1}^{m} p_j(\mathbf{s})\left[A^{-1}(\mathbf{s})B(\mathbf{s})\right]_{jI} \qquad A(\mathbf{s}) = \sum_{I=1}^{N} w_I(\mathbf{s})\mathbf{p}(\mathbf{s}^I)\mathbf{p}^{\mathrm{T}}(\mathbf{s}^I)$$

$$\Phi_{I,k} = \sum_{j=1}^{m}[p_{j,k}(A^{-1}B)_{jI} + p_j(A^{-1}B_{,k} + A_{,k}^{-1}B)_{jI}]$$

$$B(\mathbf{s}) = \left[w_1(\mathbf{s})\mathbf{p}(\mathbf{s}^1), w_2(\mathbf{s})\mathbf{p}(\mathbf{s}^2), \cdots, w_N(\mathbf{s})\mathbf{p}(\mathbf{s}^N)\right]$$

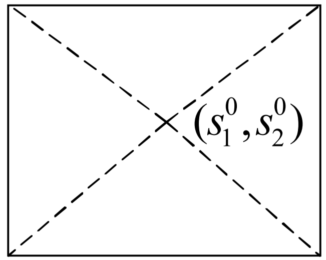
$$\tilde{u}(\mathbf{s}) = \sum_{j=1}^{k} l_{n,j}(s_1)\tilde{u}^j(s_2)$$

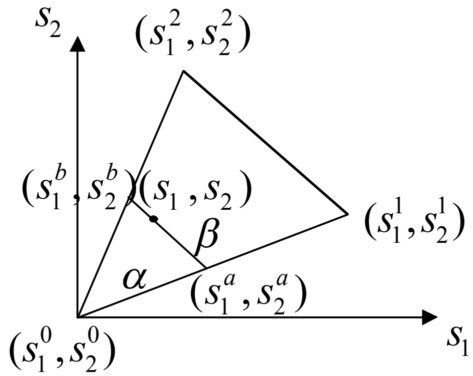
$$l_{n,j}(s_1) = \prod_{j\neq i}\left(\frac{(s_1 - s_1^j)}{(s_1^i - s_1^j)}\right) \qquad \tilde{u}^j(s_2) = \sum_{I=1}^{N_k}\phi_I(s_2)\hat{u}_I^k$$



Boundary Face Method (5)

- Weakly singular integration



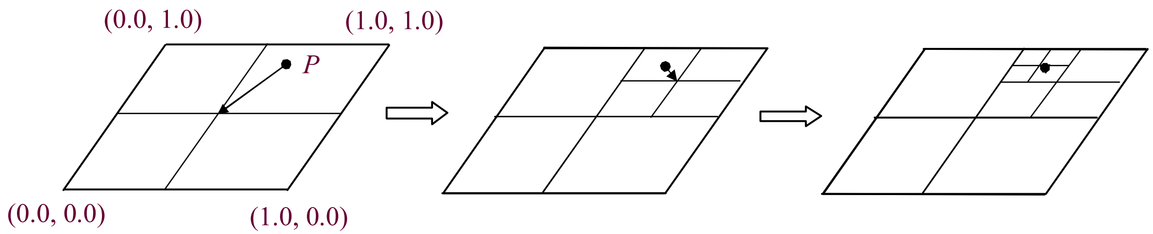


$$\begin{cases} s_1^a = s_1^0 + (s_1^1 - s_1^0)\alpha \\ s_2^a = s_2^0 + (s_2^1 - s_2^0)\alpha \end{cases} \qquad \begin{cases} s_1^b = s_1^0 + (s_1^2 - s_1^0)\alpha \\ s_2^b = s_2^0 + (s_2^2 - s_2^0)\alpha \end{cases} \qquad \begin{cases} t_1 = t_1^a + (t_1^b - t_1^a)\beta \\ t_2 = t_2^a + (t_2^b - t_2^a)\beta \end{cases}$$

$$I = \sum_{i=1}^{4} \int_0^1 \int_0^1 O(1/r) J_S(\mathbf{s}) J_L^{(i)}(\alpha)\, d\alpha\, d\beta \qquad J_L^{(i)} = \alpha S_\Delta$$

$$S_\Delta = \left| s_1^1 s_2^2 + s_1^2 s_2^0 + s_1^0 s_2^1 - s_1^2 s_2^1 - s_1^0 s_2^2 - s_1^1 s_2^0 \right|$$

- Nearly singular integration



Numerical results by BFM

- Governing equation

$$\nabla^2 u = 0$$

- Analytical solution

(1) Linear solution $u = x + y + z$

(2) Quadratic solution-1 $u = -2x^2 + y^2 + z^2$

(3) Quadratic solution-2 $u = xy + yz + zx$

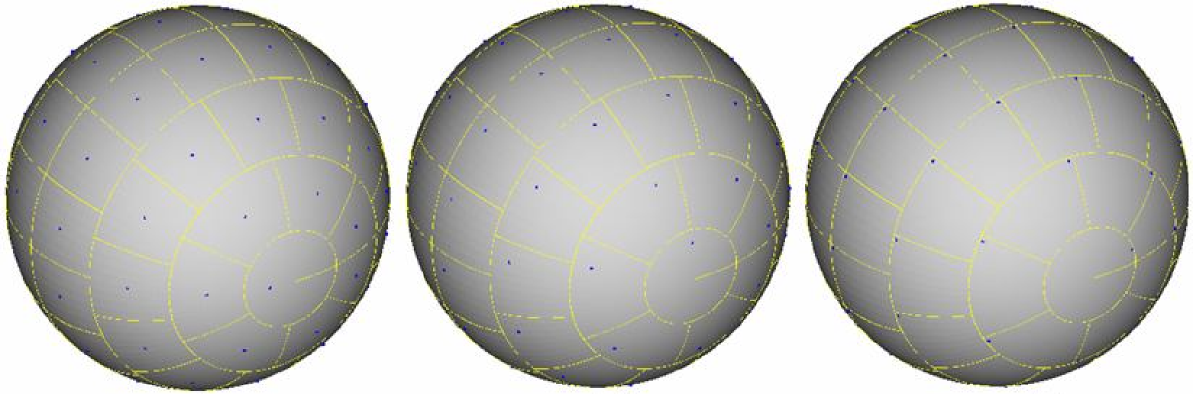
(4) Cubic solution $u = x^3 + y^3 + z^3 - 3yx^2 - 3xz^2 - 3zy^2$

- Error estimation

$$e = \frac{1}{|v|_{\max}} \sqrt{\frac{1}{N} \sum_{i=1}^{N} (v_i^{(e)} - v_i^{(n)})^2}$$

Numerical results by BFM (2)

- Sensitivity study of source location in a cell



$\alpha = 1$ $\alpha = 0.5$ $\alpha = 0.1$

α	u=linear %	u=quadratic-1 %	u=cubic %
0.05	0.420	1.219	0.937
0.2	0.262	0.572	1.000
0.4	0.173	0.375	0.748
0.6	0.145	0.292	0.485
0.8	0.127	0.210	0.313
1.0	0.135	0.281	0.243

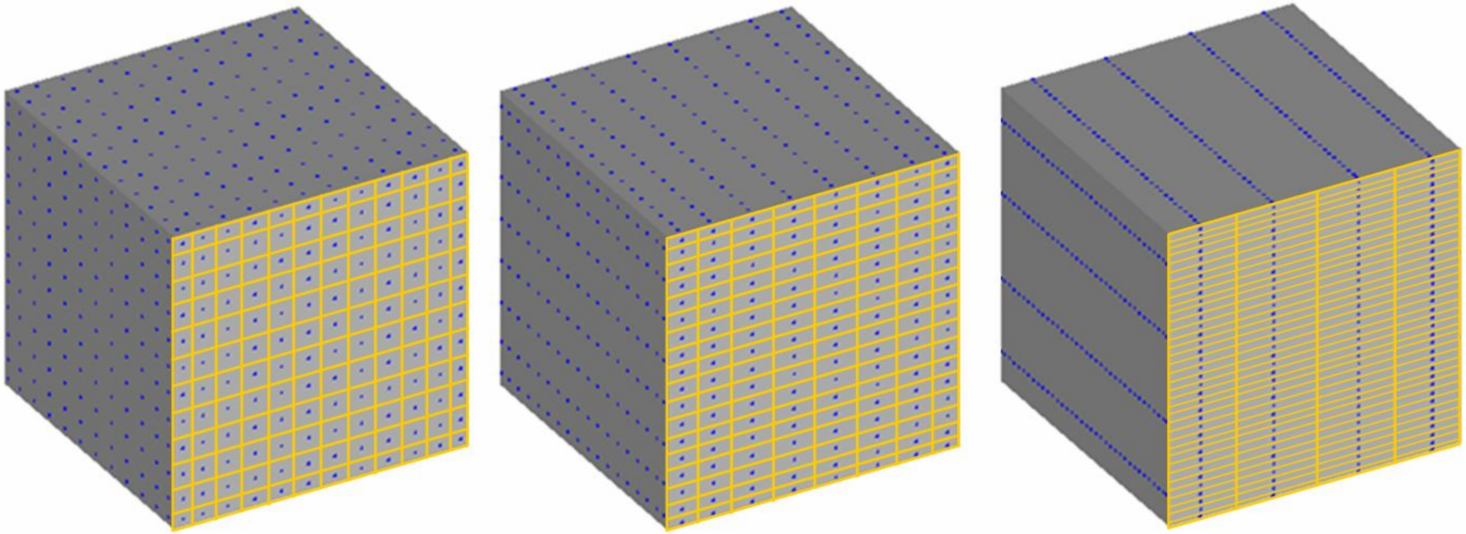
$s=t$	$u=$ linear	$u=$ quadratic-1	$u=$ cubic
0.05	63.469	94.647	346.47
0.1	25.833	17.562	53.187
0.15	2.443	3.615	9.241
0.2	1.864	2.695	7.734
0.25	2.009	4.403	6.798
0.3	1.129	1.947	3.074
1/3	**1.067**	**1.696**	**2.018**
0.35	3.765	4.121	4.469
0.4	1.198	3.006	5.199
0.45	1.952	4.191	9.852

BNM by MK. Chatiz and S. Mukherjee

Int. J. Numer. Meth. Engng. 2000; 47:1523-1547

Numerical results by BFM (3)

- Sensitivity to cell shape

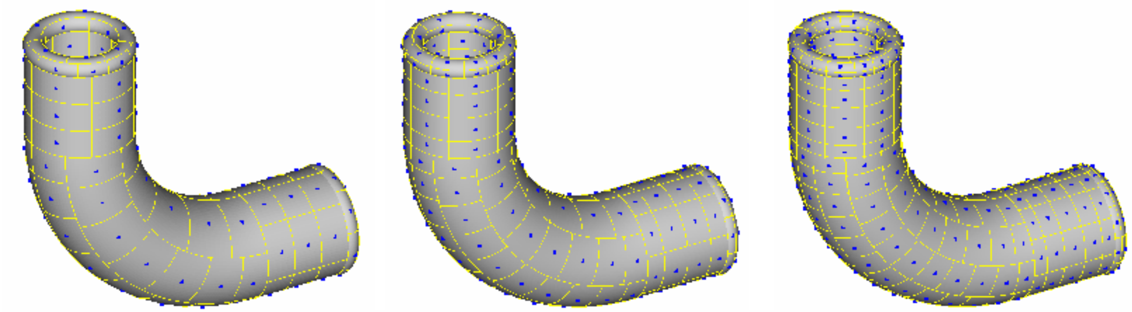


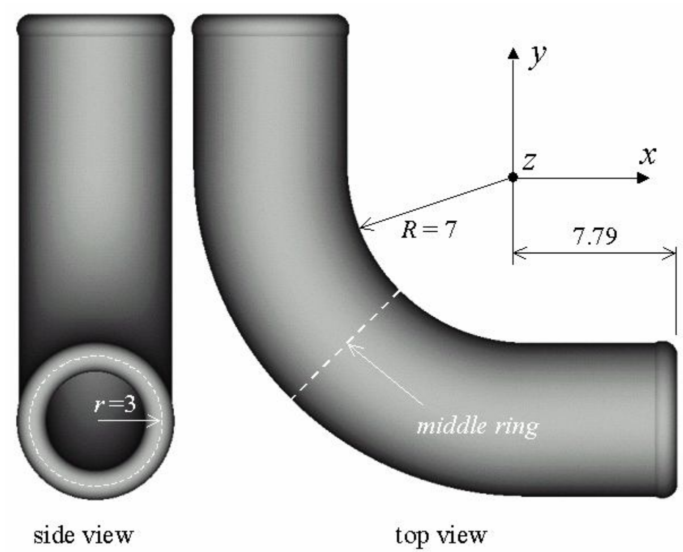
(a) 12×12 regular nodes (b) 8×18 irregular nodes (c) 4×36 irregular nodes

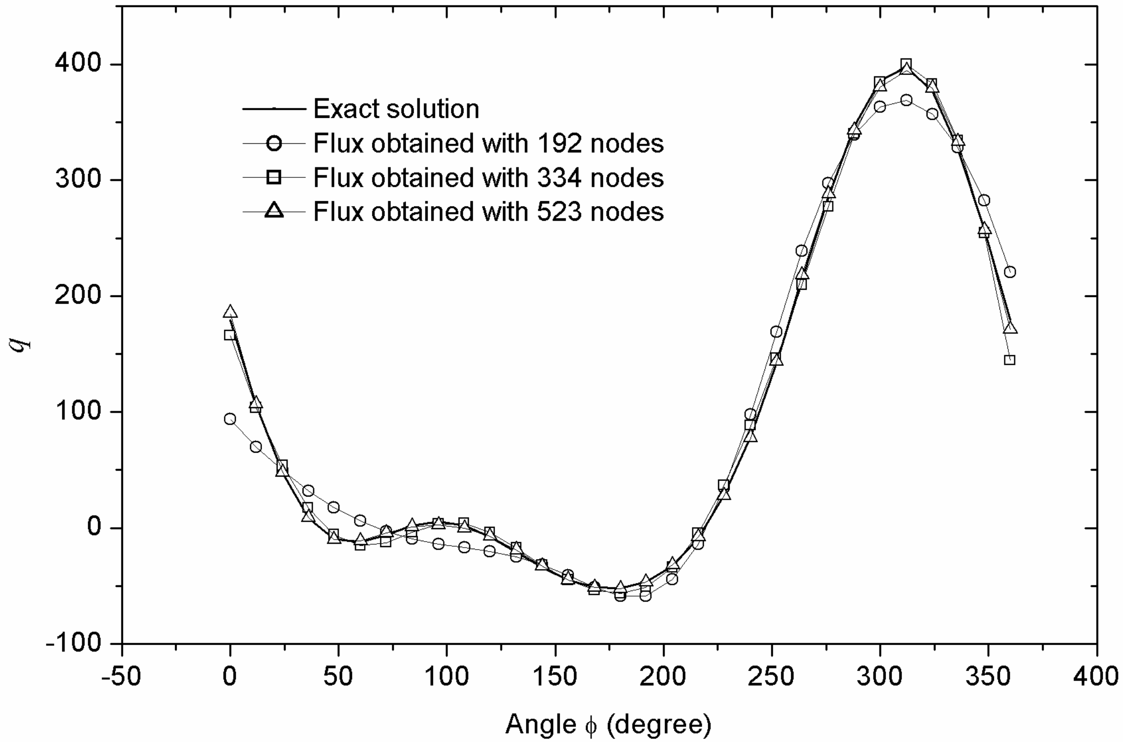
Node spacing	12x12	8x18	4x36
u=linear	0.04226	0.091	0.05908
u=quadratic-1	0.01355	0.03865	0.02179
u=quadratic-2	0.03552	0.06985	1.091
u=cubic	0.02694	0.1203	1.638

Numerical results by BFM (5)

- Convergence study on a elbow pipe

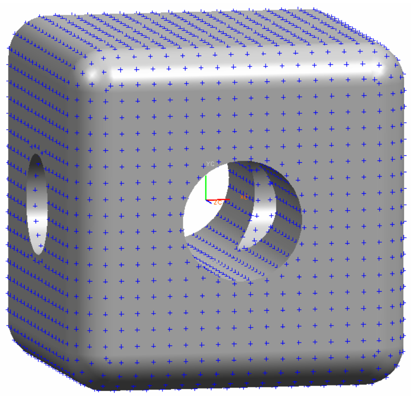


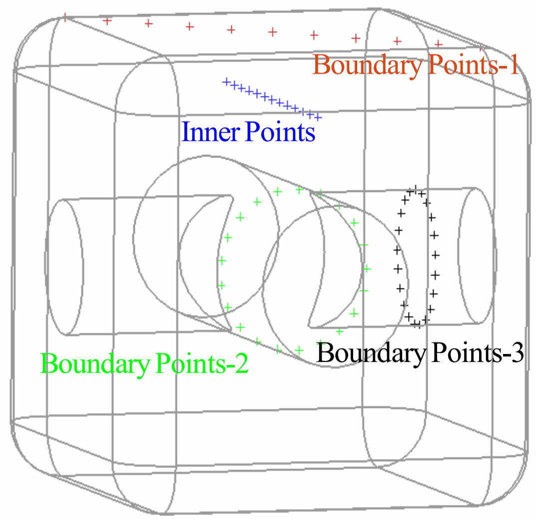


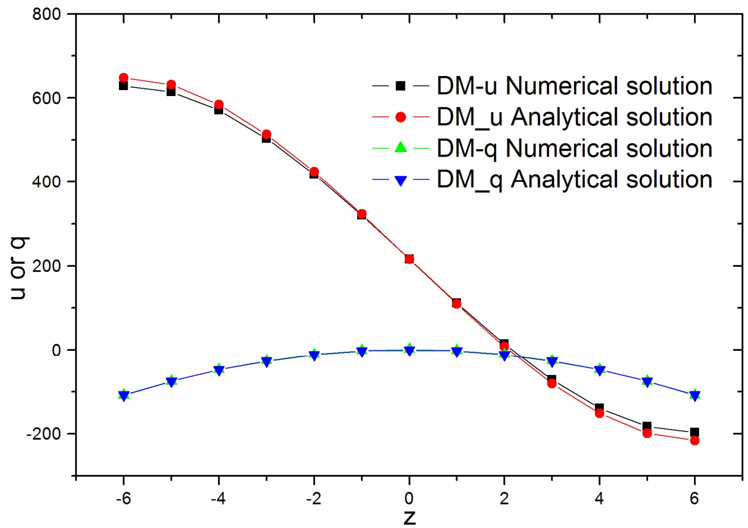


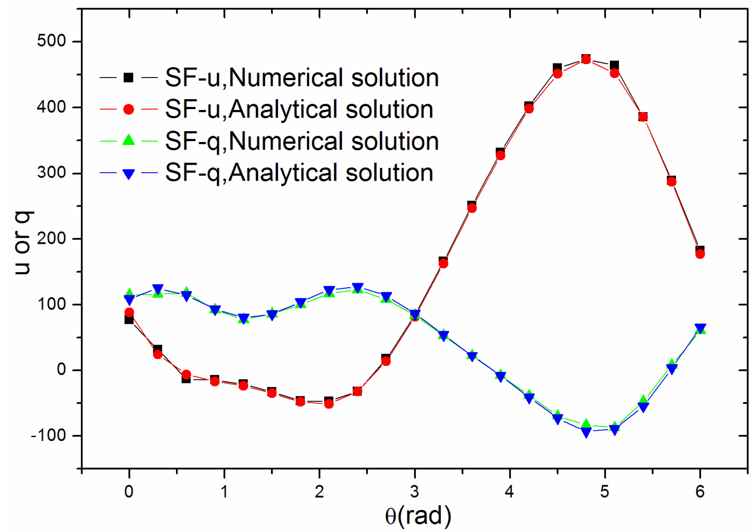
Numerical results by BFM (6)

- Body with trimmed surfaces

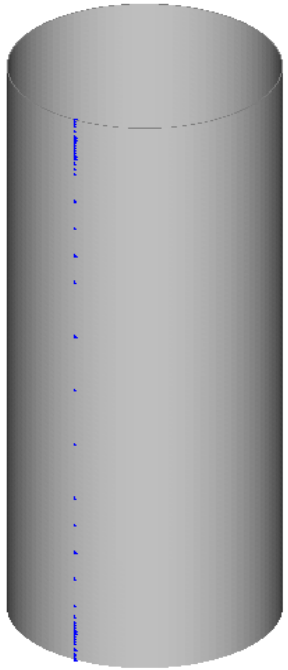


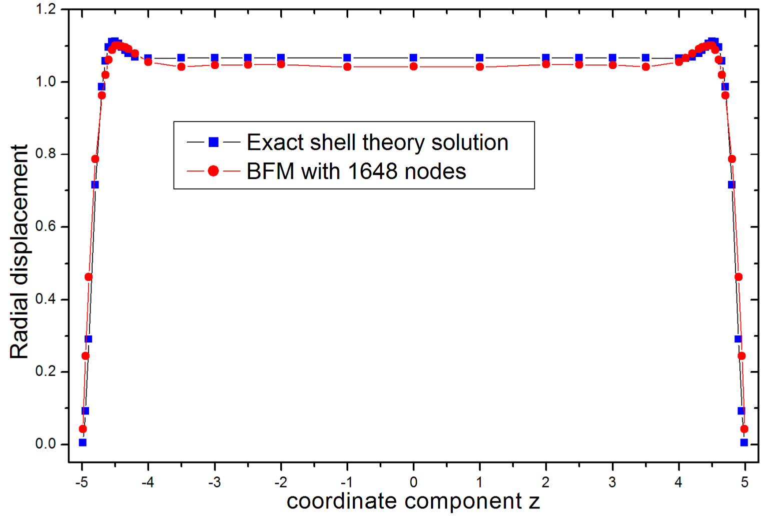


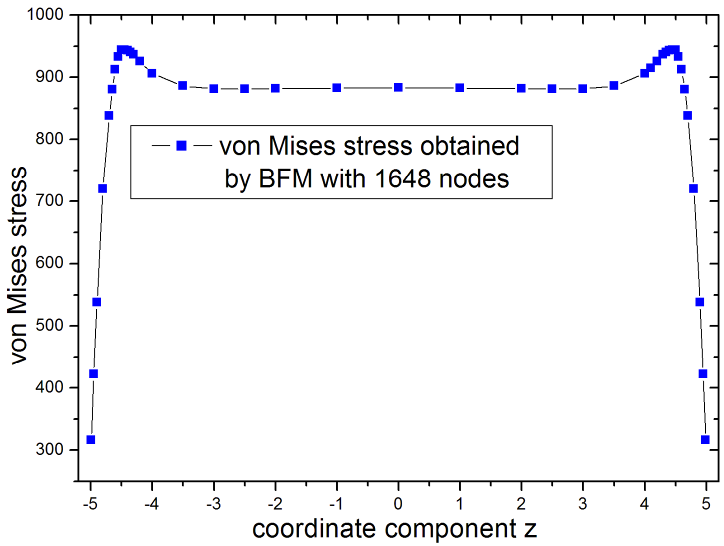




Numerical results by BFM (7)

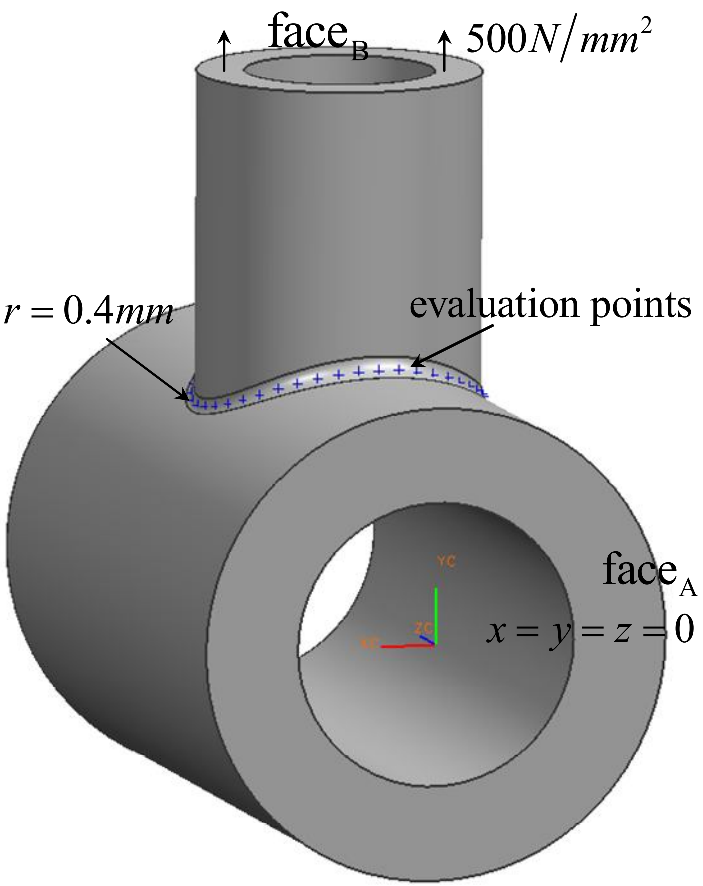


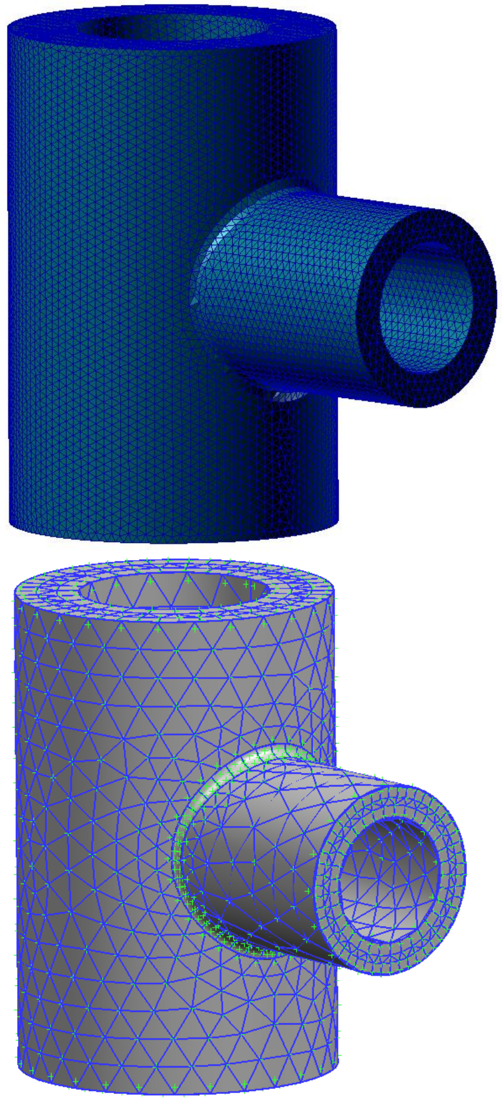




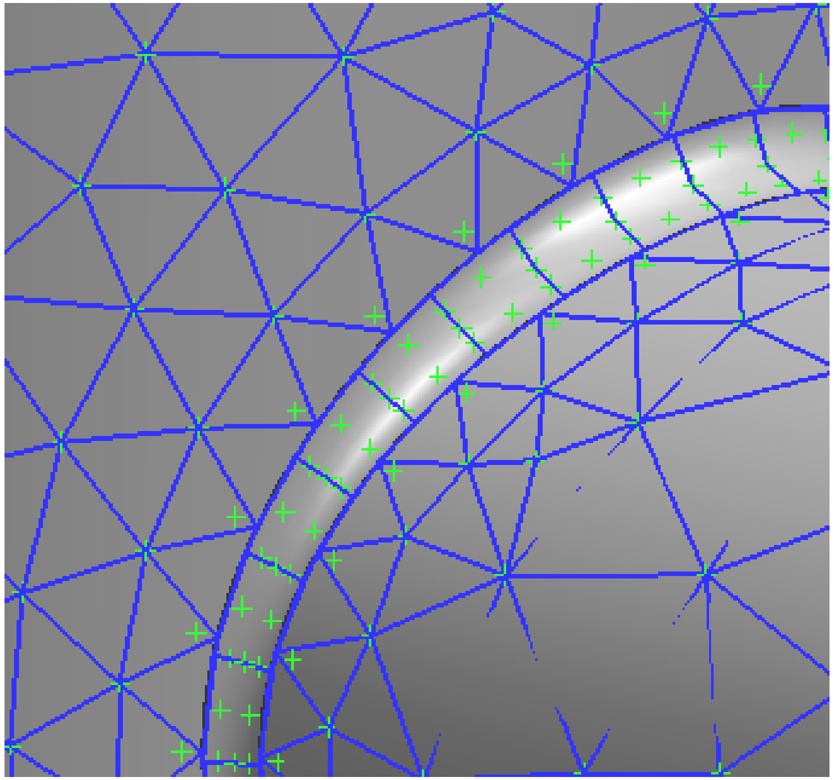
中性面半径：R=2.0
圆柱厚度：t=0.02
圆柱高度：L=10.0
L与t比值：500

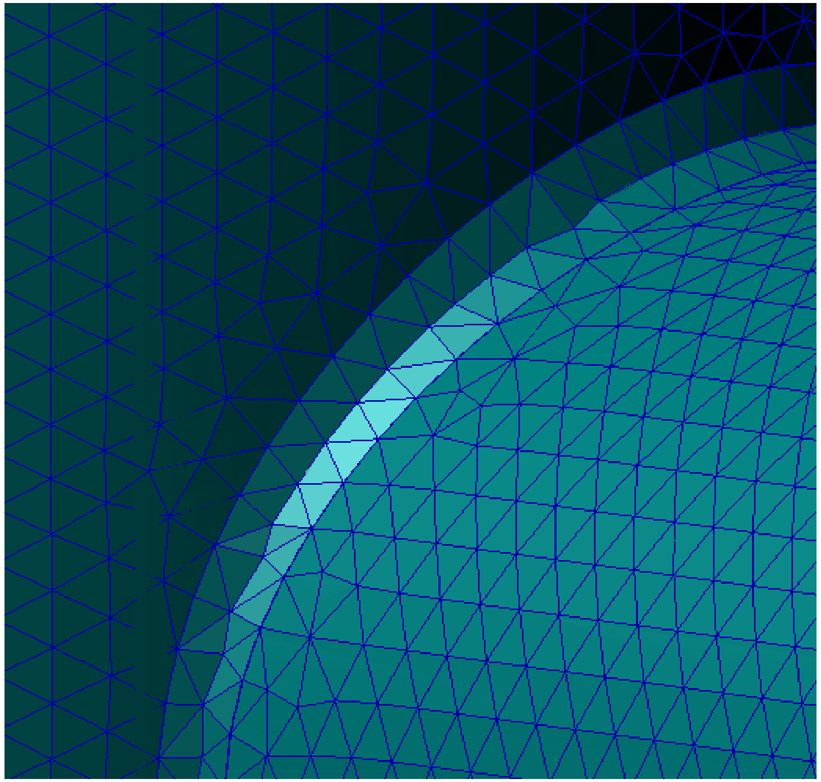
Numerical results by BFM (8)





Numerical results by BFM (9)





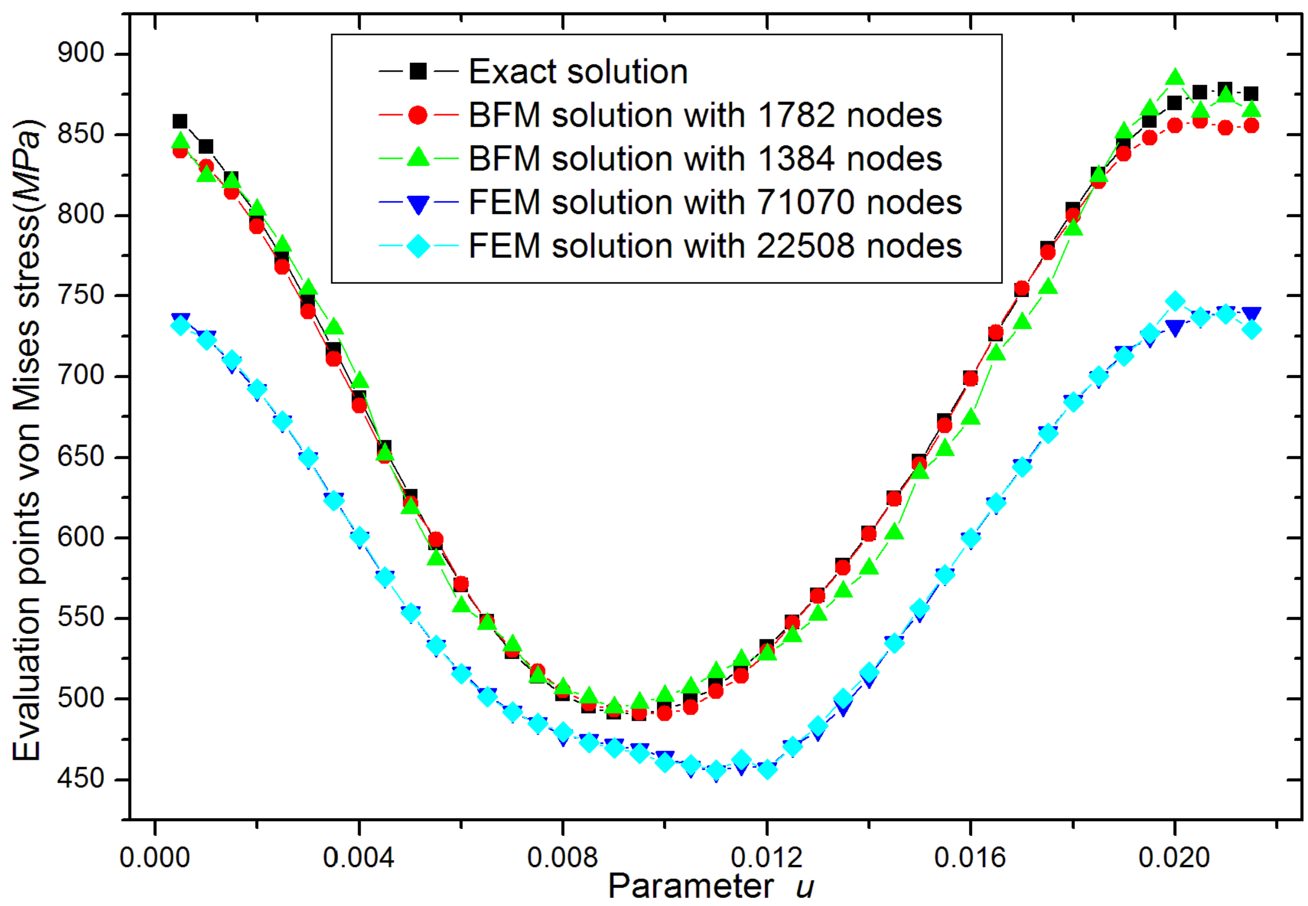
Numerical results by BFM (10)

位移精确解为边界条件：

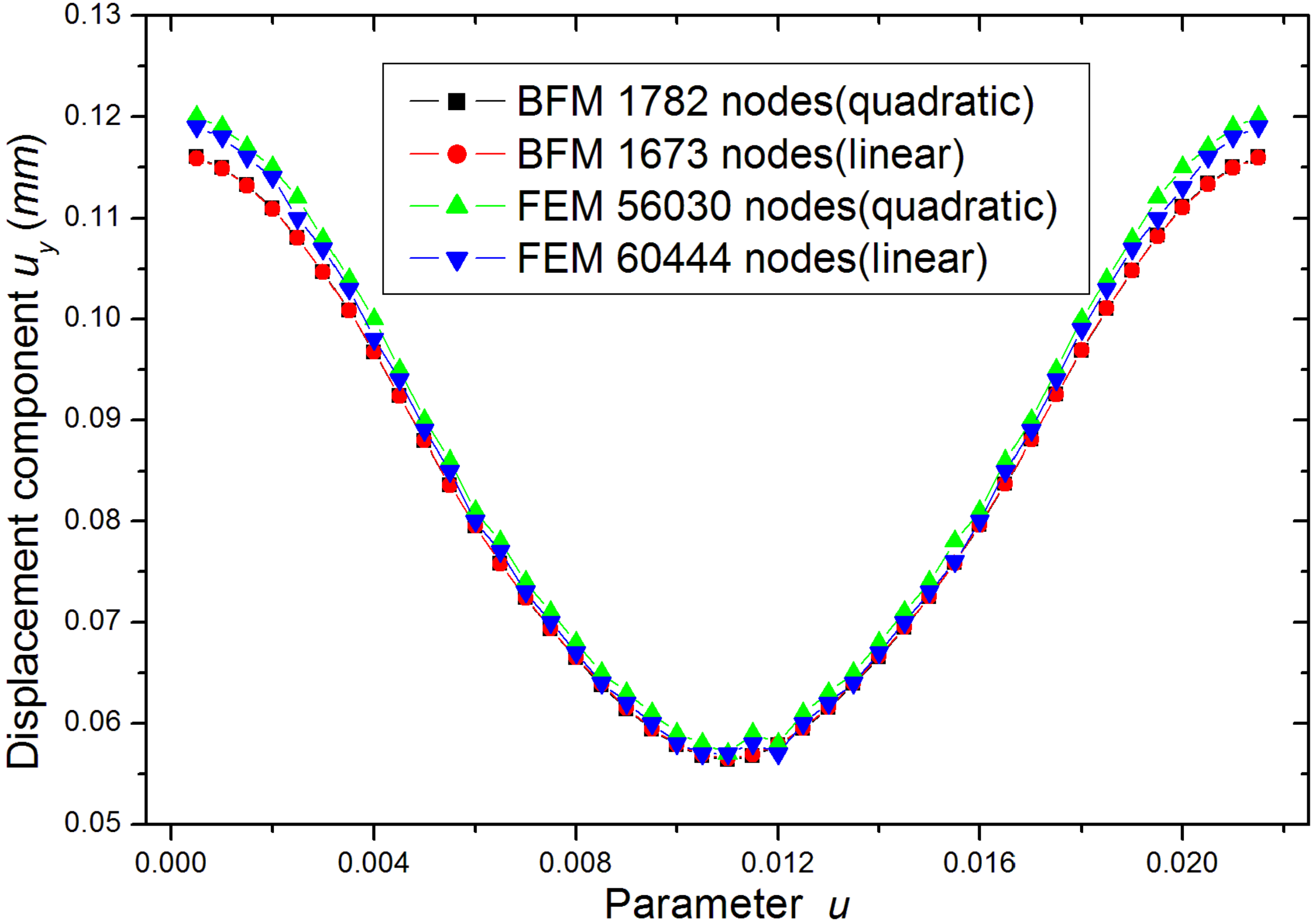
$$\begin{cases} u_x = -2x^2 + 3y^2 + 3z^2 \\ u_y = 3x^2 - 2y^2 + 3z^2 \\ u_z = 3x^2 + 3y^2 - 2z^2 \end{cases}$$

杨氏模量10Mpa,
泊松比0.25，
应力精确解为：

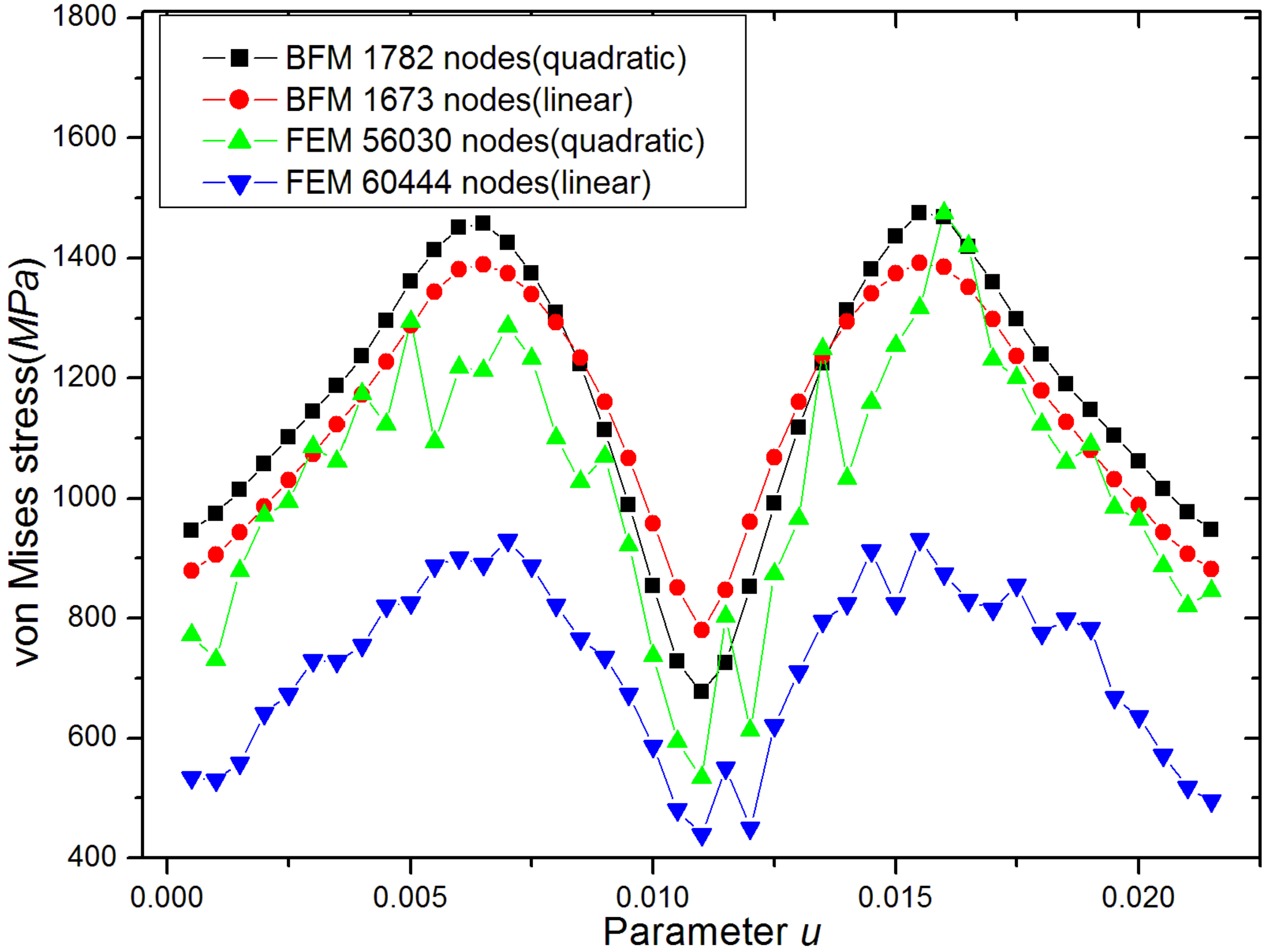
$$\begin{cases} \sigma_{xx} = -16(3x + y + z) \\ \sigma_{yy} = -16(x + 3y + z) \\ \sigma_{zz} = -16(x + y + 3z) \\ \sigma_{xy} = 2.4(x + y) \\ \sigma_{xz} = 2.4(x + z) \\ \sigma_{yz} = 2.4(y + z) \end{cases}$$



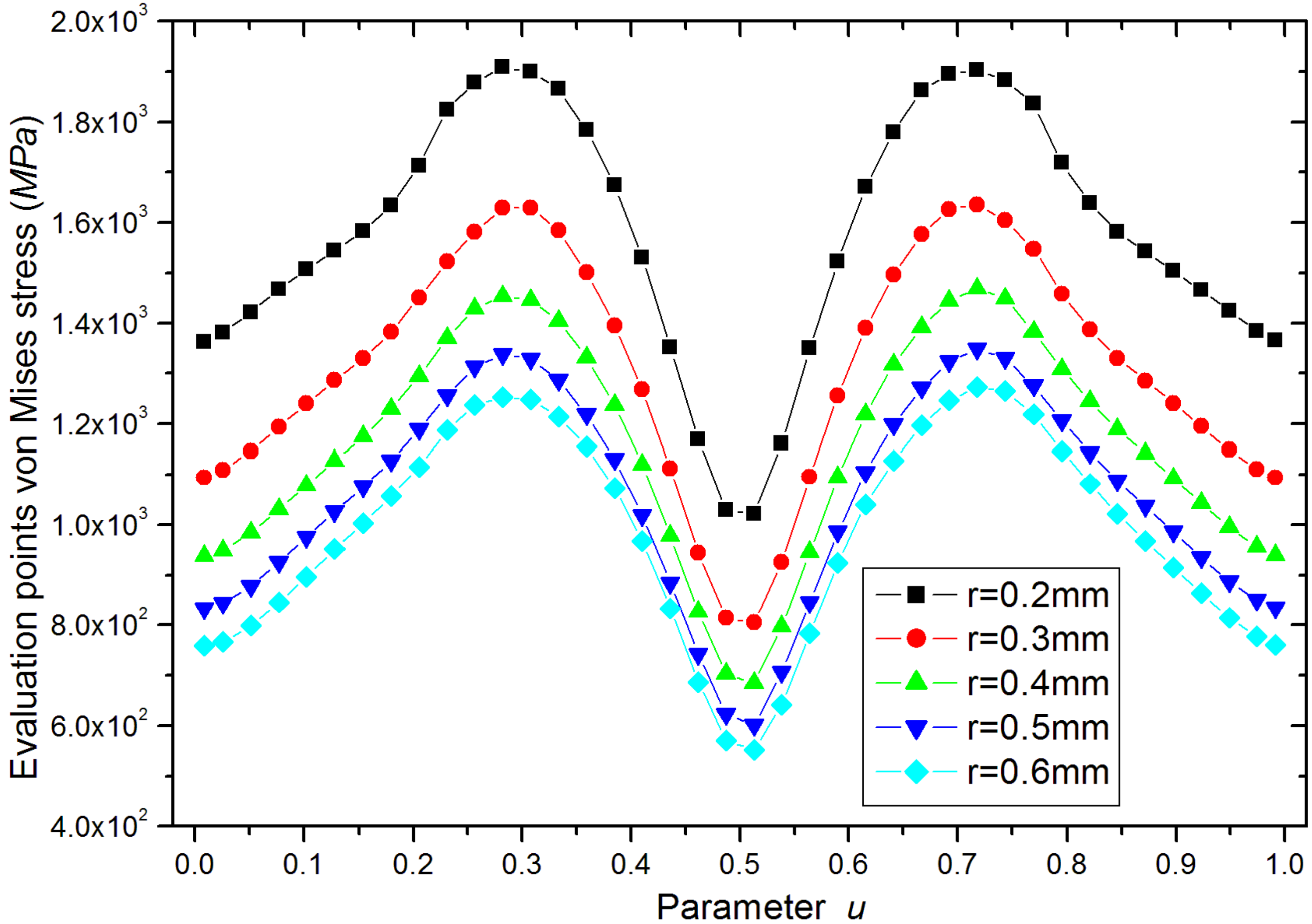
Numerical results by BFM (11)



Numerical results by BFM (12)



Numerical results by BFM (13)

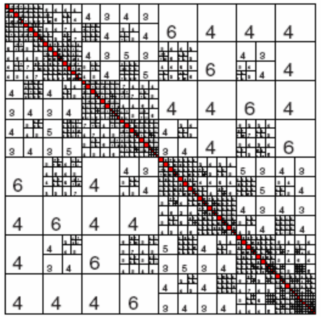


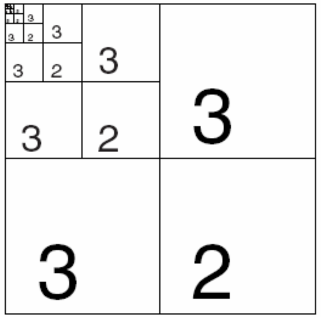
Hierarchical matrix and ACA

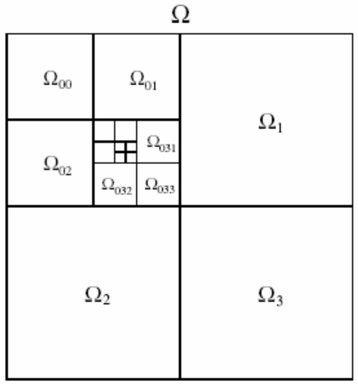
- Why **Hierarchical matrix and ACA, not FMM？**
  - Although the FMM possesses better asymptotic complexity (linear order). It depends on a priori knowledge of the kernel function, which is to be expanded by spherical harmonic series.
  - The Hierarchical Matrix and ACA are kernel independent.
  - The Hierarchical Matrix and ACA are a purely algebraic algorithms, namely the computational speed-up is achieved through linear algebra manipulations of the matrix, e.g. QR decomposition, SVD, LU decomposition, etc.
  - The Hierarchical Matrix and ACA can provide a direct solver for large scale computations, or an efficient preconditioner for an iterative solver as well, e.g. GMRES, CGN, etc.

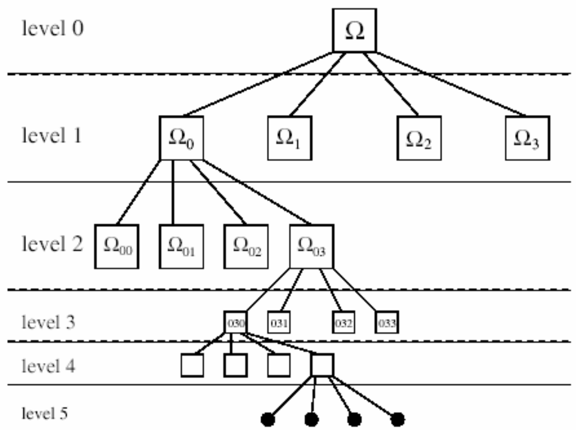
Hierarchical matrix and ACA

- **Hierarchical matrix ($\mathcal{H}$-matrix)**
  - **Block partitioning with tree**

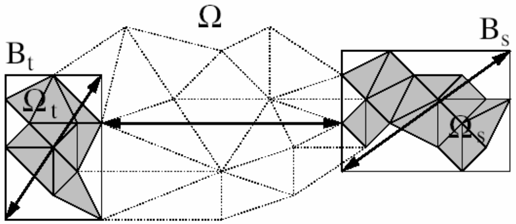


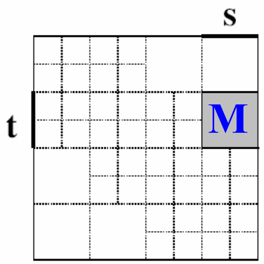


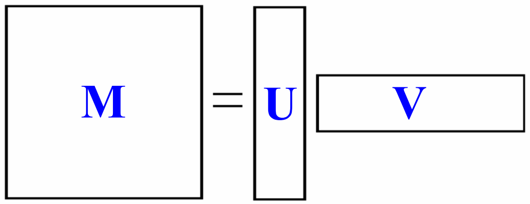




  - **Low rank approximation**







  - **Arithmetic operations and complexity**

Storage	Addition	Matrix-vector multiplication	Matrix-matrix multiplication	Matrix-inversion	LU-Decomposition
$O(n\log n)$	$O(n\log n)$	$O(n\log n)$	$O(n\log^2 n)$	$O(n\log^2 n)$	$O(n\log^2 n)$

Hierarchical matrix and ACA

- **Singular Value Decomposition (SVD)**

$$\underset{M\times N}{\begin{pmatrix} \\ \mathrm{A} \\ \\ \end{pmatrix}} = \underset{M\times N}{\begin{pmatrix} \\ \mathrm{U} \\ \\ \end{pmatrix}} \cdot \begin{pmatrix} w_1 & & \\ & \ddots & \\ & & w_N \end{pmatrix} \cdot \underset{N\times N}{\begin{pmatrix} \mathrm{V} \end{pmatrix}} \qquad M > N$$

  - If $w_i=0$ for $i>k$ (rank A=k), then

$$\underset{M\times N}{\begin{pmatrix} \\ \mathrm{A} \\ \\ \end{pmatrix}} = \underset{M\times k}{\begin{pmatrix} \\ \mathrm{U} \\ \\ \end{pmatrix}} \cdot \underset{k\times N}{\begin{pmatrix} \mathrm{V} \end{pmatrix}} \qquad k<<N$$

$$\begin{pmatrix} \\ \mathrm{A} \\ \\ \end{pmatrix} \cdot \begin{pmatrix} x \end{pmatrix} = \underbrace{\begin{pmatrix} \\ \mathrm{U} \\ \\ \end{pmatrix} \cdot \underbrace{\begin{pmatrix} \mathrm{V} \end{pmatrix} \cdot \begin{pmatrix} x \end{pmatrix}}_{kN}}_{kM}$$

Both memory requirement and operations are reduced to $k(M+N)$ from $M\times N$!!!

Hierarchical matrix and ACA

- **Low rank approximation**
  - **Definition (ε - rank).** The $\varepsilon-$rank of a matrix $A \in \mathbb{C}^{M\times N}$ with respect to the matrix norm $\|\cdot\|$ is defined as

$$\operatorname{rank}_{\varepsilon} A = \min\left\{\operatorname{rank} T : \|A-T\| < \varepsilon\right\}$$

  - **Theorem.** The best approximation of at most rank k to A is

$$\|A-A_k\| = \min\left\{\|A-T\| : \operatorname{rank} T \le k\right\}$$

where $A_k = \sum_{i=1}^{k} w_i u_i v_i^*$ and $(w_i, u_i, v_i), i = 1,\cdots,k$ are the k largest singular triplets.

$$\begin{pmatrix} A_k \end{pmatrix} = \begin{pmatrix} U \end{pmatrix} \cdot \begin{pmatrix} w_1 & & & & \\ & \ddots & & & \\ & & w_k & & \\ & & & w_{k+1}=0 & \\ & & & & \ddots \\ & & & & & w_N = 0 \end{pmatrix} \cdot \begin{pmatrix} V \end{pmatrix}$$

Hierarchical matrix and ACA

- **Early version of $\mathcal{H}$-matrix**

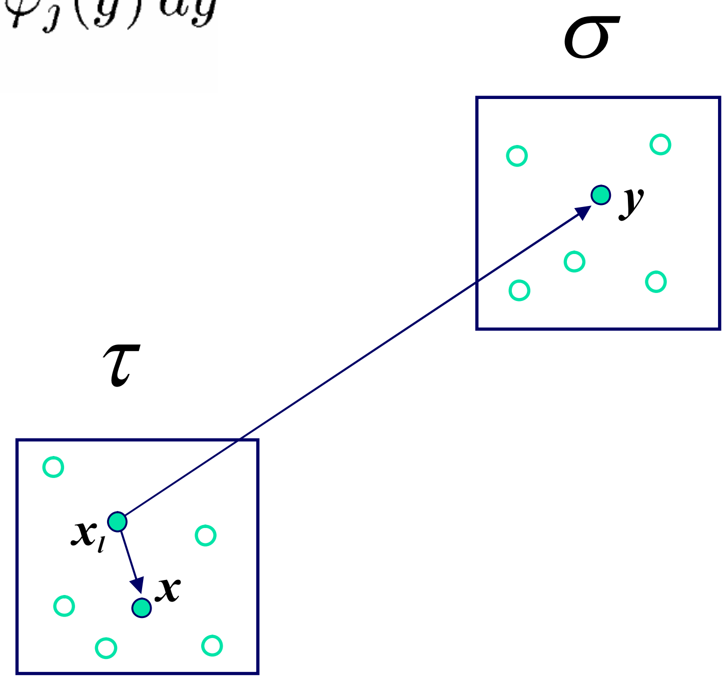
$$L_{ij} = \sum_{\iota=1}^{k} \int_{\Omega} p_{\iota}^{\tau}(x)\varphi_i(x)\,dx \int_{\Omega} g(x_{\iota}^{\tau}, y)\varphi_j(y)\,dy$$

$$\tilde{g}(x,y) := \sum_{\iota=1}^{k} p_{\iota}^{\tau}(x) g(x_{\iota}^{\tau}, y)$$

$$A_{i\iota} := \int_{\Omega} p_{\iota}^{\tau}(x)\varphi_i(x)\,dx$$

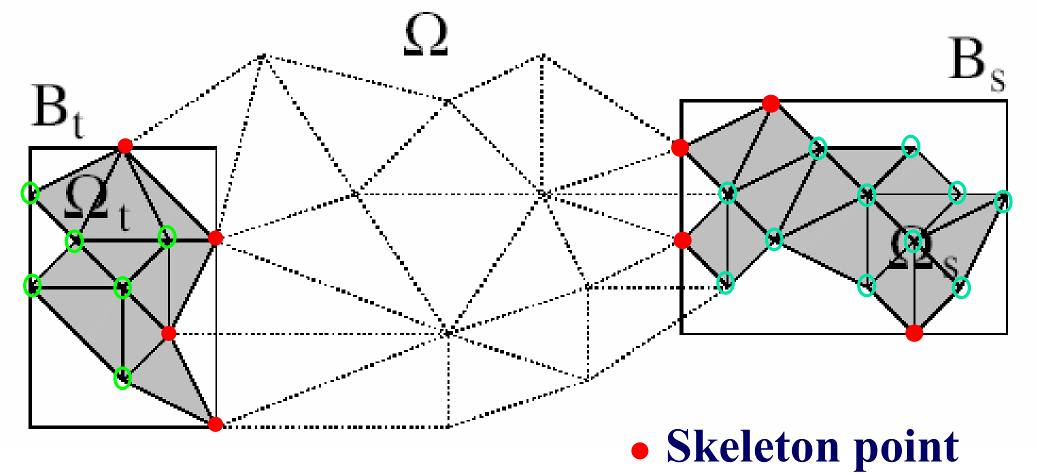
$$B_{j\iota} := \int_{\Omega} g(x_{\iota}^{\tau}, y)\varphi_j(y)\,dy$$

$$L\big|_{\tau\times\sigma} \approx AB^{T}$$



Hierarchical matrix and ACA

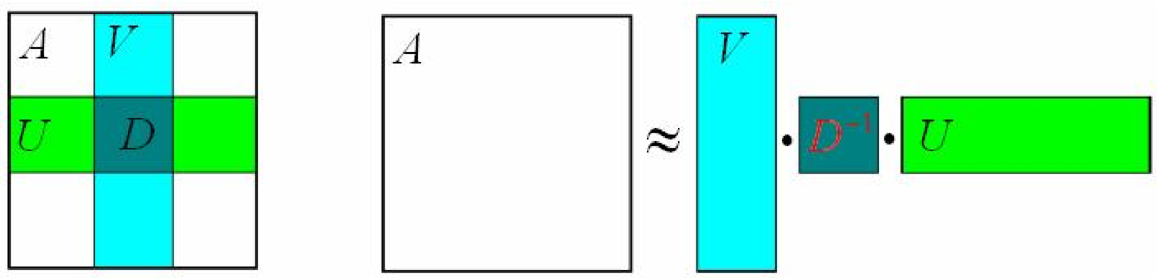
- **Adaptive Cross Approximation (ACA)**



$$a_{ij} = \int_{\Gamma} \kappa(x, y_i)\varphi_j(x)dS_x$$

$$\kappa(x,y) = -\frac{1}{4\pi}\frac{(x-y, n_x)}{|x-y|^2}$$

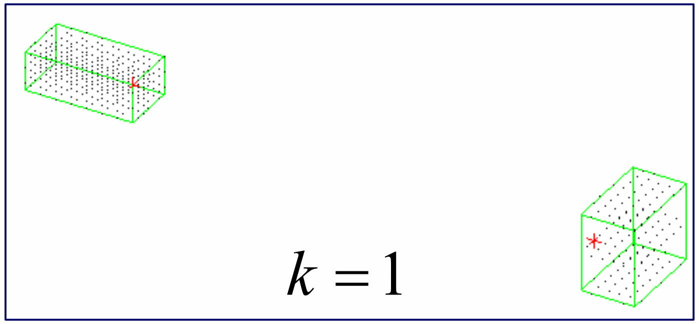
$$i \in t, \quad j \in s$$

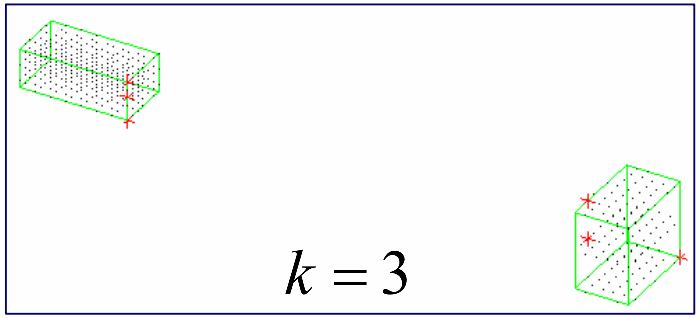


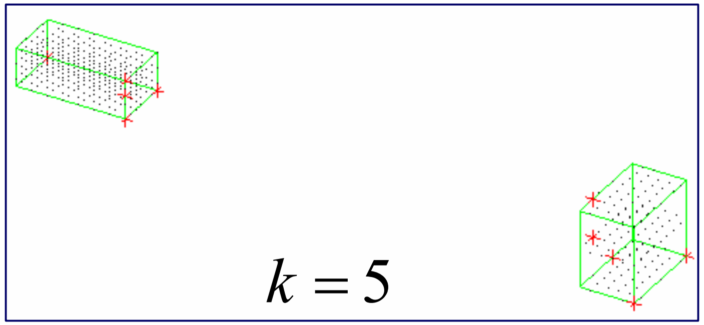
By M. Bebendorf
Numer. Math.
Vol. 86 (2000), pp. 565-589

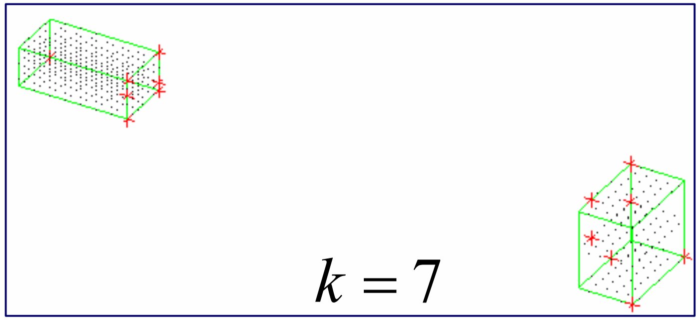
Hierarchical matrix and ACA

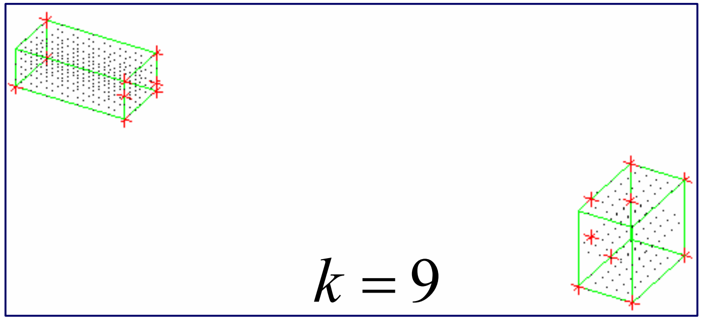
- **ACA without iteration**







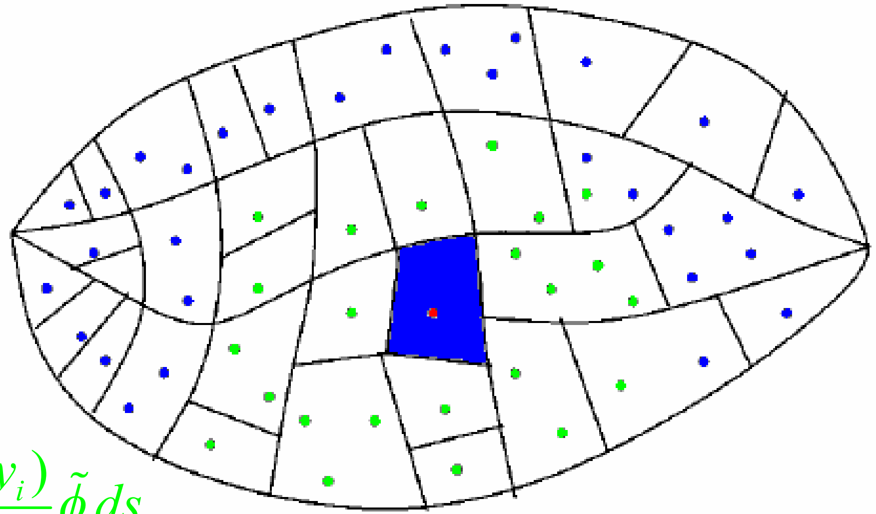




Determine the skeleton points using geometric method. The predetermination of skeleton points is particularly helpful when evaluating the entries by boundary integration.

Hierarchical matrix and ACA

- Shape functions other than constant element

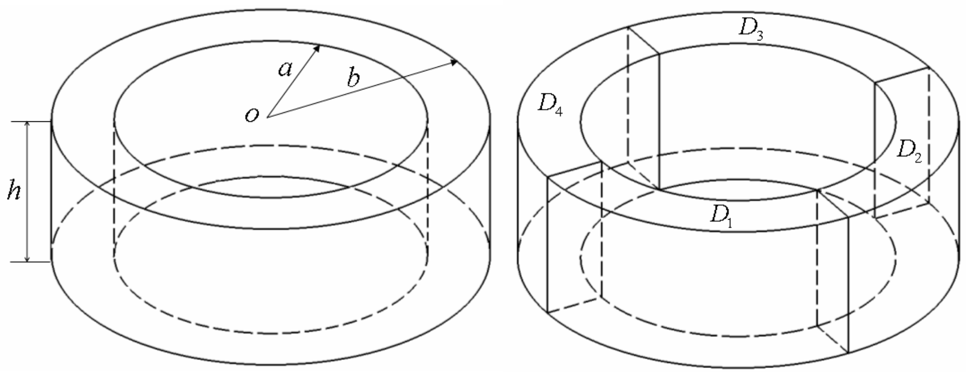


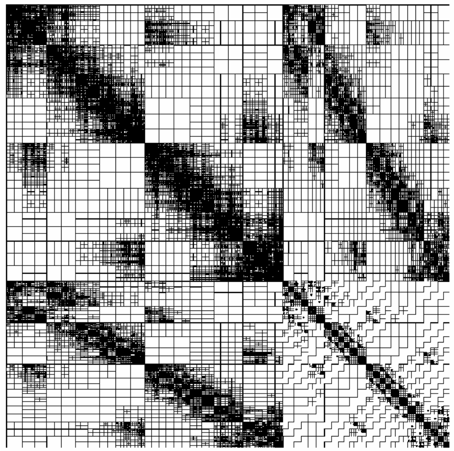
$$\int_{S_i} \frac{\partial \phi^s(s,y_i)}{\partial n(s)} \sum_{k=1}^{N} \Phi'_k(s)\tilde{\phi}_k ds - \int_{S_i} \frac{\partial \phi^s(s,y_i)}{\partial n(s)} \tilde{\phi}_i ds$$
$$+\sum_{j\neq i}^{near} \int_{S_j} \frac{\partial \phi^s(s,y_i)}{\partial n(s)} \sum_{k=1}^{N} \Phi'_k(s)\tilde{\phi}_k ds - \sum_{j\neq i}^{near} \int_{S_j} \frac{\partial \phi^s(s,y_i)}{\partial n(s)} \tilde{\phi}_i ds$$
$$-\sum_{j}^{near} \int_{S_j} \phi^s(s,y_i) \sum_{k=1}^{N} \Phi'_k(s)\tilde{q}_k ds$$
$$+\sum_{j}^{far} \frac{\partial \phi^s(s,y_i)}{\partial n(s)} \tilde{\phi}_j A_{S_j} - \sum_{j}^{far} \frac{\partial \phi^s(s,y_i)}{\partial n(s)} \tilde{\phi}_i A_{S_j} - \sum_{j}^{far} \phi^s(s,y_i)\tilde{q}_j A_{S_j}$$
$$= 0$$

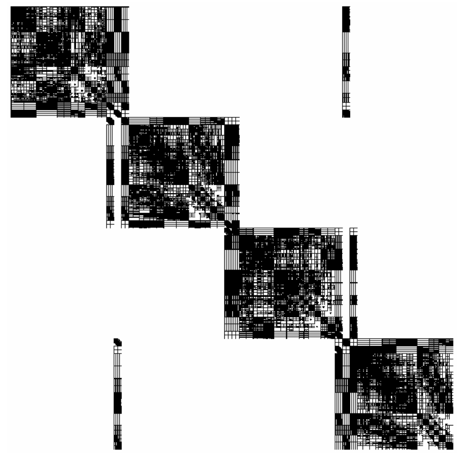
The near part and far part may be overlapped each other!

Numerical results

- Comparison with FMM







Multi-domain model			
DOFs	T_{coef} (s)	T_{equ} (s)	err_{ϕ}
11888	711	1167	5.1×10^{-4}
47448	4127	6418	1.6×10^{-4}
106688	7753	12516	9.9×10^{-5}
Single-domain model			
DOFs	T_{coef} (s)	T_{equ} (s)	err_{ϕ}
10288	530	803	5.7×10^{-4}
41048	3906	5203	1.6×10^{-4}
92288	8065	8937	9.5×10^{-5}

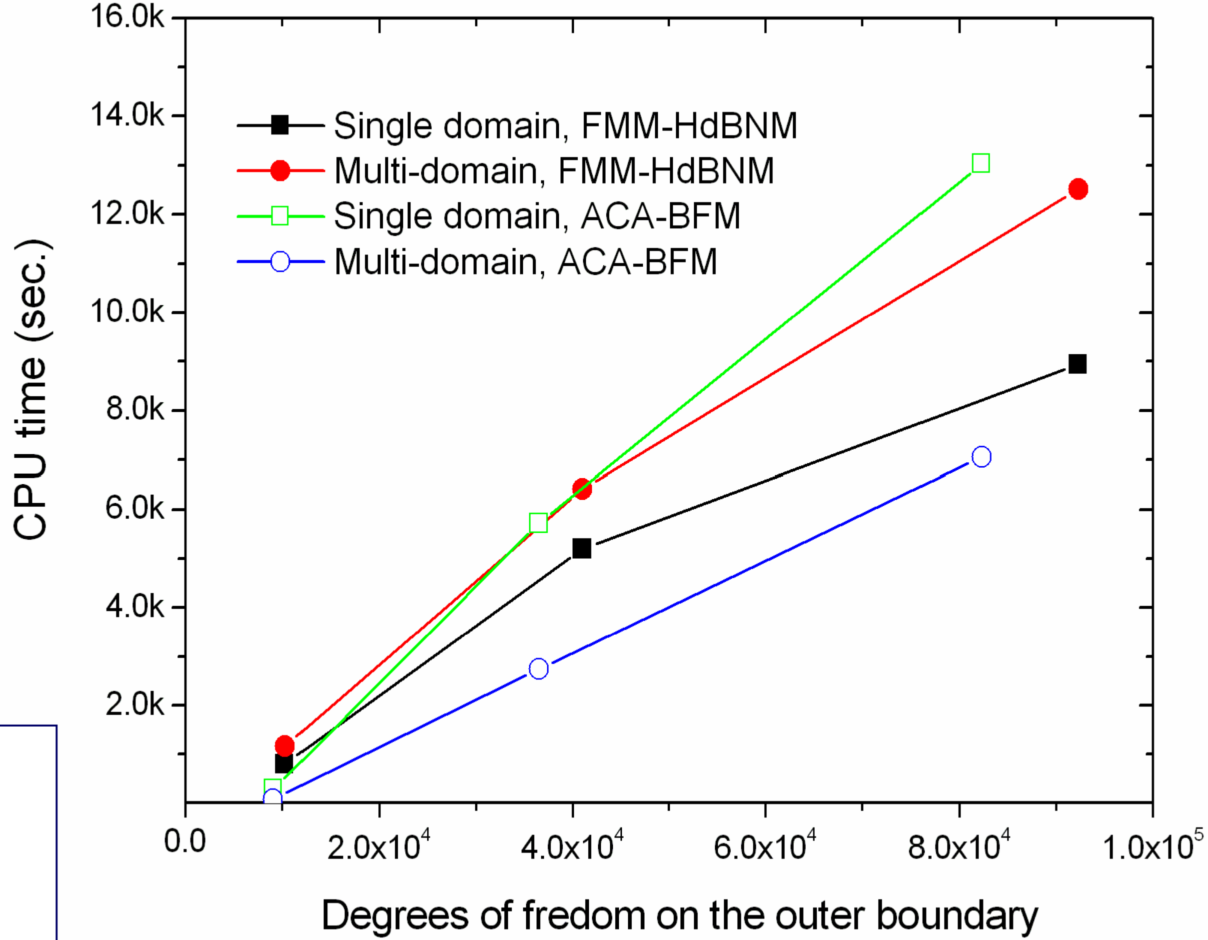
By HdBNM-FMM

Multi-domain model			
DOFs	T_{coef} (s)	T_{equ} (s)	err_{ϕ}
10720	249	88	7.9×10^{-5}
42944	2101	2744	7.0×10^{-5}
98682	4731	5357	9.9×10^{-6}
Single-domain model			
DOFs	T_{coef} (s)	T_{equ} (s)	err_{ϕ}
9120	407	295	5.2×10^{-5}
36544	3307	5720	5.1×10^{-5}
82272	7017	13035	4.5×10^{-5}

By BFM-ACA

Numerical results

- CPU seconds for solving the system equation

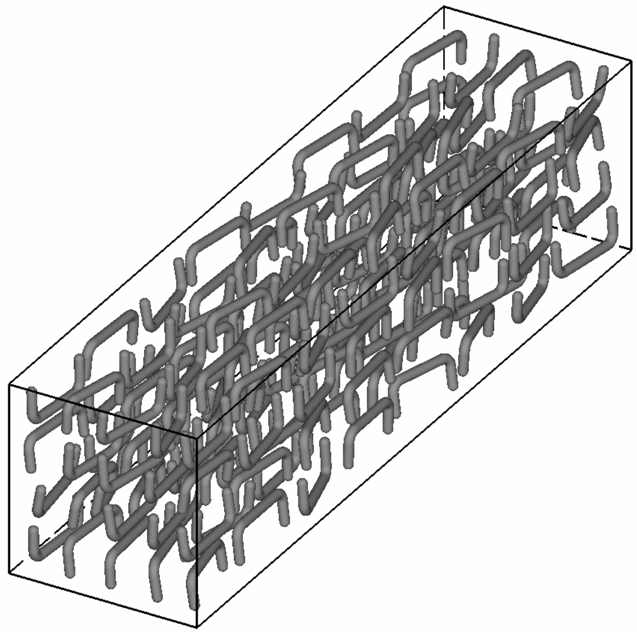


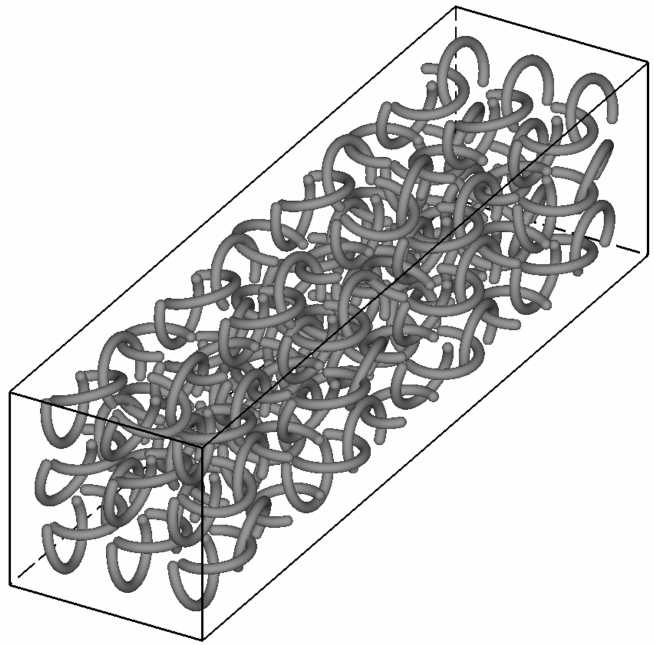
Timing results for the thick cylinder problem

The hierarchical LU-decomposition will be considerably beneficial to equations that have multiple right hand sides

Numerical results

- CNT composite simulation



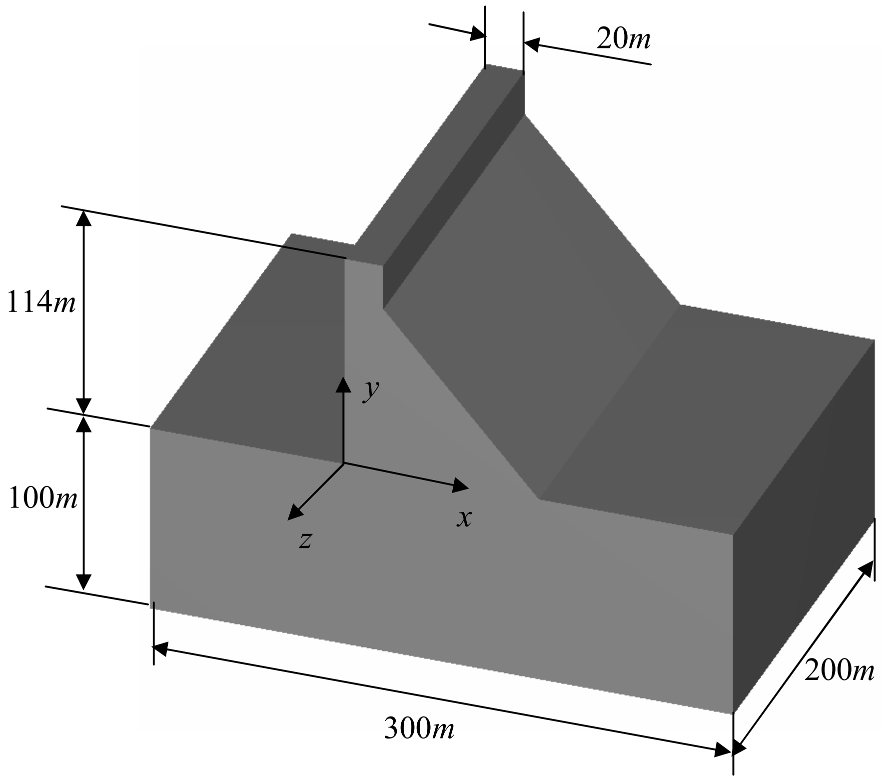


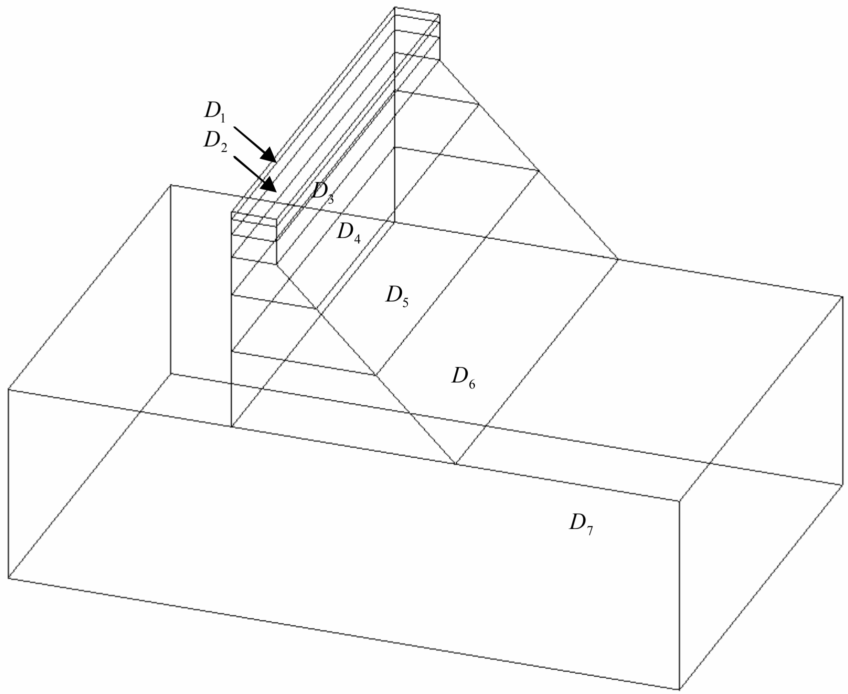
	κ	Nodes	Time (s)
HdBNM-FMM	1.337	165153	9776
BFM-ACA	1.353	165153	11945

	κ	Nodes	Time (s)
HdBNM-FMM	0.919	109314	5396
BFM-ACA	0.954	109314	6127

Thermal analysis of concrete dam

➢ **Geometric model**



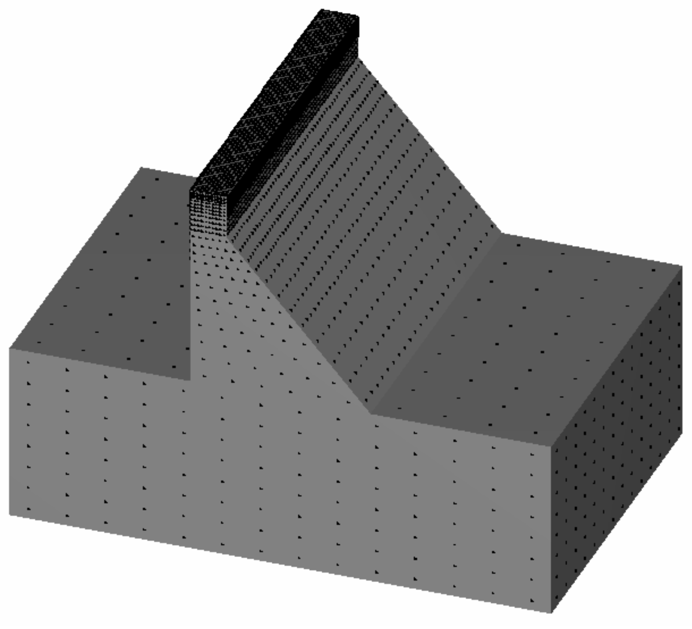


Thermal analysis of concrete dam

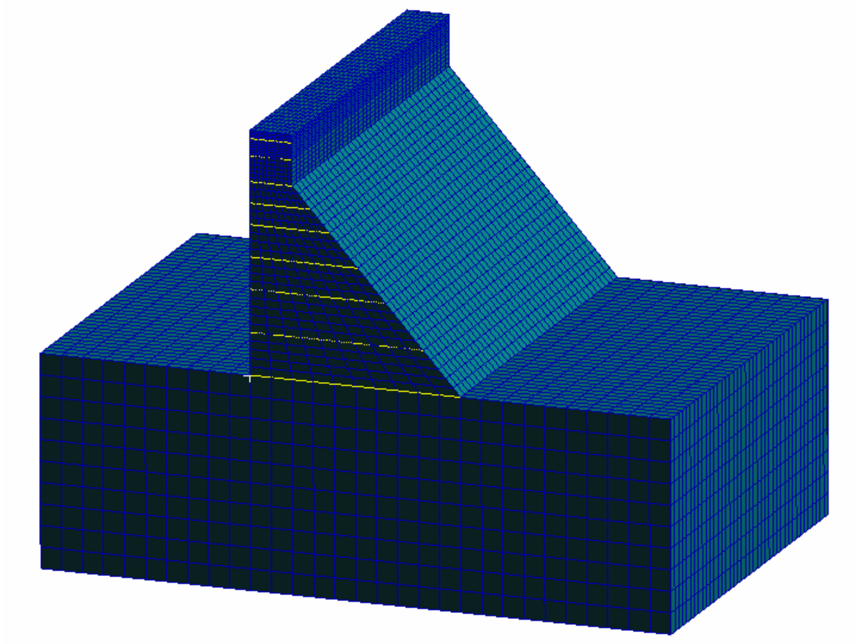
- **Discrete models**

27100, 15381, 8492 boundary Nodes used for BFM analysis

112000、56000、29201 elements used for MSC/NASTRAN



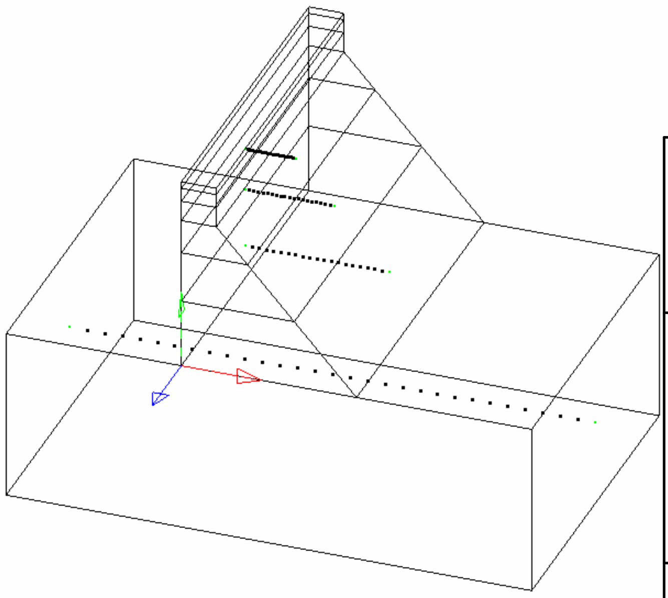
8492 boundary Nodes



29201 elements (FEM)

Thermal analysis of concrete dam

- **Homogeneous Case**



Evaluation Points

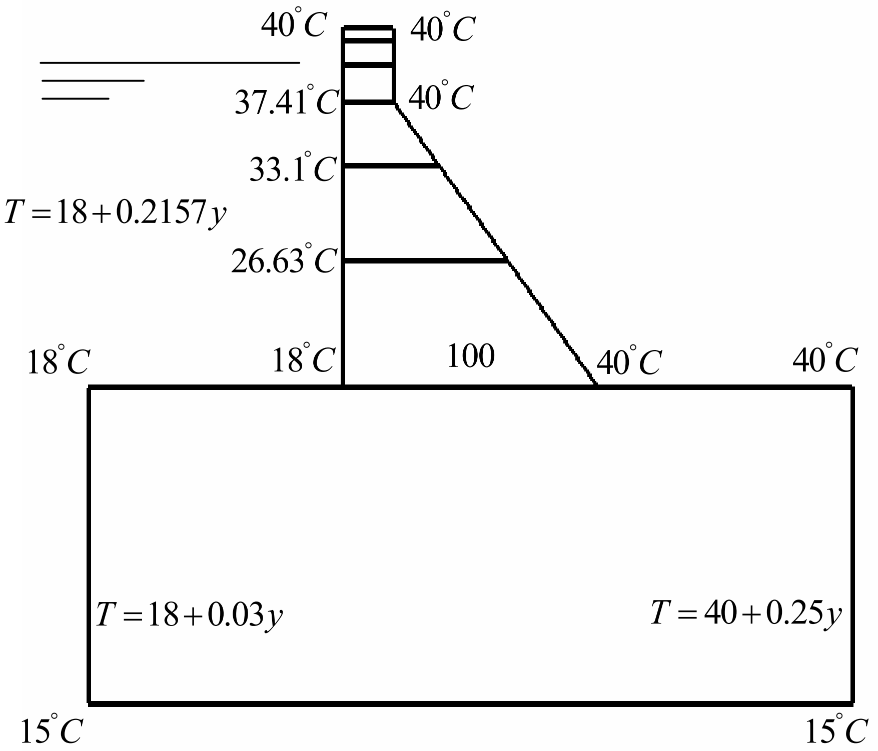
Analytical solution:

$$T = x^3 + y^3 + z^3 - 3yx^2 - 3xz^2 - 3zy^2$$

	Dofs	Time (sec.)	Relative Error（%）			
			Layer 4	Layer 5	Layer 6	Layer 7
BFM	27100	144	0.16	0.28	0.41	0.04
	15384	40	0.58	0.67	0.50	0.04
	8492	16	1.02	1.58	1.80	0.07
FEM	121023	265	0.32	0.37	0.45	0.45
	69679	82	0.33	0.38	0.70	0.46
	33386	25	0.65	0.66	1.46	0.90

Thermal analysis of concrete dam

- **Heterogeneous Case**



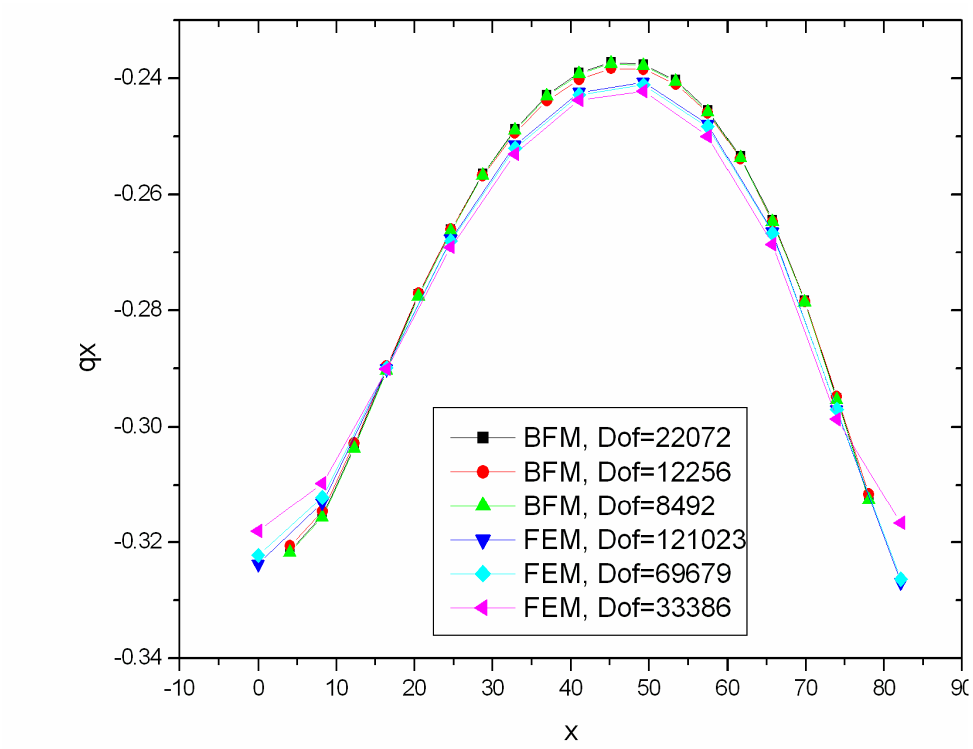
Boundary Condition

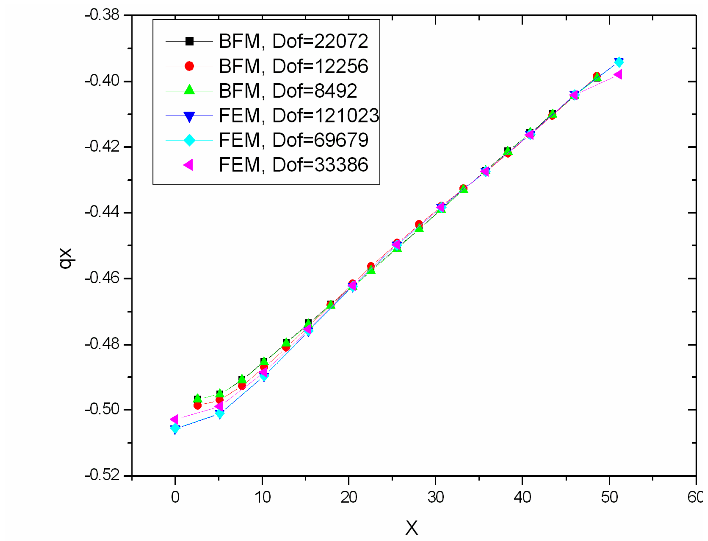
Layer	λ (W/m°C)
1	2.575
2	2.475
3	2.375
4	2.325
5	2.275
6	1.275
7	2.4378

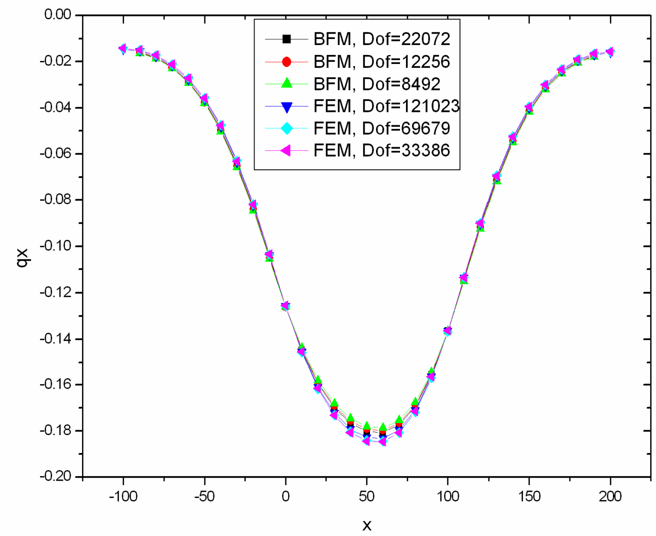
Materials for different layers

Thermal analysis of concrete dam

- **Results of q_x at evaluation points**

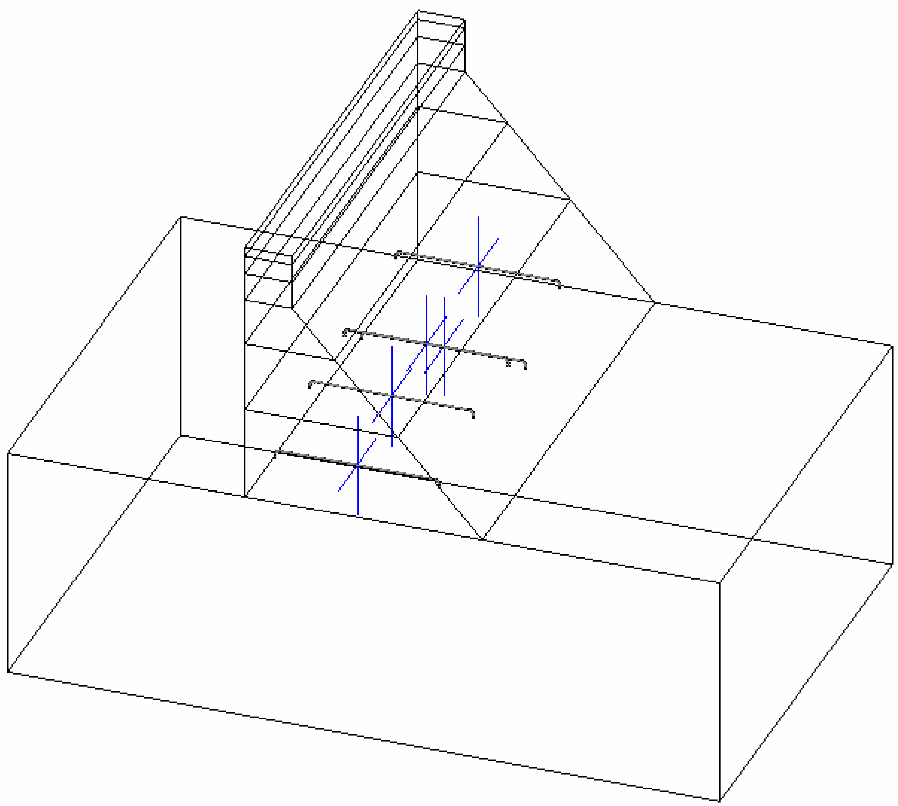




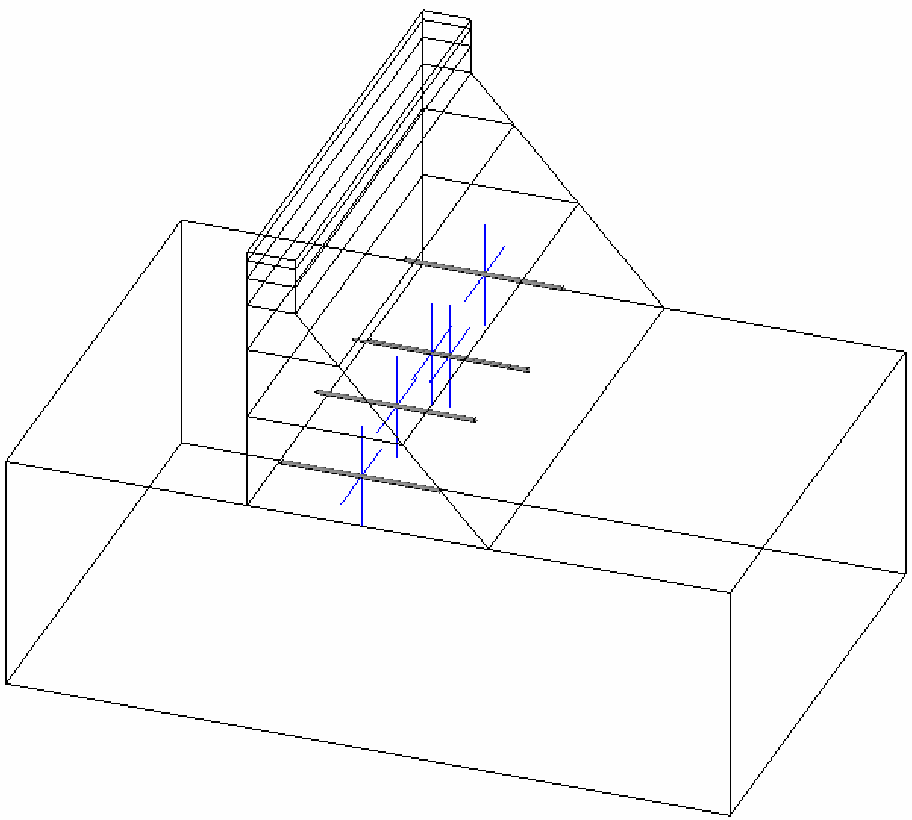


Thermal analysis of concrete dam

- **Cases with rebars and water pipes embeded**



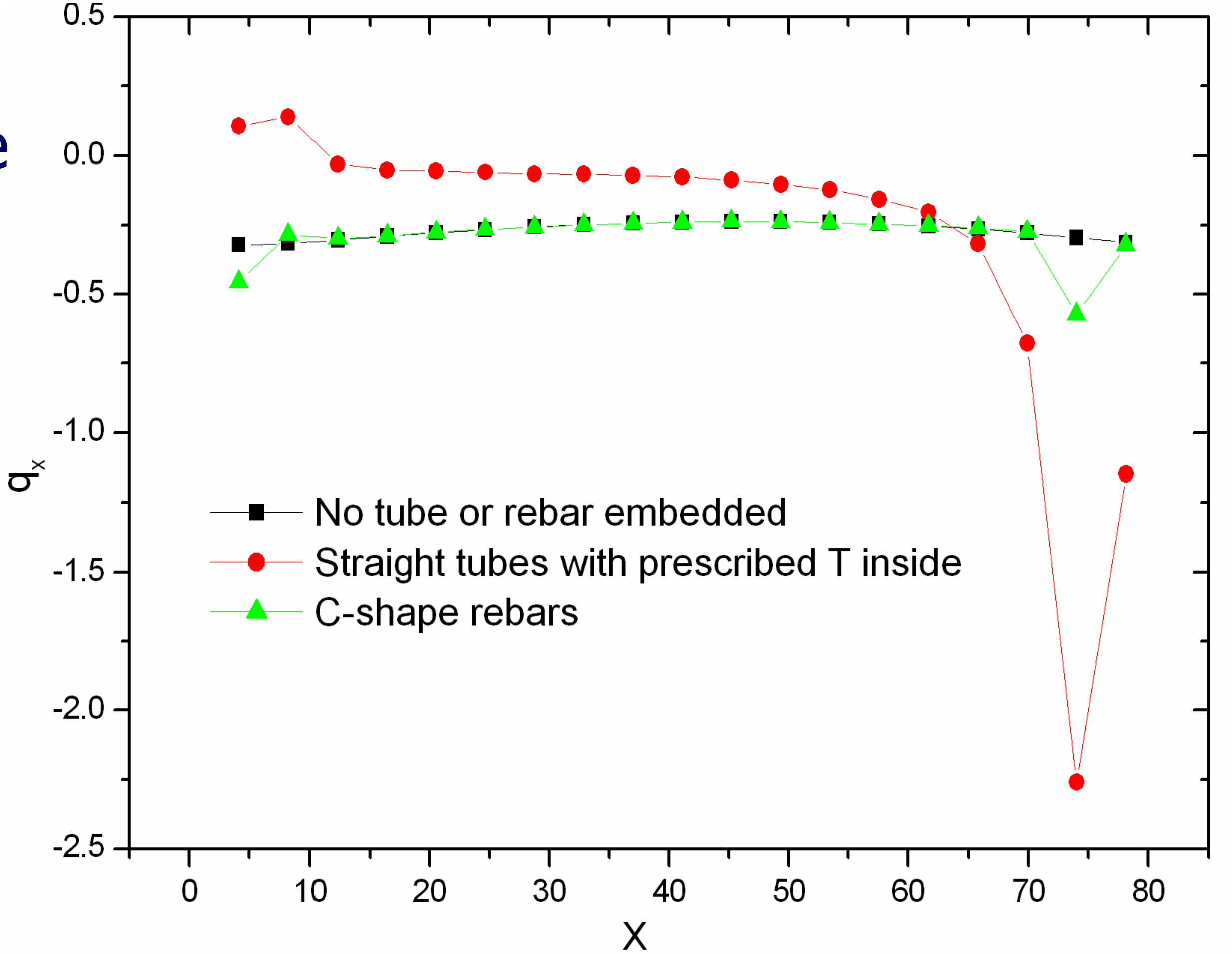
Five embeded rebars



Five embeded water pipes

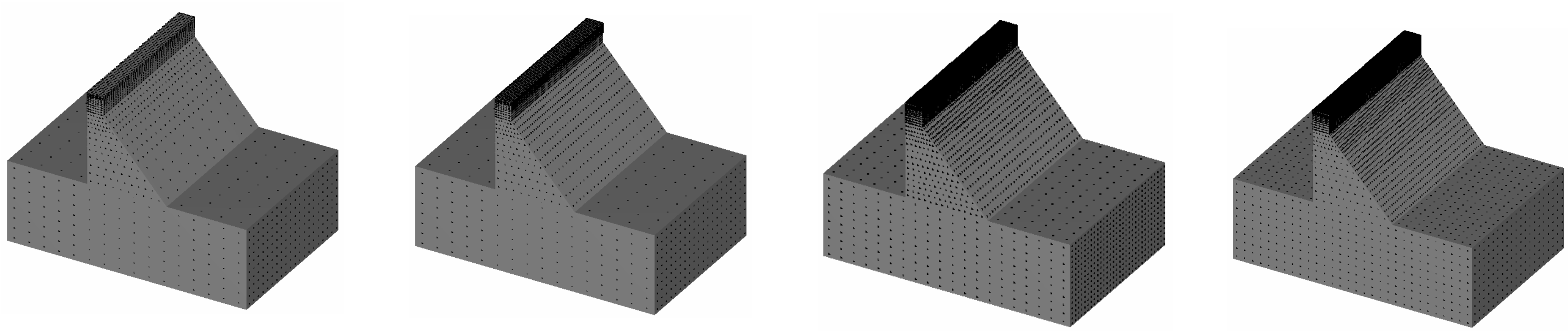
Thermal analysis of concrete dam

- The qx along a line segment near the rebar and pipe



Elastic analysis of concrete dam

- **Linear field** $u_x = \frac{2x+y+z}{2}, \quad u_y = \frac{x+2y+z}{2}, \quad u_z = \frac{x+y+2z}{2}$

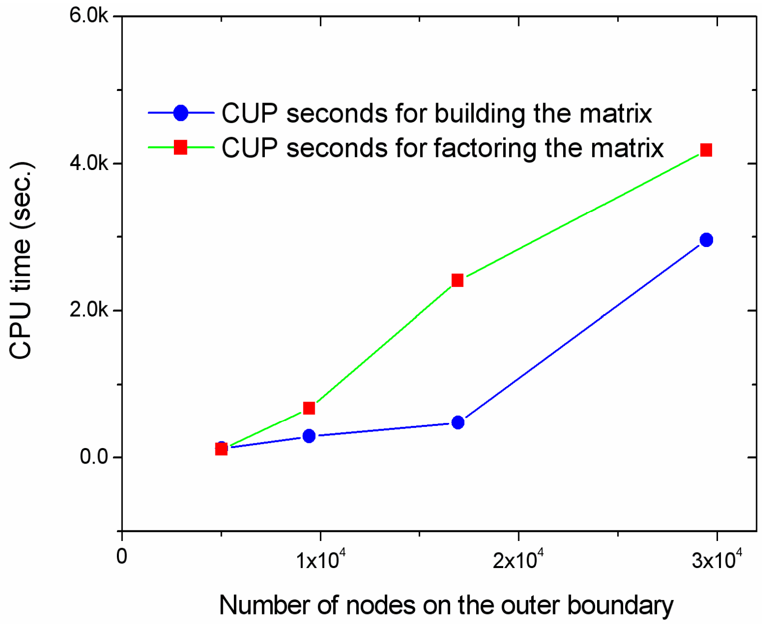


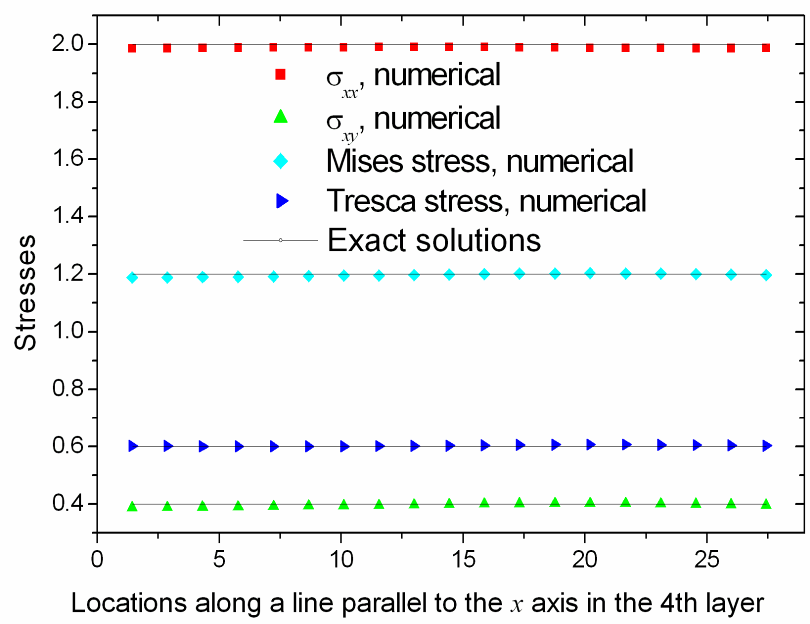
5016 Nodes 9444 Nodes 16956 Nodes 29480 Nodes

Layer No.	*1*	*2*	*3*	*4*	*5*	*6*	*7*
Traction_*x*	4.835	2.48	2.191	0.9077	0.2727	0.1369	0.01504
Traction_*y*	2.878	1.702	1.481	1.19	0.3241	0.08328	0.01582
Traction_*z*	1.769	0.5676	0.6066	0.2394	0.07427	0.03572	0.01752

Relative Error (%) for nodal tractions for layers when 5016 nodes are used

Elastic analysis of concrete dam (2)





DOFs	Time (sec.)		Relative Error（%）for top layer		
	T_{coef}	T_{equ}	Traction_*x*	Traction_*y*	Traction_*z*
5016	128	111	4.835	2.878	1.769
9444	290	676	3.043	1.764	0.8932
16956	379	2406	2.788	1.051	0.8932
29480	2960	4184	2.540	0.536	0.4101

Software development (research goal)

Automatic, accurate and efficient analysis of large-scale complex structures with arbitrary geometries and material composition

- **Structure (solver)**
  - **Surfaces**
    - Node generation
    - Intersections
    - TreeMLS
    - Derived classes
      - Parallelogram
      - Quadrilateral
      - Disc
      - Spherical
      - Ellipsoidal
      - Cylindrical
      - Conical
      - Torus
      - Sinusoidal tube
      - Spiral tube
  - **Domains**
    - Material
    - Fundamental solutions
    - FMM
    - Nodes
    - Faces
      - Pointer to surface
      - Boundary conditions
      - Adaptive integration
  - **Interfaces**
    - Perfect bond
    - Contact
    - Fracture
    - Welded

Conclusions and ongoing work

- Conclusions

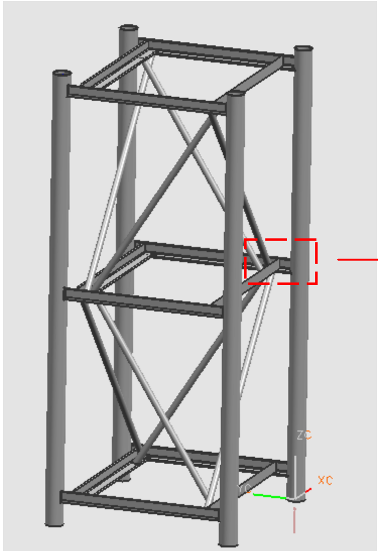
  - Thanks to the breakthroughs in BEM techniques, the situation for the BIE's application is not what it used to be. The BIE is ready to be used to develop analysis tools, which can perform much better than domain-type methods .

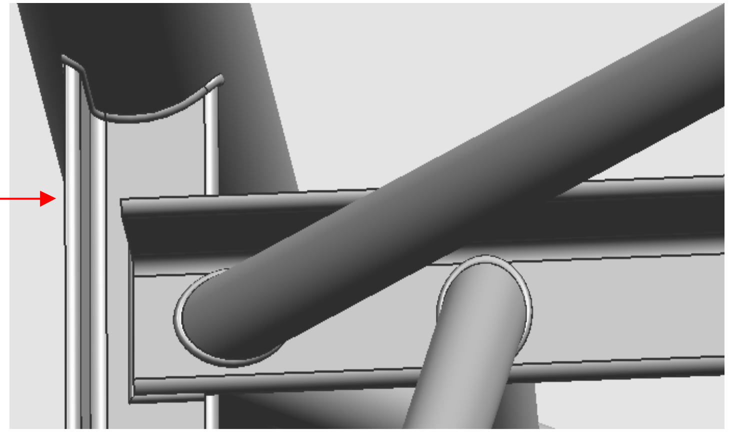
  - The BFM is a general framework for implementation of numerical methods based on BIE. It has real potential for seamless interactions with the solid modeling packages (NX UG, Pro/E, etc.).

  - The Hierarchical Matrix and ACA are purely algebraic algorithms. The BFM accelerated with Hierarchical Matrix and ACA can provide an automatic tool for large scale analysis of structures with complex geometries and material compositions.

Conclusions and ongoing work

- Ongoing work
  - Stress analysis of frame structures considering welding seams.





  - Combining the present method with DRM to perform visco-thermoelastic analysis on a concrete dam and simulate its construction process.