Boundary-only and fast meshless analysis of 3D potential problems

Masa. Tanaka, J. Zhang, M. Endo

Shinshu University
Faculty of Engineering

Nov. 17, 2004

Motivation

Two main difficulties in numerical analysis of engineering problems

- Mesh generation
- Large computational scale

- To overcome the first difficulty, we developed the

Hybrid Boundary Node Method (Hybrid BNM)

- To overcome the second difficulty, we implemented the

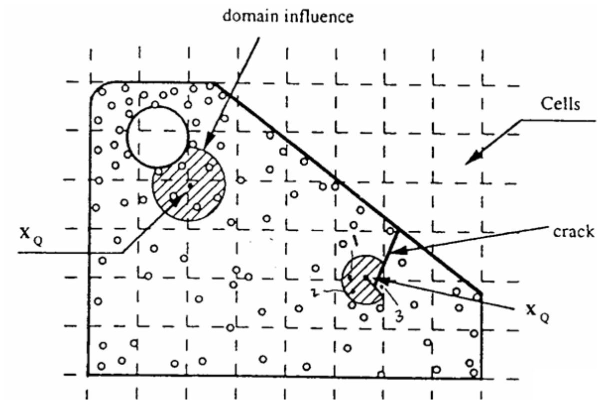
Fast Multipole Techniques (FMM)

Outline

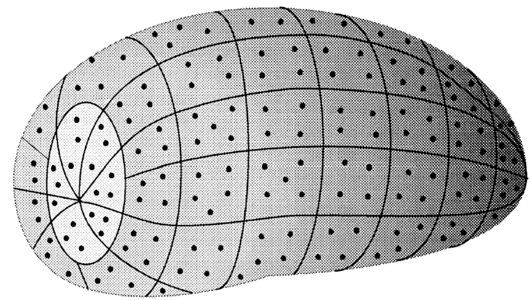
- Introduction to Hybrid BNM
- Formulations of the Hybrid BNM for 3D potential problems
- Accelerate Hybrid BNM with FMM
- Numerical results
- Conclusions

Introduction to Hybrid BNM

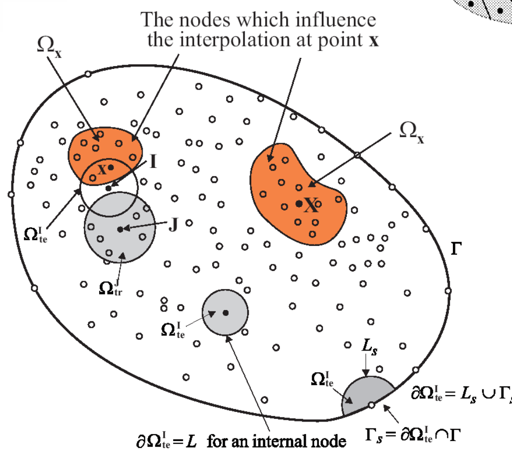
➢Element free Galerkin method (EFG)



➢Boundary Node Method (BNM)



➢Meshless Local Boundary Integral Equation (MLBIE)



Introduction to Hybrid BNM (2)

	Domain type	Boundary type
Pseudo meshless	Element free Galerkin method	Boundary node method
Truly meshless	Meshless Local Boundary Integral Equation	?

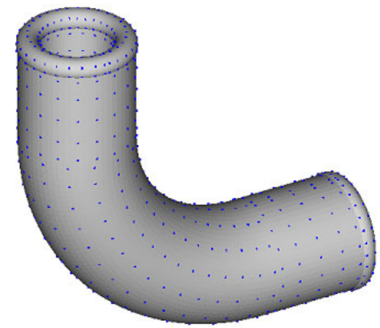
The answer is positive:

Hybrid Boundary Node Method

Formulations of Hybrid BNM

Main features:

- Combines modified functional with the *Moving Least Squares* (MLS) approximation
- Boundary-only truly meshless method



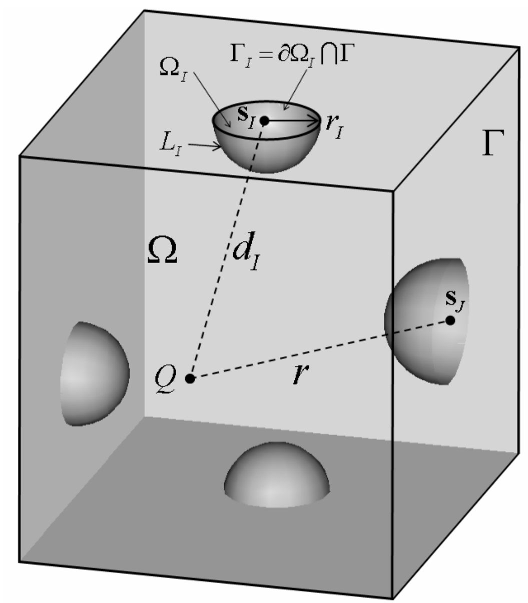
Example of meshless discretization

- Three independent variables
  - internal temperature $u = \sum_{J=1}^{N} u_J^s x_J \quad u_J^s = \frac{1}{\kappa} \frac{1}{4\pi r(Q, \mathbf{s}_J)}$
  - boundary temperature $\tilde{u}(\mathbf{s}) = \sum_{J=1}^{N} \Phi_J(\mathbf{s}) \hat{u}_J$
  - boundary normal flux $\tilde{q}(\mathbf{s}) = \sum_{J=1}^{N} \Phi_J(\mathbf{s}) \hat{q}_J$

Formulations of Hybrid BNM(2)

➢Local weak form

$$\int_{\Gamma}(q-\tilde{q})\delta u d\Gamma - \int_{\Omega} u_{,ii}\delta u d\Omega + \int_{\Gamma_q}(\tilde{q}-\bar{q})\delta\tilde{u} d\Gamma - \int_{\Gamma}(u-\tilde{u})\delta\tilde{q} d\Gamma = 0$$



$$\sum_{J=1}^{N}\int_{\Gamma_I} u_J^s v_I(Q) x_J d\Gamma = \sum_{J=1}^{n}\int_{\Gamma_I} \Phi_J(\mathbf{s}) v_I(Q)\hat{u}_J d\Gamma$$

$$\sum_{J=1}^{N}\int_{\Gamma_I} \frac{\partial u_J^s}{\partial n} v_I(Q) x_J d\Gamma = \sum_{J=1}^{n}\int_{\Gamma_I} \Phi_J(\mathbf{s}) v_I(Q)\hat{q}_J d\Gamma$$

Formulations of Hybrid BNM(3)

➢System of equations – final form

$$\boxed{\mathbf{Ux} = \mathbf{H\hat{u}}}$$

$$\boxed{\mathbf{Qx} = \mathbf{H\hat{q}}}$$

where

$$U_{IJ} = \int_{\Gamma_I} u_J^s v_I(Q)\,d\Gamma$$

$$Q_{IJ} = \int_{\Gamma_I} \frac{\partial u_J^s}{\partial n} v_I(Q)\,d\Gamma$$

$$H_{IJ} = \int_{\Gamma_I} \Phi_J(\mathbf{s}) v_I(Q)\,d\Gamma$$

➢Solution procedure

$$\boxed{\begin{array}{c}\mathbf{Ux} = \mathbf{H\hat{u}} \\ \mathbf{Qx} = \mathbf{H\hat{q}}\end{array}} \longrightarrow \boxed{\mathbf{Ax} = \mathbf{d}} \longrightarrow \boxed{\mathbf{x} = \mathbf{A}^{-1}\mathbf{d}}$$

$$\boxed{\mathbf{Ux} = \mathbf{H\hat{u}}} \longrightarrow \boxed{\mathbf{\hat{u}} = \mathbf{H}^{-1}\mathbf{Ux}}$$

$$\boxed{\mathbf{Qx} = \mathbf{H\hat{q}}} \longrightarrow \boxed{\mathbf{\hat{q}} = \mathbf{H}^{-1}\mathbf{Qx}}$$

For a direct solver

Memory requirement: $O(N^2)$

Computational time: $O(N^3)$

Accelerate Hybrid BNM with FMM

- **Iterative equation solver – GMRES**

The most time-consuming part of an iterative method is the calculation of matrix-vector product, $\mathbf{Ux}$ or $\mathbf{Qx}$, at each iteration step.

$$\text{The } I\text{-th row of } \mathbf{Ux} = \sum_{J=1}^{N} \int_{\Gamma_I} u_J^s v_I(Q) x_J \, d\Gamma$$

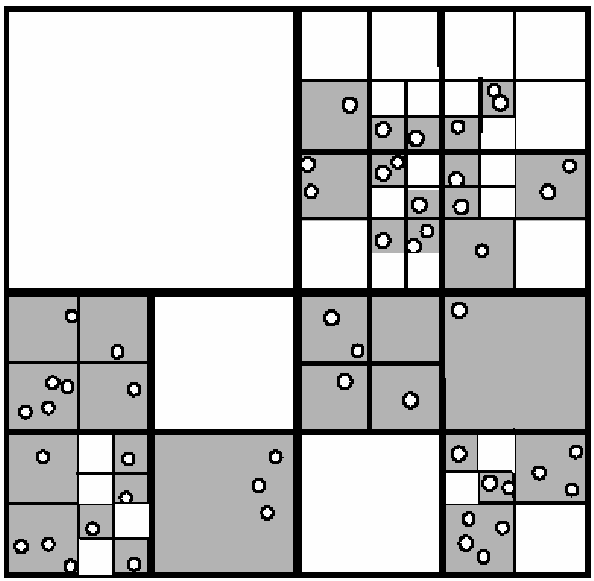
$$\text{The } I\text{-th row of } \mathbf{Qx} = \sum_{J=1}^{N} \int_{\Gamma_I} \frac{\partial u_J^s}{\partial n} v_I(Q) x_J \, d\Gamma$$

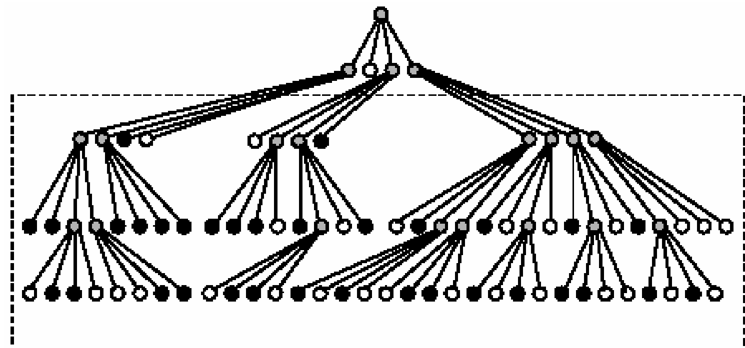
For an iterative solver
Memory requirement: $O(N^2)$
Computational time: $O(N^2)$

We use FMM to speed up these summations.

Accelerate Hybrid BNM with FMM (2)

- First, construct a hierarchy of boxes which refine the computational domain into small regions



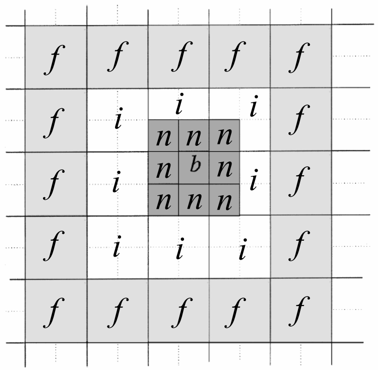


Quad-tree in 2D
Oct-tree in 3D

For simplicity, taking 2D case as an example.

Accelerate Hybrid BNM with FMM (3)

- **Using the hierarchical decomposition, we divide the sum into two parts**



f: far field

i: interaction list

n: neighbor

Near nodes $\sim n+b$

Far nodes $\sim i+f$

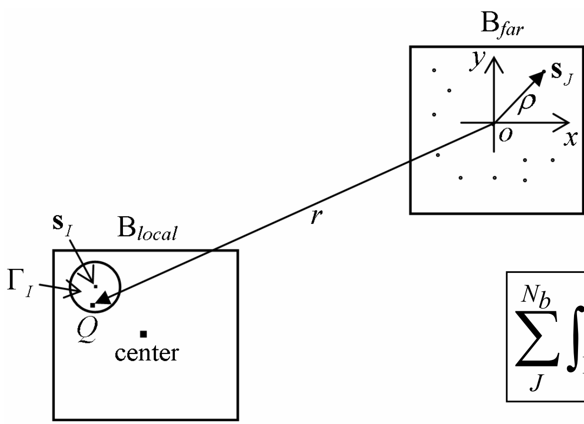
$$\boxed{\sum_{J=1}^{N}\int_{\Gamma_I} u_J^s v_I(Q) x_J \, d\Gamma = \sum_{J}^{\text{Near nodes}}\int_{\Gamma_I} u_J^s v_I(Q) x_J \, d\Gamma + \sum_{J}^{\text{Far nodes}}\int_{\Gamma_I} u_J^s v_I(Q) x_J \, d\Gamma}$$

(**Qx** is omitted temporarily)

We compute the sum over near nodes directly, while use multipole expansions to speed up the summation over far nodes.

Accelerate Hybrid BNM with FMM (4)

- **Consider two leaves B_{local} and B_{far}. B_{far} is on the interaction list of B_{local}. B_{local} contains node $\mathbf{s}_I$, while B_{far} contains N_b nodes**



$$u_J^s = \frac{1}{4\pi}\frac{1}{r(Q,\mathbf{s}_J)} = \frac{1}{4\pi}\sum_{n=0}^{\infty}\sum_{m=-n}^{n}\rho^n Y_n^{-m}(\alpha,\beta)\frac{Y_n^m(\theta,\phi)}{r^{n+1}}$$

$$\boxed{\sum_J^{N_b}\int_{\Gamma_I} u_J^s v_I(Q)x_J d\Gamma = \frac{1}{4\pi}\sum_{n=0}^{\infty}\sum_{m=-n}^{n} M_n^m \int_{\Gamma_I}\frac{Y_n^m(\theta,\phi)}{r^{n+1}}v_I(Q)d\Gamma}$$

$$\boxed{M_n^m = \sum_J^{N_b}\rho_J^n Y_n^{-m}(\alpha_J,\beta_J)x_J}$$

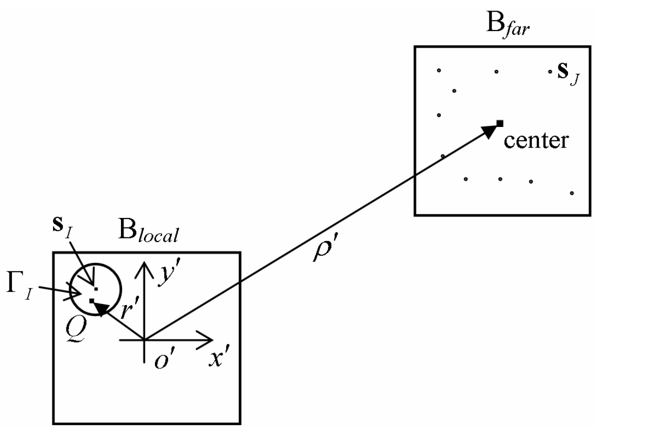
Condition: $r > \rho$

In above expanded expression, the pair of points Q and $\mathbf{s}_J$ is separated.

Accelerate Hybrid BNM with FMM (5)

- **Move origin of spherical coordinate system to B_{local}'s center**

$$\frac{Y_n^m(\theta,\phi)}{r^{n+1}} = \sum_{j=0}^{\infty}\sum_{k=-j}^{j} \frac{i^{|k-m|-|k|-|m|} A_n^m A_j^k Y_{n+j}^{m-k}(\alpha',\beta')}{(-1)^n A_{n+j}^{m-k} \rho'^{\,j+n+1}} Y_j^k(\theta',\phi') r'^{\,j}$$



$$\boxed{\sum_J^{N_b} \int_{\Gamma_I} u_J^s v_I(Q) x_J d\Gamma = \sum_{j=0}^{\infty}\sum_{k=-j}^{j} L_j^k S_j^k}$$

$$\boxed{L_j^k = \sum_{n=0}^{\infty}\sum_{m=-n}^{n} M_n^m \frac{i^{|k-m|-|k|-|m|} A_n^m A_j^k Y_{n+j}^{m-k}(\alpha',\beta')}{(-1)^n A_{n+j}^{m-k} \rho'^{\,j+n+1}}}$$

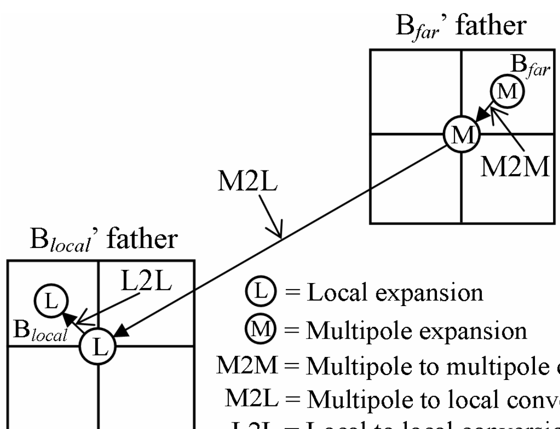
multipole to local conversion

Condition: $\rho' > 2r'$

$$\boxed{S_j^k = \frac{1}{4\pi}\int_{\Gamma_I} Y_j^k(\theta',\phi') r'^{\,j} v_I(Q) d\Gamma}$$

Accelerate Hybrid BNM with FMM (6)

- **When B_{far}'s parent belongs to the interaction list of B_{local}'s parent**



Ⓛ = Local expansion
Ⓜ = Multipole expansion
M2M = Multipole to multipole conversion
M2L = Multipole to local conversion
L2L = Local to local conversion

$$\boxed{M'^{k}_{j} = \sum_{n=0}^{j} \sum_{m=-n}^{n} M^{k-m}_{j-n} \frac{i^{|k|-|m|-|k-m|} A^{m}_{n} A^{k-m}_{j-n} \rho^{n} Y^{-m}_{n}(\alpha,\beta)}{A^{k}_{j}}}$$

multipole to multipole conversion

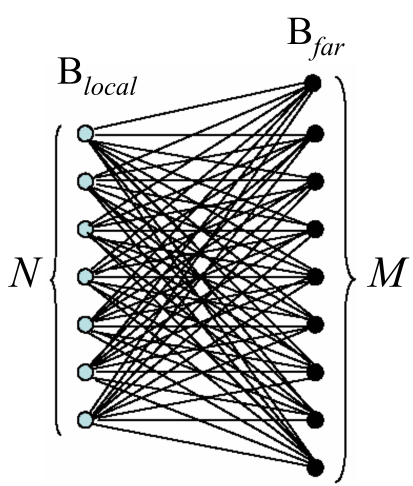
$$\boxed{L^{k}_{j} = \sum_{n=j}^{\infty} \sum_{m=-n}^{n} L'^{m}_{n} \frac{i^{|m|-|m-k|-|k|} A^{m-k}_{n-j} A^{k}_{j} Y^{m-k}_{n-j}(\alpha,\beta) \rho^{n-j}}{(-1)^{n+j} A^{m}_{n}}}$$

local to local conversion

Accelerate Hybrid BNM with FMM (7)

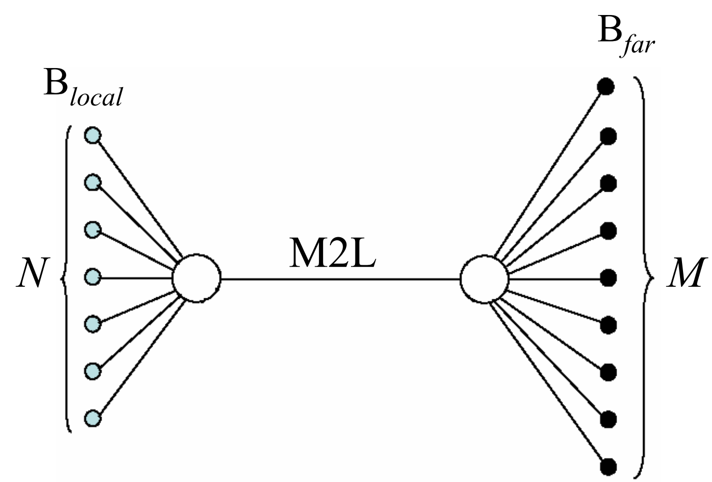
- **Comparison of the standard and multipole algorithms**

Standard algorithm



Total number of operations O(NM)

Multipole expansion algorithm

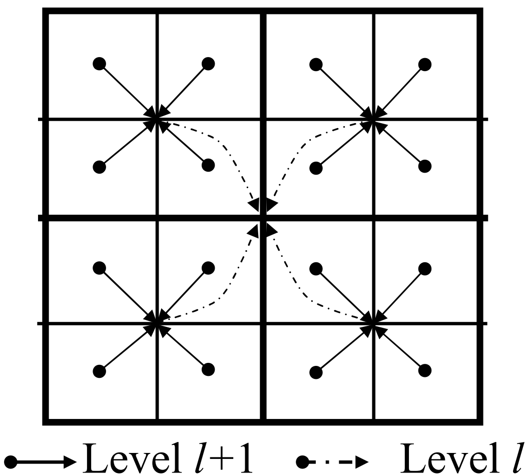


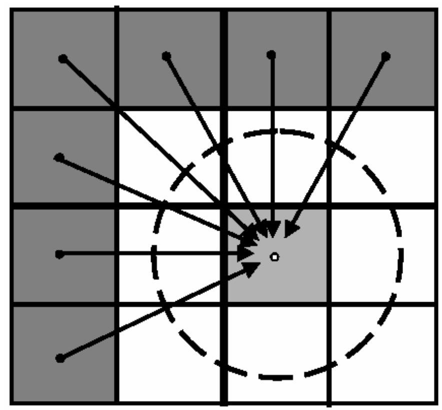
Total number of operations O(N+M)

Accelerate Hybrid BNM with FMM (8)

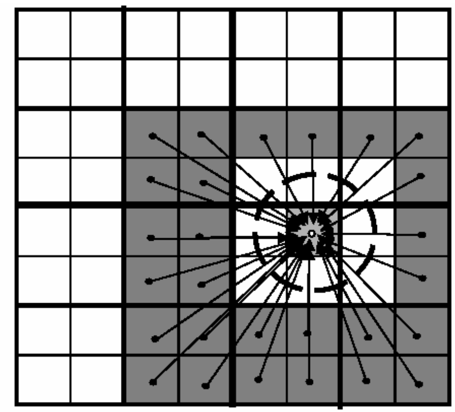
- **Upward pass and Downward pass**

In FMM, the multipole and local moments are orchestrated in the hierarchy of boxes.





Level l



Level l+1

Multipole moments are accumulated from leaves to the root (***Upward pass***); and local moments are distributed from the root to the leaves (***Downward pass***). This is accomplished in order of N operations.

Accelerate Hybrid BNM with FMM (9)

- **Product** $\mathbf{Qx} \sim \sum_{I=1}^{N} \int_{\Gamma_J} \frac{\partial u_I^s}{\partial n} v_J(Q) x_I d\Gamma$

$$\mathbf{Qx} \sim \sum_{J}^{N_b} \int_{\Gamma_I} \frac{\partial u_J^s}{\partial n} v_I(Q) x_J d\Gamma = \sum_{j=0}^{\infty} \sum_{k=-j}^{j} L_j^k \frac{\partial S_j^k}{\partial n}$$

$$\frac{\partial S_j^k}{\partial n} = \frac{1}{4\pi} \int_{\Gamma_I} \frac{\partial Y_j^k(\theta',\phi') r'^j}{\partial n} v_I(Q) d\Gamma$$

$$L_j^k = \sum_{n=0}^{\infty} \sum_{m=-n}^{n} M_n^m \frac{i^{|k-m|-|k|-|m|} A_n^m A_j^k Y_{n+j}^{m-k}(\alpha',\beta')}{(-1)^n A_{n+j}^{m-k} \rho'^{j+n+1}}$$

$$\mathbf{Ux} \sim \sum_{J=1}^{N_b} \int_{\Gamma_I} u_J^s v_I(Q) x_J d\Gamma = \sum_{j=0}^{\infty} \sum_{k=-j}^{j} L_j^k S_j^k$$

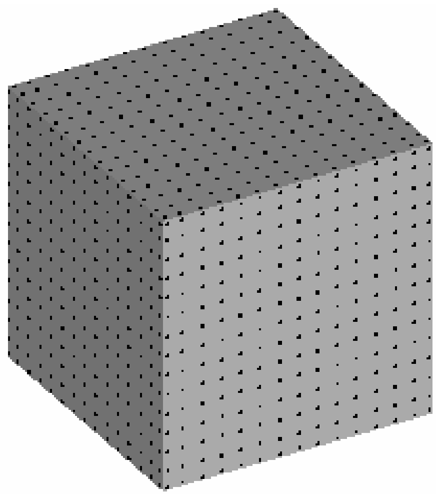
$$S_j^k = \frac{1}{4\pi} \int_{\Gamma_I} Y_j^k(\theta',\phi') r'^j v_I(Q) d\Gamma$$

$$L_j^k = \sum_{n=0}^{\infty} \sum_{m=-n}^{n} M_n^m \frac{i^{|k-m|-|k|-|m|} A_n^m A_j^k Y_{n+j}^{m-k}(\alpha',\beta')}{(-1)^n A_{n+j}^{m-k} \rho'^{j+n+1}}$$

Coefficients of local expansion L_j^k are the same for **Qx** and **Ux**.

Numerical results

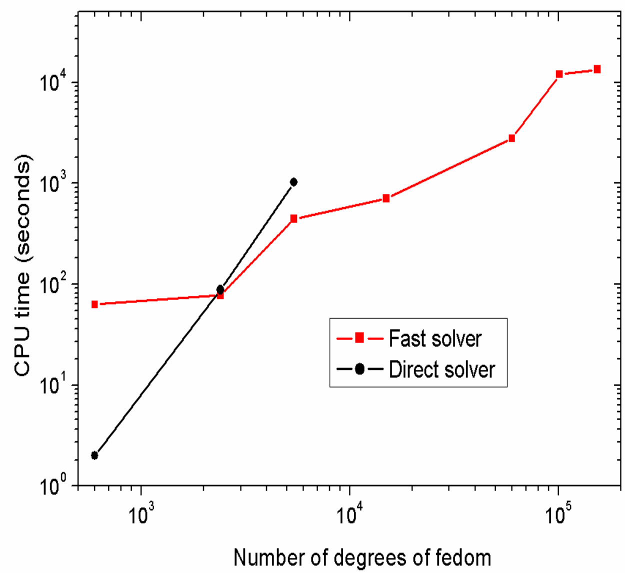
- **Results for a cube**



Max number of unknowns

Direct computation: 5400

FMM computation: 153600

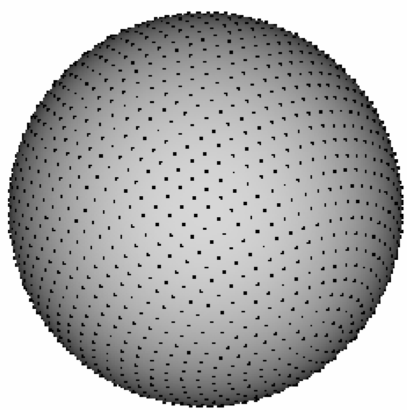


CPU time vs number of unknowns

Performed on a desktop computer with an Intel(R) Pentium(R) 4 CPU (1.99GHz)

Numerical results (2)

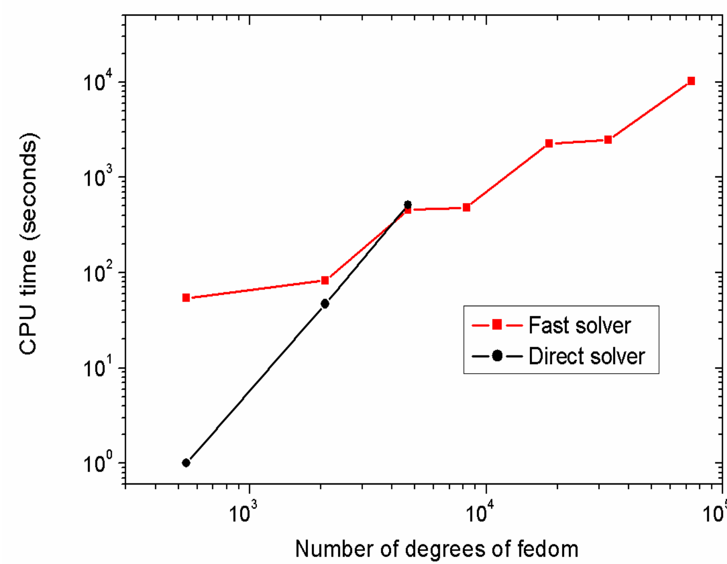
- **Results for a sphere**



Max number of unknowns

Direct computation: 4664

FMM computation: 73664

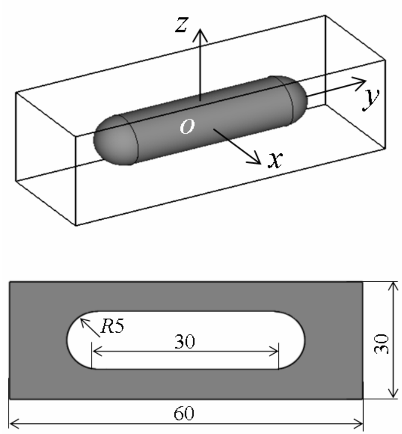


CPU time vs number of unknowns

Performed on a desktop computer with an Intel(R) Pentium(R) 4 CPU (1.99GHz)

Numerical results (3)

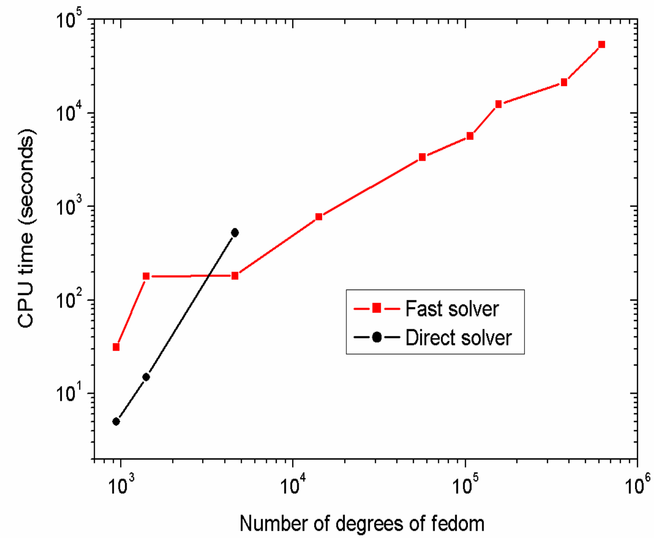
- **Results for a cuboid**



Max number of unknowns

Direct computation: 4598

FMM computation: 620708

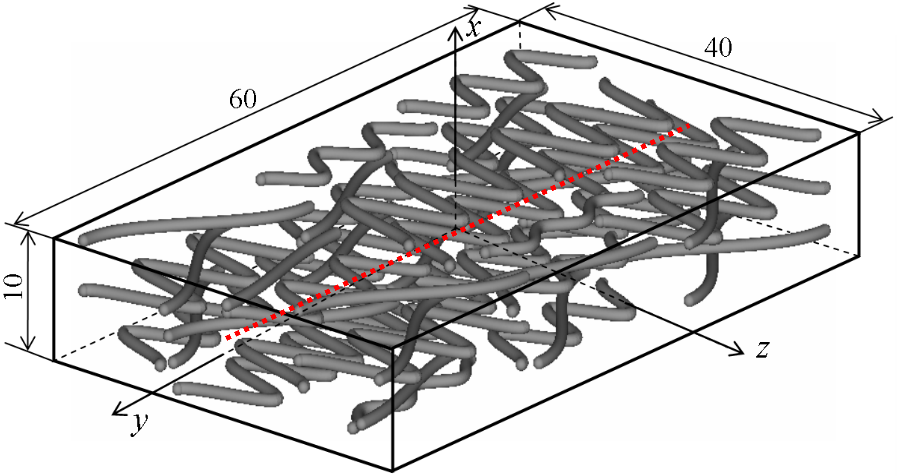


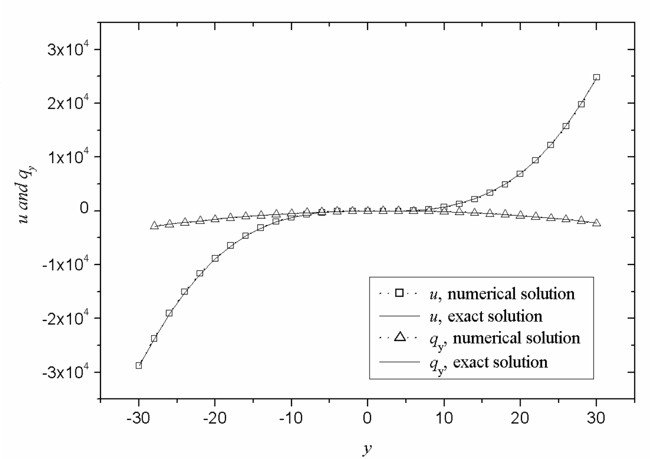
CPU time vs number of unknowns

Performed on a desktop computer with an Intel(R) Pentium(R) 4 CPU (1.99GHz)

Numerical results (4)

- **Results for a cuboid including many cavities in curved tube shape**





Number of nodes	108960	137370	159680	192500	255180
Relative error of nodal values	6.42%	3.81%	3.35%	2.33%	1.91%
CPU time (seconds)	25386	37840	43058	55670	67725

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

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

Conclusions

- The FMM has been incorporated into the Hybrid BNM. Numerical results clearly demonstrate that FM-HBNM is more efficient than the original Hybrid BNM when the number of unknowns is more than around 2400.

- The FM-HBNM retains the advantages of both the meshless method and the fast solver. It not only results in considerable savings in computing time and memory, but also substantially simplifies the discretization tasks for problems with complicated geometries. Therefore, the proposed method is especially applicable for large-scale simulations of bodies with intricate geometries.