INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Fast Multipole Method for the Symmetric Boundary Element Method in MEG/EEG

Jan Kybic — Maureen Clerc — Olivier Faugeras — Renaud Keriven — Théo Papadopoulo

N° 5415

Décembre 2004

Thème BIO



ISSN 0249-6399 ISRN INRIA/RR--5415--FR+ENG

INRIA
SOPHIA ANTIPOLIS

Fast Multipole Method for the Symmetric Boundary Element Method in MEG/EEG

Jan Kybic*, Maureen Clerc, Olivier Faugeras, Renaud Keriven, Théo Papadopoulo

Thème BIO — Systèmes biologiques
Projet Odyssée†

Rapport de recherche n° 5415 — Décembre 2004 — 34 pages

Abstract: The accurate solution of the forward electrostatic problem is an essential first step before solving the inverse problem of magneto- and electro-encephalography (MEG/EEG). The symmetric Galerkin boundary element method is accurate but is difficule to use for very large problems because of its computational complexity and memory requirements. We describe a fast multipole-based acceleration for the symmetric BEM with complexity. It creates a hierarchical structure of the elements and approximates far interactions using spherical harmonics expansions. The accelerated method is shown to be as accurate as the direct method, yet for large problems it is both faster and more economical in terms of memory consumption.

Key-words: Fast Multipole Method, Boundary Element Method, EEG, MEG

* Center for Applied Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague, Czech Republic. email: kybic@fel.cvut.cz, tel: +420 2 2435 7264.

† Odyssée is a joint project between ENPC - ENS Ulm - INRIA.

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles, BP 93, 06902 Sophia Antipolis Cedex (France)
Téléphone : +33 4 92 38 77 77 —Télécopie : +33 4 92 38 77 65

Méthode Multipole Rapide pour la résolution de la MEG/EEG par éléments finis surfaciques

Résumé : La résolution d'un problème direct électrostatique est un prérequis pour le problème inverse de la magnéto / électroencéphalographie. La formulation par éléments finis surfaciques symétriques est certes précise, mais est difficile à mettre en œuvre sur de gros maillages pour des raisons de temps de calcul et de place mémoire. Nous décrivons une accélération de la méthode symétrique par une méthode multipôle rapide. Le principe est de créer une structure hiérarchique et d'approcher les interactions lointaines par des développements en harmoniques sphériques. La solution accélérée est aussi précise que la solution de départ, tout en étant plus rapide et moins gourmande en mémoire sur de gros maillages.

Mots-clés : Méthode Multipôle Rapide, Eléments finis surfaciques, EEG, MEG

1 Introduction

The Boundary Element Method is widely used for solving the forward and inverse problems of Magneto - Electroencephalography (MEG/EEG) on realistic geometries [1, 2]. It unfortunately leads to huge linear systems which can be hard to handle.

The Fast Multipole Method (FMM) [3] acceleration significantly decreases the asymptotical time and memory complexity of solving the forward MEG/EEG problem. We described earlier a preliminary single-level FMM for the double-layer approach [4] that we extend here to multi-level FMM and adapt for the symmetric BEM [5]. To the best of our knowledge, this article describes the only implementation capable of accurately solving the MEG/EEG forward problem for realistic head models described by meshes with over 30,000 points (70,000 unknowns) on a single personal computer. Another unique feature of our approach is the ability to apply the symmetric BEM on the forward MEG/EEG problem with general topology consisting of a number of constant conductivity regions, instead of the classical nested volume topology.

1.1 Related work

The Finite Element Method (FEM) implementation [6] is capable of solving a problem of this size but requires a well-formed and topologically correct 3D mesh which is very difficult to automatically generate from the head MRI scans or the surface meshes.

There is an extensive literature dealing with FMM [3, 7–10] for gravitational or electromagnetic scattering calculations. However, only a few authors consider the electrostatic Maxwell problem and the symmetric BEM approach and even so they only treat problems with only one interface [11].

The precorrected-FFT acceleration [12], an alternative acceleration method, is compared to our approach in Section 5.1.

1.2 Problem definition

The relationship between the current source density $\mathbf{J}^{\mathrm{p}}$ and the corresponding induced potential V in a conducting environment is given by the quasi-static approximation of Maxwell equations [13, 14]

$$\nabla \cdot \left(\sigma \nabla V\right) = f = \nabla \cdot \mathbf{J}^{\mathrm{p}} \quad \text{in } \mathbb{R}^3 \ . \tag{1}$$

The forward problem consists of calculating V given $\mathbf{J}^{\mathrm{p}}$ and σ.

1.3 Generalized head model

We only consider piecewise constant conductivity head models, with different conductivities corresponding to different tissue types. Such models can be constructed readily (although laboriously) from segmented MRI data.

The previous approaches [1, 5, 15–18] have used a simple layered model (Figure 1, left). However, it is clear that a real head (Figure 1, middle) is topologically more intricate.

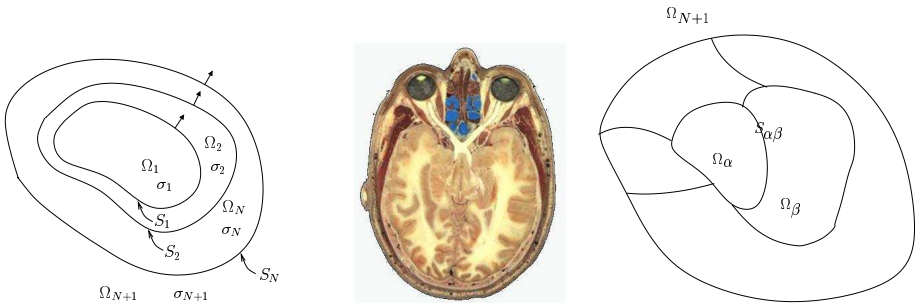


Figure 1: Traditionally, the head is modeled as a set of nested regions (left), while a real head is much more complicated (middle, transversal cut from the Visible Human project). The model proposed here assumes piecewise constant conductivity, with arbitrary partitioning (right).

The layered model fails to for example to model the openings present in the skull (eyes), or the junction between the two hemispheres via the corpus callosum, or the brain and skull defects caused by surgery. The effect of such defects on the localization accuracy can be significant [19]. We therefore propose a new formulation that can handle arbitrary partitioning of the space into volumes corresponding to different tissue types.

We consider a piecewise constant conductivity head model partitioning the space into $N+1$ disjoint connected open sets $\Omega_1, \ldots, \Omega_{N+1}$ such that $\bigcup_{\alpha=1}^{N} \overline{\Omega}_\alpha \cup \Omega_{N+1} = \mathbb{R}^3$. The volumes $\Omega_1, \ldots, \Omega_N$ with conductivities σ_α correspond to head tissues and are bounded, while Ω_{N+1} represents the air, has conductivity $\sigma_{N+1} = 0$ and extends to infinity (Figure 1, right)

For each pair $\Omega_\alpha, \Omega_\beta$ their common boundary $S_{\alpha\beta} = \partial\Omega_\alpha \cap \partial\Omega_\beta$ is either empty or can be decomposed into a finite number of connected regular surfaces[1], their normals $\mathbf{n}$ pointing by definition from Ω_α to Ω_β. Note that each $S_{\alpha\beta}$ is regular almost everywhere.

1.4 Connected Laplace problems

Since the conductivity is supposed to be piecewise constant, we can factor out σ from (1) to yield a set of Laplace problems connected by boundary conditions imposing the continuity of potential V and current $p = \sigma\partial_{\mathbf{n}} V$ across the interfaces:

$$\sigma_\alpha \Delta V = f \quad \text{in all } \Omega_\alpha \tag{2}$$

$$\left[V\right]_{S_{\alpha\beta}} = \left[p\right]_{S_{\alpha\beta}} = 0 \quad \text{on all } S_{\alpha\beta} \ . \tag{3}$$

[1] A surface S is connected if for each pair of points $A, B \in S$ there is a path in S between A and B. A surface is regular if at each point it can be locally approximated by a hyperplane.

1.5 Symmetric Boundary Element Method

The symmetric boundary element method (BEM) [20] transforms the differential equations (2) for V in $\mathbb{R}^3$ into a set of integral equations with unknowns V and p on the boundaries $S_{\alpha\beta}$, reducing the dimensionality of the problem. Their discretization leads to a symmetric system of linear equations. The symmetric BEM formulation is more complicated but more accurate and numerically stable than the alternative double-layer and single-layer BEM [21]. The original symmetric BEM was formulated for the layered model [21], we shall give its version for the generalized head model in Section 2.

1.6 Fast multipole method

The Fast Multipole Method (FMM) [3, 4, 7, 9] is a hierarchical approximation algorithm which significantly reduces the time and memory complexity required for the resolution of the linear system of equations produced by the BEM. To calculate the matrix-vector products needed by the iterative solver, it takes advantage of the fact that interaction between surface elements decreases quickly with distance. The influence of a group of elements which is sufficiently far from a point of interest can be expressed globally instead of individually. This decreases the overall complexity of calculating the matrix-vector product representing the pairwise interactions between N elements from $O(N^2)$ to $O(N)$. Our adaptation of the FMM algorithm is described in Section 3.

2 Symmetric Boundary Element Method

The symmetric BEM uses Green identities to convert the differential form (2) into a set of integral equations. Though its complete derivation is available in [21], we go over it again briefly, making the appropriate changes to allow for the generalized head model (Section 1.3), as already hinted in [22].

2.1 Free-space solution

Let us first consider a solution of (2) without considering the boundary conditions (3), as if we were in an infinite space of constant conductivity. We shall call

$$v_\alpha = -\big(f_\alpha * G\big)/\sigma_\alpha \tag{4}$$

a free-space solution of $\sigma_\alpha \Delta v_\alpha = f_\alpha$ in each Ω_α with $\sigma_\alpha \neq 0$. We have decomposed the sources as $f = \sum_\alpha f_\alpha$, such that for all $\mathbf{x} \notin \Omega_\alpha$, $f_\alpha(\mathbf{x}) = 0$. The function $G(\mathbf{r}) = 1/\big(4\pi\|\mathbf{r}\|\big)$ is the Green function of the Laplacian operator. It follows from the properties of the Green function [21] that v_α satisfies (2).

The free space solution v_α satisfies the "zero at infinity" (noted $\mathscr{H}$ in [21]) condition, which ensures that v_α approaches zero infinitely far from all sources.

In Ω_{N+1} (air), where $\sigma_{N+1} = 0$, we choose $v_{N+1} = 0$, a valid solution of the Poisson equation which is also compatible with the boundary condition $\mathscr{H}$.

2.2 Continuous form of the symmetric BEM

Let us now describe how to convert the partial differential equations (2), and boundary conditions (3) into the integral formulation. In each Ω_α we define a function

$$u_\alpha = \begin{cases} V - v_\alpha & \text{in } \Omega_\alpha \\ -v_\alpha & \text{elsewhere} , \end{cases} \tag{5}$$

harmonic ($\Delta u_\alpha = 0$) in $\mathbb{R}^3 \backslash \partial\Omega_\alpha$ and jumping across the boundary $\partial\Omega_\alpha$ (between Ω_α and $\mathbb{R}^3 \backslash \Omega_\alpha$) according to $[u_\alpha]_{\partial\Omega_\alpha} = V_\alpha$ and $\sigma_\alpha [\partial_{\mathbf{n}} u_\alpha]_{\partial\Omega_\alpha} \stackrel{\text{def}}{=} p_\alpha$, where V_α is the restriction of V on the boundary. (The jump of f between Ω_α and Ω_β is defined as $[f]_{S_{\alpha\beta}} = f^\alpha_{S_{\alpha\beta}} - f^\beta_{S_{\alpha\beta}}$, where $f^\alpha_{S_{\alpha\beta}}$, resp. $f^\beta_{S_{\alpha\beta}}$, are the limits of f when approaching a point on the surface $S_{\alpha\beta}$ from Ω_α, resp. Ω_β.)

Consider the surface $S_{\alpha\beta} = \Gamma$. First, we apply the Representation Theorem (Appendix 2) to calculate the limit of $u_\alpha = V - v_\alpha$ from Ω_α towards Γ, using the values of V and p on all the boundary $\partial\Omega_\alpha$, $\Gamma \subseteq \partial\Omega_\alpha$:

$$\left(V - v_\alpha\right)^-_\Gamma = \frac{[u_\alpha]_\Gamma}{2} - \sum_{\Theta = S_{\alpha\delta}} \left(\mathcal{D}_{\Gamma\Theta}\left[u_\alpha\right]_\Theta - \mathcal{S}_{\Gamma\Theta}\left[\partial_{\mathbf{n}} u_\alpha\right]_\Theta\right) = \frac{V_\Gamma}{2} - \sum_{\Theta = S_{\alpha\delta}} \left(\mathcal{D}_{\Gamma\Theta} V_\Theta - \sigma_\alpha^{-1} \mathcal{S}_{\Gamma\Theta} p_\Theta\right)$$

where $\mathcal{D}$ and $\mathcal{S}$ are the double- resp. single-layer potential integral operators [5, 21] (see also Appendix 1), the subscripts indicate the target and source surfaces they operate upon, and we sum over all regions Ω_δ adjacent to Ω_α. Second, we apply the Theorem to the limit of $u_\beta = V - v_\beta$ from Ω_β towards Γ:

$$\left(V - v_\beta\right)^+_\Gamma = \frac{[u_\beta]_\Gamma}{2} + \sum_{\Theta = S_{\beta\delta}} \left(\mathcal{D}_{\Gamma\Theta}\left[u_\beta\right]_\Theta - \mathcal{S}_{\Gamma\Theta}\left[\partial_{\mathbf{n}} u_\beta\right]_\Theta\right) = \frac{V_\Gamma}{2} + \sum_{\Theta = S_{\beta\delta}} \left(\mathcal{D}_{\Gamma\Theta} V_\Theta - \sigma_\beta^{-1} \mathcal{S}_{\Gamma\Theta} p_\Theta\right)$$

where the sign change is caused by the "inward" orientation of Γ with respect to Ω_β; this changes the sign of the normal derivative $\partial_{\mathbf{n}}$ involved in $\mathcal{D}$ and p. Thanks to the continuity of v_α,v_β, and V across Γ we can subtract the two previous equations:

$$\left(v_\beta - v_\alpha\right)_\Gamma = -2\mathcal{D}_{\Gamma\Gamma} V_\Gamma + (\sigma_\beta^{-1} + \sigma_\alpha^{-1}) \mathcal{S}_{\Gamma\Gamma} p_\Gamma - \sum_{(\Theta,\gamma)} \left(\mathcal{D}_{\Gamma\Theta} V_\Theta - \sigma_\gamma^{-1} \mathcal{S}_{\Gamma\Theta} p_\Theta\right) \tag{6}$$

with $(\Theta, \gamma) \in \left\{(S_{\alpha\delta}, \alpha); \delta \neq \beta\right\} \cup \left\{(S_{\delta\beta}, \beta); \delta \neq \alpha\right\}$. Similarly, by evaluating $\left(\sigma \partial_{\mathbf{n}} u\right)$ on both sides of Γ we get:

$$\left(\sigma_\beta \partial_{\mathbf{n}} v_\beta - \sigma_\alpha \partial_{\mathbf{n}} v_\alpha\right)_\Gamma = -(\sigma_\alpha + \sigma_\beta) \mathcal{N}_{\Gamma\Gamma} V_\Gamma + 2\mathcal{D}^*_{\Gamma\Gamma} p_\Gamma - \sum_{(\Theta,\gamma)} \left(\sigma_\gamma \mathcal{N}_{\Gamma\Theta} V_\Theta - \mathcal{D}^*_{\Gamma\Theta} p_\Theta\right) \tag{7}$$

with the same (Θ, γ). We have obtained a set of equations for V and p that must hold on all surfaces. Note that since $\sigma_{N+1} = 0$ the flow p across the external surfaces is 0 and the corresponding terms disappear.

2.3 Discretization of unknowns and surfaces

The surfaces $S_{\alpha\beta}$ are triangulated, with vertices common to several surfaces shared. To balance approximation errors, the potential V is discretized on the surfaces using piecewise linear P1 elements $\{\varphi_k\}$ and the flow p using the piecewise constant P0 elements $\{\psi_l\}$, like in [21] However, unlike the ψ_l, the P1 functions φ_k span several triangles which might belong to different surfaces. We shall therefore decompose them as $\varphi_k = \sum_{k'(k)} \varphi'_{k'}$, each of the partial functions $\varphi'_{k'}$ supported only on one (oriented) triangle $T_{k'}$ and hence belonging only to one surface. We approximate the $N = N_v + N_t$ unknowns as

$$V = \sum_{k=1}^{N_v} \sum_{k'(k)} x_k \varphi'_{k'} \qquad \text{and} \qquad p = \sum_{l=1}^{N_t} y_l \psi_l \tag{8}$$

where N_v, resp. N_t are the total numbers of vertices, resp. triangles, across all surfaces.

2.4 Discretization of the equations

Using a Galerkin approach, we take scalar products of both sides of (7) with P1 basis functions φ_i and of both sides of (6) with P0 basis functions ψ_j, on $S_{\alpha\beta} = \Gamma$:

$$\underbrace{\left\langle (\sigma_\beta \partial_{\mathbf{n}} v_\beta - \sigma_\alpha \partial_{\mathbf{n}} v_\alpha), \varphi_i \right\rangle}_{w_i} = \sum_{i'(i)} \Big(\sum_k x_k \sum_{k'(k)} \delta_{i'k'} \big(\mathsf{N}\big)_{i'k'} + \sum_l y_l \mu_{i'l} \big(\mathsf{D}^*\big)_{i'l} \Big) \tag{9}$$

$$\underbrace{\left\langle (v_\beta - v_\alpha), \psi_j \right\rangle}_{z_j} = \sum_k x_k \sum_{k'(k)} \gamma_{jk'} \big(\mathsf{D}\big)_{jk'} + \sum_l y_l \nu_{jl} \big(\mathsf{S}\big)_{jl} \, . \tag{10}$$

The discretized operator matrices are

$$\big(\mathsf{N}\big)_{i'k'} = \left\langle \mathcal{N}_{\Gamma\Theta} \varphi'_{k'}, \varphi'_{i'} \right\rangle \tag{11}$$

$$\big(\mathsf{S}\big)_{jl} = \left\langle \mathcal{S}_{\Gamma\Theta} \psi_l, \psi_j \right\rangle \tag{12}$$

$$\big(\mathsf{D}\big)_{jk'} = \big(\mathsf{D}^*\big)_{k'j} = \left\langle \mathcal{D}_{\Gamma\Theta} \varphi'_{k'}, \psi_j \right\rangle = \left\langle \mathcal{D}^*_{\Theta\Gamma} \psi_j, \varphi'_{k'} \right\rangle, \tag{13}$$

where φ_i, ψ_j are defined on the surface $\Theta = S_{\gamma,\delta}$ or $S_{\delta,\gamma}$, $\gamma = \alpha, \beta$, Ω_δ adjacent to Ω_γ and the constants are given by:

$\delta_{i'k'}$	$\mu_{i'l}$	$\gamma_{jk'}$	ν_{jl}	condition
$-(\sigma_\alpha + \sigma_\beta)$	2	-2	$(\sigma_\alpha^{-1} + \sigma_\beta^{-1})$	$\Theta = \Gamma$
$-\sigma_\alpha$	1	-1	$-\sigma_\alpha^{-1}$	$\Theta = S_{\alpha\delta}, \delta \neq \beta$
$-\sigma_\alpha$	-1	1	$-\sigma_\alpha^{-1}$	$\Theta = S_{\delta\alpha}, \delta \neq \beta$
$-\sigma_\beta$	1	-1	$-\sigma_\beta^{-1}$	$\Theta = S_{\delta\beta}, \delta \neq \alpha$
$-\sigma_\beta$	-1	1	$-\sigma_\beta^{-1}$	$\Theta = S_{\beta\delta}, \delta \neq \alpha$
0	0	0	0	otherwise

(14)

The advantage of this discretization is a good balance of the regularity of the integrated terms and the symmetry and sparsity of the system matrix A, provided that we order the equations as written above, (9), (10) and the unknowns as in $\mathsf{A}\,[\mathbf{x}\ \mathbf{y}]^T$, (8).

The double integrals involved in evaluating the elements of matrices D^*, D, S are done partly analytically [15, 23, 24]; the outer integrals are calculated numerically using 16-point triangle quadrature rule [25]. Elements $\mathbf{c}$ of the RHS are calculated by an adaptive numerical quadrature using the Cubpack++ library [26].

Discretized operator N is approximated using the relation ([20],Theorem 3.3.2):

$$\left\langle \mathcal{N}\varphi'_{i'}, \varphi'_{j'} \right\rangle = -(\mathbf{q}_{i'} \times \mathbf{n}_i)(\mathbf{q}_{j'} \times \mathbf{n}_j)\left\langle \mathcal{S}\psi_{j'}, \psi_{i'} \right\rangle \tag{15}$$

where the partial P1 basis functions $\varphi_{i'}(\mathbf{x}) = \big(\mathbf{q}_{i'} \cdot \mathbf{x} + \alpha_{i'}\big)\psi_{i'}(\mathbf{x})$ and $\varphi_{j'}(\mathbf{x}) = \big(\mathbf{q}_{j'} \cdot \mathbf{x} + \alpha_{j'}\big)\psi_{j'}(\mathbf{x})$ are supported only on triangles i' resp. j' with normals $\mathbf{n}_{i'}$,$\mathbf{n}_{j'}$.

Three nested layers

Suppose the head model to consist of three nested layers, denoted $1, 2, 3$. The system (9), (10) writes

$$\underbrace{\begin{pmatrix} (\sigma_1+\sigma_2)\mathsf{N}_{11} & -\sigma_2\mathsf{N}_{12} & 0 & -2\mathsf{D}^*_{11} & \mathsf{D}^*_{12} \\ -\sigma_2\mathsf{N}_{21} & (\sigma_2+\sigma_3)\mathsf{N}_{22} & -\sigma_3\mathsf{N}_{23} & \mathsf{D}^*_{21} & -2\mathsf{D}^*_{22} \\ 0 & -\sigma_3\mathsf{N}_{32} & \sigma_3\mathsf{N}_{33} & 0 & \mathsf{D}^*_{32} \\ -2\mathsf{D}_{11} & \mathsf{D}_{12} & 0 & (\sigma_1^{-1}+\sigma_2^{-1})\mathsf{S}_{11} & -\sigma_2^{-1}\mathsf{S}_{12} \\ \mathsf{D}_{21} & -2\mathsf{D}_{22} & \mathsf{D}_{23} & -\sigma_2^{-1}\mathsf{S}_{21} & (\sigma_1^{-1}+\sigma_2^{-1})\mathsf{S}_{22} \end{pmatrix}}_{\mathsf{A}} \cdot \underbrace{\begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \\ \mathbf{y}_1 \\ \mathbf{y}_2 \end{pmatrix}}_{\mathbf{u}} = \underbrace{\begin{pmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \\ \mathbf{w}_3 \\ \mathbf{z}_1 \\ \mathbf{z}_2 \end{pmatrix}}_{\mathbf{c}}$$

where $\mathbf{x}_\alpha$, $\mathbf{y}_\alpha$ are the unknown terms from (8) corresponding to the surface α. Similarly, $\mathbf{w}_\alpha$, $\mathbf{z}_\alpha$ are the corresponding source terms from (9), (10).

The zero blocks come from the fact that layers 1 and 3 do not touch a common volume and thus do not interact.

More general topology

If the topology is more general, we can no longer uniquely identify unknowns with surfaces. The block-diagonal structure is lost but the symmetry and the overall four-block structure of A remain:

$$\underbrace{\begin{pmatrix} \mathsf{N} & \mathsf{D}^* \\ \mathsf{D} & \mathsf{S} \end{pmatrix}}_{\mathsf{A}} \underbrace{\begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix}}_{\mathbf{u}} = \underbrace{\begin{pmatrix} \mathbf{w} \\ \mathbf{z} \end{pmatrix}}_{\mathbf{c}} . \tag{16}$$

For example, consider the model in Figure 2, consisting of two half-spheres S_a and S_b with the same radius, enclosing volumes with conductivities σ_1 and σ_2, glued together along a disk S_c and included within a larger sphere S_d enclosing a volume with conductivity σ_3, itself placed in a non-conductive medium. In this case, the submatrices have the following

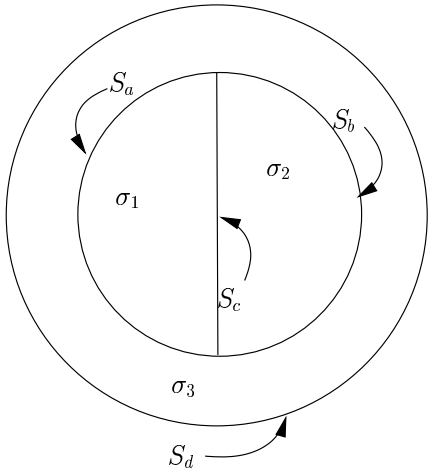


Figure 2: Example of a general, non-nested topology.

structure:

$$\mathsf{N} = \begin{pmatrix} (\sigma_1+\sigma_3)\mathsf{N}_{aa} & -\sigma_3\mathsf{N}_{ab} & -\sigma_1\mathsf{N}_{ac} & -\sigma_3\mathsf{N}_{ad} \\ -\sigma_3\mathsf{N}_{ba} & (\sigma_2+\sigma_3)\mathsf{N}_{bb} & -\sigma_2\mathsf{N}_{bc} & -\sigma_3\mathsf{N}_{bd} \\ -\sigma_1\mathsf{N}_{ca} & -\sigma_2\mathsf{N}_{cb} & (\sigma_1+\sigma_2)\mathsf{N}_{cc} & 0 \\ -\sigma_3\mathsf{N}_{da} & -\sigma_3\mathsf{N}_{db} & 0 & \sigma_3\mathsf{N}_{dd} \end{pmatrix}$$

$$\mathsf{D} = \begin{pmatrix} -2\mathsf{D}_{aa} & \mathsf{D}_{ab} & \mathsf{D}_{ac} & \mathsf{D}_{ad} \\ \mathsf{D}_{ba} & -2\mathsf{D}_{bb} & \mathsf{D}_{bc} & \mathsf{D}_{bd} \\ \mathsf{D}_{ca} & \mathsf{D}_{cb} & -2\mathsf{D}_{cc} & 0 \end{pmatrix}$$

and

$$\mathsf{S} = \begin{pmatrix} (\sigma_1^{-1}+\sigma_3^{-1})\mathsf{S}_{aa} & -\sigma_3^{-1}\mathsf{S}_{ab} & -\sigma_1^{-1}\mathsf{S}_{ac} \\ -\sigma_3^{-1}\mathsf{S}_{ba} & (\sigma_2^{-1}+\sigma_3^{-1})\mathsf{S}_{bb} & -\sigma_2^{-1}\mathsf{S}_{bc} \\ -\sigma_1^{-1}\mathsf{S}_{ca} & -\sigma_2^{-1}\mathsf{S}_{cb} & (\sigma_1^{-1}+\sigma_2^{-1})\mathsf{S}_{cc} \end{pmatrix} .$$

2.5 Deflation and preconditioning

The matrix A as defined by (16) has a kernel of dimension 1, related to the indeterminacy of the absolute levels of the potential V. Using deflation [27, 28] we obtain a regular matrix

$$\mathsf{A}' = \mathsf{A} + \omega \mathsf{I}_{\text{defl}} \tag{17}$$

where I_{defl} chooses for example a solution with mean surface potential zero and ω is a small constant chosen to approximately maximize the conditioning of A'.

The matrix A' is then preconditioned by its diagonal. In other words, instead of solving $\mathsf{A}'\mathbf{u} = \mathbf{c}$, we solve $(\mathsf{D}\mathsf{A}'\mathsf{D})(\mathsf{D}^{-1}\mathbf{u}) = \mathsf{D}\mathbf{c}$, where D is a diagonal matrix such that all diagonal elements of $\mathsf{D}\mathsf{A}'\mathsf{D}$ are equal to 1 in magnitude.

2.6 Iterative solver

The matrix A is big but relatively well structured: since the $\mathcal{N}$, $\mathcal{S}$, $\mathcal{D}$ interactions all decrease with distance between interacting elements, the elements tend to get smaller away from the diagonal. Therefore, it is advantageous to use iterative methods for solving $\mathsf{A}\mathbf{u} = \mathbf{c}$. We have selected the MINRES algorithm [29], which is a Krylov subspace method similar to conjugate gradient and designed for symmetric but not necessarily positive-definite matrices. Preconditioning reduces the number of iterations by approximately two.

3 Fast Multipole Method

The iterative (MINRES) method accesses the matrix A only through matrix-vector multiplications $\mathsf{A}\mathbf{u}$. The complexity of assembling the matrix A and of the $\mathsf{A}\mathbf{u}$ multiplication is $O(N^2)$, where N is the total number of unknowns. With the FMM [3, 7], the matrix does not need to be formed explicitly and multiplications are accelerated. We first introduce the notations and restate the key ideas behind the FMM method, then present the specific spherical harmonic expansion (Section 4) and tailor the FMM to our problem.

3.1 Idealised FMM

Let us present the key idea behind the FMM [7]. We do this in fairly abstract terms in order not to hide the main ideas behind complicated formulae. We then specialise the ideas for the MEG/EEG problem.

Suppose that we have to calculate the interaction between two groups of elements, $\mathcal{A}$ and $\mathcal{B}$, through the formula

$$y_j = \sum_{i \in \mathcal{A}} f_{ij} x_i, \quad \text{for all } j \in \mathcal{B} \,. \tag{18}$$

Suppose further that for each element i in $\mathcal{A}$, resp. j in $\mathcal{B}$, the term f_{ij} can be approximately written as

$$f_{ij} \approx \phi_j(\mathbf{C}) \odot \widetilde{\phi}_i(\mathbf{C}) \tag{19}$$

$\phi_j(\mathbf{C})$, resp. $\widetilde{\phi}_i(\mathbf{C})$, is called an expansion (to be made more precise later) and $\mathbf{C}$ is the center of the expansion. The expansion $\phi_i(\mathbf{C})$ is called outer (far, or multipole); the expansion $\widetilde{\phi}_i(\mathbf{C})$ is called inner (near, or local). These are functions $\phi_j : \mathbb{R}^3 \to \mathcal{Q}$, $\widetilde{\phi}_i : \mathbb{R}^3 \to \widetilde{\mathcal{Q}}$, with suitable domains $\mathcal{Q}, \widetilde{\mathcal{Q}}$. The operator $\odot : \mathcal{Q} \times \widetilde{\mathcal{Q}} \to \mathbb{R}$ is a bilinear, not necessarily commutative, operator.

We also define addition operators $\oplus : \mathcal{Q}\times\mathcal{Q} \to \mathcal{Q}$ and $\widetilde{\oplus} : \widetilde{\mathcal{Q}}\times\widetilde{\mathcal{Q}} \to \widetilde{\mathcal{Q}}$, with distributivity laws $(a \oplus b) \odot c = (a \odot c) + (b \odot c)$ and $a \odot (b \mathbin{\widetilde{\oplus}} c) = (a \odot b) + (a \odot c)$ and shortcuts for summation $\sum^{\oplus}$ and $\sum^{\widetilde{\oplus}}$. It is then easy to rearrange (18) to

$$y_j \approx \phi_j(\mathbf{C}) \odot \widetilde{\Phi}(\mathcal{A}, \mathbf{C}) = \phi_j(\mathbf{C}) \odot \Big(\sum_{i\in\mathcal{A}}^{\widetilde{\oplus}} x_i \widetilde{\phi}_i(\mathbf{C})\Big), \tag{20}$$

where we suppose a multiplication (scaling) operation $\mathbb{R}\times\mathcal{Q} \to \mathcal{Q}$ with the natural semantics. The term $\widetilde{\Phi}(\mathcal{A}, \mathbf{C})$, called an *inner-field*, represents the collective influence of all elements in $\mathcal{A}$ at point $\mathbf{C}$ and can be precalculated. This way, we have reduced the number of operations from $\|\mathcal{A}\|\|\mathcal{B}\|$ to $\|\mathcal{A}\| + \|\mathcal{B}\|$, i.e., from $O(N^2)$ to $O(N)$. In a dual fashion, we shall define an outer field $\Phi(\mathcal{A}, \mathbf{C}) = \sum_{i\in\mathcal{A}}^{\oplus} x_i \phi_i(\mathbf{C})$.

3.2 Grouping algorithm

In order for (19) to approximate f_{ij} with a given precision ε, it is necessary that the elements i and j be *well-separated*. More precisely,

$$d_i(\mathbf{C}) > \lambda d_j(\mathbf{C}) \overset{\text{def}}{\iff} \mathscr{D}_{ij} \implies \mid f_{ij} - \phi_j(\mathbf{C}) \odot \widetilde{\phi}_i(\mathbf{C}) \mid \leq \varepsilon, \tag{21}$$

where $d_i(\mathbf{C})$, $d_j(\mathbf{C})$ are distances from elements i resp. j to point $\mathbf{C}$, λ a relative distance. Consequently, as the condition $\mathscr{D}_{ij}$ in (21) does not hold for all pairs in $\mathcal{A}\times\mathcal{B}$, a more complicated algorithm than the one suggested by (20) is needed, using the expansion (19) only when it is safe to do so, i.e. when condition $\mathscr{D}_{ij}$ is satisfied. The simplest one is called the *grouping* or *middle-man* algorithm (Algorithm 1). It is based on dividing the elements in $\mathcal{A}$ and $\mathcal{B}$ into spatially constrained cells, so that the interaction between far cells can be carried out using the expansion (19).

We shall say that groups $\mathcal{A}_k$ and $\mathcal{B}_l$ are *well-separated* (a condition denoted $\mathscr{D}(\mathcal{A}_k, \mathcal{B}_l)$) iff for all $i \in \mathcal{A}_k$, $j \in \mathcal{B}_l$ $\mathscr{D}_{ij}$ holds. A pair $\mathcal{A}_k$, $\mathcal{B}_l$ which is not well-separated will be called *near*. The separability condition $\mathscr{D}(\mathcal{A}_k, \mathcal{B}_l) \iff \forall i \in \mathcal{A}_k, j \in \mathcal{B}_l;\ d_i(\mathbf{C}) > \lambda\, d_j(\mathbf{C})$ can be replaced by a stronger but easier to check condition

$$\mathscr{D}(\mathcal{A}_k, \mathcal{B}_l) \impliedby \mathscr{D}'(\mathcal{A}_k, \mathcal{B}_l) \iff \underbrace{d(\mathcal{A}_k, \mathbf{C})}_{\text{distance}} > \lambda \underbrace{\varrho_{\mathcal{B}_l}(\mathbf{C})}_{\text{radius}} \tag{22}$$

$$\text{with} \quad d(\mathcal{A}_k, \mathbf{C}) = \min_{i\in\mathcal{A}_k} d_i(\mathbf{C}) \quad \text{and} \quad \varrho_{\mathcal{B}_l}(\mathbf{C}) = \max_{j\in\mathcal{B}_l} d_j(\mathbf{C})\ .$$

A classical way of dividing $\mathcal{A}$ and $\mathcal{B}$ into groups is to partition space into rectangular cells of identical size [7].

Precision

The algorithm (and all its improvements that follow) is approximate in the sense that instead of calculating an exact value of $\mathbf{y} = \mathsf{F}\mathbf{x}$, where $\mathsf{F} = \{f_{ij}\}_{ij}$ is the interaction matrix, it

Algorithm 1: Middle-man interaction

Input: Sets of elements $\mathcal{A}$ and $\mathcal{B}$; a real vector x_i, $i \in \mathcal{A}$
Output: A vector of values y_j so that approximately $y_j = \sum_{i\in\mathcal{A}} x_i f_{ij}$ for all $j \in \mathcal{B}$.

Partition the elements from $\mathcal{A}$ into cells $\mathcal{A}_k$ with centers $\mathbf{C}_k$.
Partition the elements $\mathcal{B}$ into cells $\mathcal{B}_l$ with centers $\mathbf{C}_l$.
foreach *cell* $\mathcal{B}_l$ **do**
  establish the list $\mathcal{W}_l$ of all cells $\mathcal{A}_k$ that are well separated from $\mathcal{B}_l$ with respect to $\mathbf{C}_l$.

$$\mathcal{W}_l = \{\mathcal{A}_k;\ \mathcal{D}'(\mathcal{A}_k, \mathcal{B}_l\} \tag{23}$$

  foreach *cell* $\mathcal{A}_k \in \mathcal{W}_l$ **do**
    calculate the inner-field of $\mathcal{A}_k$ at $\mathbf{C}_l$

$$\widetilde{\Phi}(\mathcal{A}_k, \mathbf{C}_l) = \sum_{i\in\mathcal{A}_k}^{\widetilde{\oplus}} x_i \widetilde{\phi}_i(\mathbf{C}_l)$$

  foreach *element* $j \in \mathcal{B}_l$ **do**
    sum the *far contributions* from cells in $\mathcal{W}_l$ and the *local contributions* from the rest ($\mathcal{A}\backslash\mathcal{W}_l$):

$$y_j = \sum_{\mathcal{A}_k\in\mathcal{W}_l} \underbrace{\phi_j(\mathbf{C}_l) \odot \widetilde{\Phi}(\mathcal{A}_k, \mathbf{C}_l)}_{\text{far interactions}} + \sum_{i\in\mathcal{A}_k\notin\mathcal{W}_l} \underbrace{x_i f_{ij}}_{\text{local interactions}}$$

calculates an approximation $\mathbf{y}_{\text{approx}} = \mathsf{F}_{\text{approx}}\mathbf{x}$. If we can bound the error ε in (21) by choosing a suitable minimum relative distance λ, then the error of the vector $\mathbf{y}$ produced by Algorithm 1 is bounded the same way. This is discussed in Section 4.7.

Asymptotic complexity

Assuming that both $\mathcal{A}$ and $\mathcal{B}$ have about $\|\mathcal{A}\| \approx \|\mathcal{B}\| \approx N$ elements, we can analyze the asymptotic complexity of Algorithm 1 as $N \to \infty$. It turns out that the local interactions need $O(N^2 K_n/M)$ operations where M is the number of cells and K_n the number of near cells to any given cell, assumed to be a small constant independent of N. The preprocessing takes $O(N)$ operations and far interactions $O(NM)$. Therefore, the best strategy is to divide the N elements into $M \sim \sqrt{N}$ groups of $O(\sqrt{N})$ elements each, which gives the grouping algorithm a final complexity of $O(N^{3/2})$ as opposed to the brute-force approach (18) which requires $O(N^2)$ operations.

3.3 Single-level FMM algorithm

The disadvantage of the middle-man algorithm (Algorithm 1) is that the inner-field expansions have to be recalculated for each pair of $\mathcal{A}_k, \mathcal{B}_l$. Suppose there is a translation operator T that can convert an outer expansion at point $\mathbf{C}_k$ into an inner one around point $\mathbf{C}_l$

$$\widetilde{\phi}_j(\mathbf{C}_l) = \mathrm{T}_{\mathbf{C}_k\mathbf{C}_l}\ \phi_j(\mathbf{C}_k)\ . \tag{24}$$

In this way, we can improve the Algorithm 1 so that the outer fields Φ only have to be calculated once in total instead of once for each interacting group. They are then translated to centers of all other cells we want to interact with. The improved algorithm is called a single-level FMM algorithm (Algorithm 2).

Algorithm 2: Single-Level Fast Multipole Method interaction

Input: Sets of elements $\mathcal{A}$ and $\mathcal{B}$; a real vector x_i, $i \in \mathcal{A}$
Output: A vector of values y_j so that approximately $y_j = \sum_{i \in \mathcal{A}} x_i f_{ij}$ for all $j \in \mathcal{B}$.

1 Partition the elements from $\mathcal{A}$ into cells $\mathcal{A}_k$ with centers $\mathbf{C}_k$.
2 Partition the elements $\mathcal{B}$ into cells $\mathcal{B}_l$ with centers $\mathbf{C}_l$.
3 **foreach** *cell* $\mathcal{A}_k$ **do**
4 calculate the outer-field of $\mathcal{A}_k$ at $\mathbf{C}_k$.

$$\Phi(\mathcal{A}_k, \mathbf{C}_k) = \sum_{i \in \mathcal{A}_k}^{\oplus} x_i \phi_i(\mathbf{C}_k)$$

5 **foreach** *cell* $\mathcal{B}_l$ **do**
6 establish the list $\mathcal{W}_l$ of all cells $\mathcal{A}_k$ well separated from $\mathcal{B}_l$ with respect to $\mathbf{C}_l$, as in (5).
7 calculate the inner-field due to $\mathcal{W}_l$ at $\mathbf{C}_l$

$$\widetilde{\Phi}(\mathcal{W}_l, \mathbf{C}_l) = \sum_{\mathcal{A}_k \in \mathcal{W}_l}^{\widetilde{\oplus}} \underbrace{\mathrm{T}_{\mathbf{C}_k \mathbf{C}_l}\ \Phi(\mathcal{A}_k, \mathbf{C}_k)}_{\widetilde{\Phi}(\mathcal{A}_k, \mathbf{C}_l)}$$

foreach *element* $j \in \mathcal{B}_l$ **do**
8 sum the *far-field contributions* from the cells in $\mathcal{W}_l$ and the *local contributions* from the rest $(\mathcal{A} \backslash \mathcal{W}_l)$:

$$y_j = \underbrace{\phi_j(\mathbf{C}_l) \odot \widetilde{\Phi}(\mathcal{W}_l, \mathbf{C}_l)}_{\text{far interactions}} + \sum_{i \in \mathcal{A}_k \notin \mathcal{W}_l} \underbrace{x_i f_{ij}}_{\text{local interactions}}$$

Asymptotic complexity

We use the same assumptions as for Algorithm 1. The local interactions will again need $O(N^2 K_n / M)$ operations while the cost of the far interactions is reduced to $O(M^2 + N)$. This gives the optimal value of $M \sim N^{2/3}$ and an overall complexity of $O(N^{4/3})$.

3.4 Multi-level FMM algorithm

The major inefficiency of the single-level FMM (Algorithm 2) is that no cell size is optimal simultaneously for handling interactions at all distances. Using big cells is beneficial for handling far interactions since many elements are grouped and taken care of together. On the other hand, using big cells is wasteful for handling close interactions since too many elements are left to be treated by direct calculations rather than multipole expansions. And

vice-versa: smaller cells improve the efficiency of handling close interactions, but perform poorly for far interactions.

An obvious solution is to build a hierarchy of cells of different sizes. We create a tree of cells, such that children of each non-leaf cell X are themselves cells contained in X. Trees $\mathcal{A}$ resp. $\mathcal{B}$ are created from elements in input set $\mathcal{A}$ resp. output set $\mathcal{B}$ (using notations from Algorithms 1 and 2). This leads to a multi-level FMM algorithm, often called simply FMM [7].

Translation operators

We assume the existence of two more translation operators; an outer-to-outer operator R:

$$\phi_j(\mathbf{C}_l) = \mathrm{R}_{\mathbf{C}_k \mathbf{C}_l}\ \phi_j(\mathbf{C}_k) \tag{25}$$

and an inner-to-inner operator S:

$$\widetilde{\phi}_j(\mathbf{C}_l) = \mathrm{S}_{\mathbf{C}_k \mathbf{C}_l}\ \widetilde{\phi}_j(\mathbf{C}_k) \tag{26}$$

The operator R is used to calculate the outer field for each non-leaf cell X in the tree $\mathcal{A}$ from the outer fields of all its children Y:

$$\Phi(X, \mathbf{C}_X) = \sum_{Y \in \text{children of } X}^{\oplus} \mathrm{R}_{\mathbf{C}_Y \mathbf{C}_X}\ \Phi(Y, \mathbf{C}_Y)\ . \tag{27}$$

It is called an *up-sweep* of the tree $\mathcal{A}$. Similarly, during the *down-sweep* phase, operator S translates the inner-field from non-leaf cells in tree $\mathcal{B}$ to their children (Figure 3).

Interaction plan

Using the translation operators R, S and T, we now have the freedom of deciding at which level to handle each interaction between cells from trees $\mathcal{A}$ and $\mathcal{B}$. An interaction between cells $X \in \mathcal{A}$ and $Y \in \mathcal{B}$ can be handled in two ways: Either as a *local* interaction by explicitly calculating all the pairwise interactions (18) between elements of X and Y. Or, as a *far* interaction by using the approximation (18). To guarantee the accuracy of the approximation, cells are only allowed to interact using far interactions if they are well-separated in the sense of (22). Note that if two cells are not well separated, some of their descendents might be, as they are smaller.

To formalize, we define a plan $\mathcal{P}$ composed of local and far interactions $\mathcal{P}_L$ and $\mathcal{P}_F$; $\mathcal{P}_L, \mathcal{P}_F \subseteq \mathcal{A} \times \mathcal{B}$. All pairs $(X, Y) \in \mathcal{P}_L$ must be treated locally. A pair $(X, Y) \in \mathcal{P}_F$ (which must be well-separated) indicates that outer field $\Phi(X, \mathbf{C}_X)$, corresponding to all elements in X, must be translated to $\mathbf{C}_Y$ and applied to calculate the contributions of X on all elements in Y.

A plan $\mathcal{P}$ is *well-formed* if an interaction between each pair of leaves of trees $\mathcal{A}$ and $\mathcal{B}$ is handled exactly once. In other terms, if we orient the edges in the tree $\mathcal{A}$ up, from leaves to the root, and the edges in the tree $\mathcal{B}$ down, from the root to the leaves, and we add edges from X to Y for each $(X, Y) \in \mathcal{P}_L \cup \mathcal{P}_F$, then for each pair of leaf nodes $U \in \mathcal{A}$ and $V \in \mathcal{B}$, there must be exactly one path going from U to V.

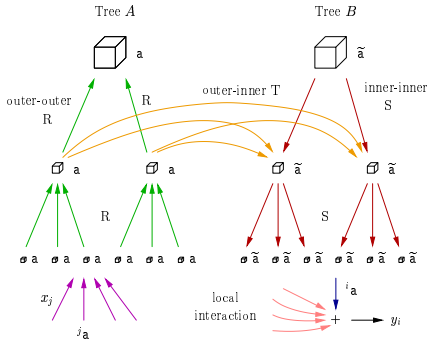


Figure 3: FMM interactions. The outer fields are propagated up the tree $\mathcal{A}$ during the up-sweep phase using operator R. They are transfered to tree $\mathcal{B}$ and converted to inner fields using operator T. The inner fields are propagated down in tree $\mathcal{B}$ using operator S. At leaf cells, the far interactions calculated from the inner-field coefficients are summed with local interactions.

Optimal interaction plan

We would like to find a plan involving the least computational effort. First of all, we want to minimize the number of local interactions $\|\mathcal{P}_L\|$. This is accomplished by using local interactions exclusively on pairs of leaf cells that are not well-separated. Once $\|\mathcal{P}_L\|$ is minimized, a good approximation of the remaining computational effort is the number of far interactions $\|\mathcal{P}_F\|$.

A classical approach is to descend simultaneously from the root to the leaves in both trees $\mathcal{A}$, $\mathcal{B}$ (which must be identical) and at each level to include all interactions that have not been already treated at higher levels, allowing only interactions between cells at the same level [3, 7].

We use a slightly more general planner, not requiring the trees $\mathcal{A}$, $\mathcal{B}$ to be identical. We would also like to allow interactions between cells at different levels (different sizes). However, the general task of choosing the smallest subset $\mathcal{P}_F$ of allowable far interactions that makes the plan well-formed is too difficult. Moreover, for implementation reasons that will be explained later (Section 4.6) we must limit the number of different $\mathrm{T}_{\mathbf{C}_k\mathbf{C}_l}$ operators as measured by the number of different $\mathbf{C}_k - \mathbf{C}_l$ vectors. We have therefore decided to apply a heuristic planner (Algorithm 3) with the additional constraint that for cells $(X, Y) \in \mathcal{P}_F$, the cell X is never smaller than cell Y. As a consequence, X and Y will be mostly the same size for X close to Y (but well-separated), while cells X further away Y may be bigger if the separability condition allows it. The condition at line 12 in Algorithm 3 that determines whether cell X is bigger than Y can also, instead of their physical size, compare their levels in the trees.

Algorithm 3: Create an interaction plan for a Multi-Level FMM algorithm

Input: Trees $\mathcal{A}$ and $\mathcal{B}$, with cells $\mathcal{A}_k$ and $\mathcal{B}_l$ as leaves.
Output: An interaction plan $\mathcal{P} = (\mathcal{P}_L, \mathcal{P}_F)$

```
1  P_L ← ∅ ; P_F ← ∅
2  return CreatePlan({ root of A}, root of B)
   // CreatePlan adds to P_F,P_L interactions between a set X (from A) and a cell Y (from B)
3  procedure CreatePlan(X, Y):
4  Z ← ∅  // Cells from X to be processed later
5  while X ≠ ∅ do
6  |  Take X from X
7  |  if  d(X, C_Y) > λ ϱ_Y(C_Y)  // are X and Y well-separated?
8  |  then
9  |  |  add (X, Y) into P_F
10 |
12 |  else if (X not a leaf) ∧ ( ϱ_X(C_X) ≥ ϱ_Y(C_Y)   ∨   Y is leaf ) then
13 |  |  put children of X into X
14 |  else add X to Z
15 if Y is leaf then
16 |  add {(Z, Y); Z ∈ Z} to P_L
17
18 else call CreatePlan(Z, Y') for all Y' children of Y.
```

Executing the interaction plan

Once the interaction plan is created, it is executed (Algorithm 4) whenever the Au product is needed, i.e. at each iteration of the optimizer. Note that the local interaction coefficients, the outer expansions ϕ_i, and the operators R, S, T can be precalculated. The remaining cost of executing the plan consists of applying the operators R, S, and T, and the precalculated coefficients f_{ij} of the interactions treated locally. Algorithms 3 and 4 describe our variant of the Multi-Level Fast Multipole Method (mlFMM or simply FMM). The three phases (planning, preparation, execution) are analogous to generating, compiling and executing a program in a very simple programming language.

Asymptotic complexity

Asymptotic complexity of the mlFMM depends on the spatial characteristics of the trees $\mathcal{A}$ and $\mathcal{B}$. We shall make two simple hypotheses:

$\mathscr{H}_1$: There are at most K_c elements in any leaf cell.

$\mathscr{H}_2$: The number of leaf cells from $\mathcal{A}$ that are not well-separated from a given leaf cell from $\mathcal{B}$ is at most K_n. (Same hypothesis as in Section 3.2.)

Both K_c, K_n are constants, independent of the total number of elements N. We shall see later that these hypotheses are reasonable and not difficult to fulfill (Section 4.6). As a consequence, the local interactions can be evaluated in $O(N)$ time, since each element can

Algorithm 4: Execute an interaction plan for a Multi-Level FMM algorithm

Input: Interaction plan $\mathcal{P} = (\mathcal{P}_L, \mathcal{P}_F)$; trees $\mathcal{A}$, $\mathcal{B}$; vector x_i. Precalculated values f_{ij}, ϕ_i, R, S, T.
Output: A vector y_j so that approximately $y_j = \sum_{i \in \mathcal{A}} x_i f_{ij}$ for all $j \in \mathcal{B}$.

$y_j \leftarrow 0$ for all j

// Treat local interactions
foreach $(X, Y) \in \mathcal{P}_L$ **do**
  foreach element $i \in X$, element $j \in Y$ **do**
    $y_j \leftarrow y_j + x_i f_{ij}$

// Up-sweep. Calculate outer-field recursively
foreach cell X from tree $\mathcal{A}$ **do**

$$\Phi(X, \mathbf{C}_X) = \begin{cases} \overset{\oplus}{\sum}_{\text{element } i \in X} x_i \phi_i(\mathbf{C}_X) & if X is a\ leaf \\ \overset{\oplus}{\sum}_{X' \text{ child of } X} \mathrm{R}_{\mathbf{C}_{X'}\mathbf{C}_X}\ \Phi(X', \mathbf{C}_{X'}) \end{cases}$$

// Down-sweep.
foreach cell Y from tree $\mathcal{B}$ **do**

$$\widetilde{\Phi}(Y, \mathbf{C}_Y) = \underbrace{\mathrm{S}_{\mathbf{C}_Z\mathbf{C}_Y} \widetilde{\Phi}(Z, \mathbf{C}_Z)}_{Z \text{ parent of } Y} \quad \widetilde{\oplus} \sum_{(X,Y) \in \mathcal{P}_F}^{\widetilde{\oplus}} \mathrm{T}_{\mathbf{C}_X\mathbf{C}_Y} \Phi(X, \mathbf{C}_X)$$

  if Y is leaf **then**
    foreach element $j \in Y$ **do**
      $y_j \leftarrow y_j + \phi_j(\mathbf{C}_Y) \odot \widetilde{\Phi}(Y, \mathbf{C}_Y)$

interact with at most $K_n K_c$ other ones. Calculating outer fields at nodes of tree $\mathcal{A}$ during the precalculation and up-sweep phases needs $O(N)$ operations as well, because there are N elements and at most N nodes and each node and each element is accessed exactly once. Propagating the inner field down in tree $\mathcal{B}$ also requires $O(N)$ operations.

The remaining cost to be accounted for concerns the outer-to-inner translations using operator T, which is proportional to the number of far interactions $\|\mathcal{P}_F\|$ in the plan. To upper bound it, we need two additional hypotheses:

$\mathscr{H}_3$: In any sphere of radius R, there are at most $K_s(R/\varrho)^{K_d}$ cells of radius bigger than ϱ.

$\mathscr{H}_4$: For any cell with radius ϱ, the radii of all its children are at least ϱ/K_r, $K_r > 1$.

Again, K_s, K_d, K_r are constants independent of N. It can be shown using $\mathscr{H}_3$ and $\mathscr{H}_4$ that the number of far interactions added to $\mathcal{P}_F$ for each cell in $\mathcal{B}$ by the planning Algorithm 3 is at most $K_s(\lambda K_r)^{K_d}$. Hence the total number of far interactions $\|\mathcal{P}_F\|$ is proportional to N. Therefore, the mlFMM is an $O(N)$ algorithm, which is a great improvement over the

$O(N^2)$ brute force approach. Unfortunately, we will see that the constants involved are rather large, so the asymptotic superiority of the (ml)FMM only appears for large values of N.

4 FMM for the MEG/EEG Boundary Element Method

Now that we have described the general FMM, in this section we apply it to the MEG/EEG forward problem, to quickly evaluate the matrix vector products ($\mathsf{N}\mathbf{x}$, $\mathsf{D}^*\mathbf{y}$, $\mathsf{D}\mathbf{x}$, $\mathsf{S}\mathbf{y}$) that appear in system (16).

4.1 Spherical harmonics

The integral operators $\mathcal{N}$, $\mathcal{S}$, $\mathcal{D}$, $\mathcal{D}^*$ all involve integration of the basis functions P0,P1 with various derivatives of the Green function $G(\mathbf{r},\mathbf{r}') \sim 1/\|\mathbf{r}-\mathbf{r}'\|$. One can decompose the Green function via the spherical harmonic approximation [8]:

$$\|\mathbf{r}'-\mathbf{C}\| > \|\mathbf{r}-\mathbf{C}\| \quad\Longrightarrow\quad \frac{1}{\|\mathbf{r}-\mathbf{r}'\|} = \sum_{n=0}^{\infty}\sum_{m=-n}^{n} I_n^{-m}(\mathbf{C}-\mathbf{r}')\, O_n^m(\mathbf{r}-\mathbf{C}) \tag{28}$$

where $\mathbf{C}$ is the center of expansion and I_n^m resp. O_n^m are the inner resp. outer spherical harmonics (Appendix 3). As the series (28) is infinite, we truncate it to order L to obtain a practically usable expression, guaranteeing an acceptable approximation error for $\mathbf{r}$, $\mathbf{r}'$ sufficiently far apart.

$$\underbrace{\|\mathbf{r}'-\mathbf{C}\| > \lambda\|\mathbf{r}-\mathbf{C}\|}_{\text{well-separatedness}} \Longrightarrow \frac{1}{\|\mathbf{r}-\mathbf{r}'\|} = \sum_{n=0}^{L}\sum_{m=-n}^{n} I_n^{-m}(\mathbf{C}-\mathbf{r}')\, O_n^m(\mathbf{r}-\mathbf{C}) + \text{error} \tag{29}$$

We can already recognize the structure of (21) with $I(\mathbf{C}) = \phi(\mathbf{C})$ and $O(\mathbf{C}) = \widetilde{\phi}(\mathbf{C})$.

4.2 Operator approximation

Each of the operators $\mathcal{N}$, $\mathcal{S}$, $\mathcal{D}$, $\mathcal{D}^*$ can now be approximated using (29), with the formalism defined in Section 3.1.

Operator $\mathcal{S}$

To approximate the discretized operator S (12), we integrate (29) with the P0 basis functions ψ_i. For each element (triangle) i we define the outer-field expansion coefficients

$${}^{i}\mathsf{a}_n^m(\mathbf{C}) = \int I_n^m(\mathbf{C}-\mathbf{r})\psi_i(\mathbf{r})\,\mathrm{d}\mathbf{r} \tag{30}$$

and the inner-field expansion coefficients

$${}^{i}\widetilde{\mathsf{a}}_n^m(\mathbf{C}) = \int O_n^m(\mathbf{C}-\mathbf{r})\psi_i(\mathbf{r})\,\mathrm{d}\mathbf{r}\ . \tag{31}$$

The operator $\mathcal{S}$ is then approximated (if elements i and j are well-separated) by

$$4\pi \left\langle \mathcal{S}\psi_i, \psi_j \right\rangle = {}^{i}\mathtt{a}(\mathbf{C}) \odot {}^{j}\widetilde{\mathtt{a}}(\mathbf{C}) \stackrel{\text{def}}{=} \sum_{\substack{n=0\ldots L \\ m=-n\ldots n}} (-1)^n \, {}^{i}\mathtt{a}_n^{-m}(\mathbf{C}) \, {}^{j}\widetilde{\mathtt{a}}_n^{m}(\mathbf{C}) \tag{32}$$

where we have used the symmetry relation $O_n^m(-\mathbf{r}) = (-1)^n O_n^m(\mathbf{r})$. We see that the expansion space $\mathcal{Q}$ as well as the dual space $\widetilde{\mathcal{Q}}$ (19) are homologous to the space of $(L+1)^2$ dimensional complex vectors; an expansion $\mathtt{a}$ or $\widetilde{\mathtt{a}}$ can be described by $(L+1)^2$ complex numbers $\mathtt{a}_n^m, \widetilde{\mathtt{a}}_n^m$. The addition $\oplus$ and multiplication $\mathbb{R} \times \mathcal{Q} \to \mathcal{Q}$ are defined as addition and multiplication of these complex coefficients.

Operator $\mathcal{D}$

We apply a gradient with respect to $\mathbf{r}$ to (29) and integrate the result with the basis functions ψ_i (P0) and $\phi'_{j'}$ (partial P1). We find that we need to define additional expansion coefficients

$${}^{i'}\mathtt{b}_n^m(\mathbf{C}) = \int \nabla I_n^m(\mathbf{C}-\mathbf{r}) \cdot \mathbf{n}_{i'} \varphi'_{i'}(\mathbf{r}) \, \mathrm{d}\mathbf{r} \tag{33}$$

$${}^{i'}\widetilde{\mathtt{b}}_n^m(\mathbf{C}) = \int \nabla O_n^m(\mathbf{C}-\mathbf{r}) \cdot \mathbf{n}_{i'} \varphi'_{i'}(\mathbf{r}) \, \mathrm{d}\mathbf{r} \tag{34}$$

so that the following approximation holds

$$-4\pi \left\langle \mathcal{D}\varphi'_{i'}, \psi_j \right\rangle = -4\pi \left\langle \varphi'_{i'}, \mathcal{D}^*\psi_j \right\rangle \quad = {}^{i'}\mathtt{b} \odot {}^{j}\widetilde{\mathtt{a}} = \sum_{n,m} (-1)^n \, {}^{i'}\mathtt{b}_n^{-m}(\mathbf{C}) \, {}^{j}\widetilde{\mathtt{a}}_n^{m}(\mathbf{C}) \tag{35}$$

$$= {}^{j}\mathtt{a} \odot {}^{i'}\widetilde{\mathtt{b}} = \sum_{n,m} (-1)^n \, {}^{j}\mathtt{a}_n^{m}(\mathbf{C}) \, {}^{i'}\widetilde{\mathtt{b}}_n^{-m}(\mathbf{C}) \tag{36}$$

Observe that even though the approximations are again representable by $(L+1)^2$ complex coefficients, the expansion spaces $\mathcal{Q}$ and $\widetilde{\mathcal{Q}}$ are no longer interchangeable. The coefficients ${}^{i}\mathtt{b}$ corresponding to a vertex (P1 basis function) i are calculated by aggregating the coefficients ${}^{i'}\mathtt{b}$ on all constituting triangles $T_{i'}$.

Operator $\mathcal{N}$

To calculate $(\mathsf{N})_{i'j'} = \left\langle \mathcal{N}\varphi'_{i'}, \varphi'_{j'} \right\rangle$ we define the expansion coefficients

$${}^{i'}\mathtt{c}_n^m = (\mathbf{q}_{i'} \times \mathbf{n}_{i'}) \, {}^{i}\mathtt{a}_n^m \qquad \text{and} \qquad {}^{j'}\widetilde{\mathtt{c}}_n^m = (\mathbf{q}_{j'} \times \mathbf{n}_{j'}) \, {}^{j}\widetilde{\mathtt{a}}_n^m \tag{37}$$

to obtain the approximation formula

$$4\pi \left\langle \mathcal{N}\varphi'_{i'}, \varphi'_{j'} \right\rangle = {}^{i'}\mathtt{c}(\mathbf{C}) \odot {}^{j'}\widetilde{\mathtt{c}}(\mathbf{C}) = \sum_{\substack{n=0\ldots L \\ m=-n\ldots n}} (-1)^n \, {}^{i'}\mathtt{c}^{-m,n}(\mathbf{C}) \; \cdot \; {}^{j'}\widetilde{\mathtt{c}}^{m,n}(\mathbf{C}) \; . \tag{38}$$

Each expansion is represented by $(L+1)^2$ 3D complex vectors, i.e. by $3(L+1)^2$ complex numbers. The usual rules of addition and multiplication by real numbers apply for the addition and scaling of the `c` expansions. The product $\odot$ (38) employs complex scalar product '$\cdot$'. As for coefficients `b`, the coefficients `c` are first calculated for the constituting triangles and then are aggregated to the vertices.

4.3 Translating multipolar representations

The formulas for the outer-outer, inner-inner and outer-inner operators R, S, resp. T (41) are generalized from [8] and are identical for all three types of expansion coefficients (`a`, `b`, `c`) provided that the addition and multiplication operations are changed accordingly (by using elementwise complex multiplication and addition for coefficients `c`).

$$\mathtt{x}_{n'}^{-m'}(\mathbf{M}) = \left(\mathrm{R}_{\mathbf{NM}}\,\mathtt{x}\right)_{n'}^{-m'} = \sum_{n=0\ldots\infty,\,m=-n\ldots n} I_n^{m-m'}(\mathbf{M}-\mathbf{N})\,\mathtt{x}_n^{-m}(\mathbf{N}) \tag{39}$$

$$\widetilde{\mathtt{x}}_{n'}^{-m'}(\mathbf{M}) = \left(\mathrm{S}_{\mathbf{NM}}\,\widetilde{\mathtt{x}}\right)_{n'}^{-m'} = \sum_{n=n'\ldots\infty,\,m=-n\ldots n} I_{n-n'}^{m-m'}(\mathbf{M}-\mathbf{N})\,\widetilde{\mathtt{x}}_n^{-m}(\mathbf{N}) \tag{40}$$

$$\widetilde{\mathtt{x}}_{n'}^{m'}(\mathbf{M}) = \left(\mathrm{T}_{\mathbf{NM}}\,\mathtt{x}\right)_{n'}^{m'} = \sum_{n=0\ldots\infty,\,m=-n\ldots n} O_{n+n'}^{m+m'}(\mathbf{M}-\mathbf{N})\,\mathtt{x}_n^{-m}(\mathbf{N})\ . \tag{41}$$

If the operators are to be used many times with the same difference vector $\mathbf{M}-\mathbf{N}$, it is advantageous to precompute the values $I_n^m(\mathbf{M}-\mathbf{N})$ and $O_n^m(\mathbf{M}-\mathbf{N})$. The same $(L+1)^2$ complex values can be used for both the operators R, S; the operator T needs $(2L+1)^2$ complex values.

4.4 Memory complexity

In order to execute Algorithm 4 as efficiently as possible, we need to precalculate and store:

a) The local interactions f_{ij} involved in the local plan $\mathcal{P}_F$, corresponding to at most $K_c K_n N/2$ real numbers.

b) The outer expansions ϕ_i at all elements, corresponding to $2(4N_v+N_t)(L+1)^2 = 4N(L+1)^2$ real values.

c) The precalculated values of the spherical harmonic expansions used by the R, S, T operators, the biggest being the operator T, corresponding to $2(2L+1)^2$ real values.

4.5 Timing of multipolar representation

All timing tests have been performed on a standard PC with Pentium processor at 1.4 GHz. The timings of the various operations involved in implementing the FMM are summarized in Table 1.

The most expensive is unfortunately unavoidable — to calculate the spherical harmonic expansions themselves. The second and third most expensive operations are applying the outer to inner translation operator T. Applying any of the translation operators by the

Table 1: Averaged timing of some operations involved in the FMM for $L = 10$, sorted by the elapsed time. The shortest operations cannot be measured reliably due to the calling overhead etc. By a we indicate the scalar expansion coefficients, while c stands for the vector expansion coefficients needed to approximate operator $\mathcal{N}$.

Operation to calculate	Time [μs]
a, b, c for 1 triangle	4218.75
apply T for c	3183.59
apply T for a	1054.69
apply R for c	933.10
apply S for c	788.92
apply R for a	310.06
apply S for a	261.23
O_n^m for T	83.62
$\mathsf{D}_{ij'}$ directly	47.00
S_{ij} directly	43.34
I_n^m for R or S	22.13
$\odot$ for c	12.28
$\odot$ for a	3.66
$y \leftarrow y + x$ for $y \in \mathbb{R}$	< 0.10
$y \leftarrow y + x$ for a	< 0.10
$y \leftarrow y + x$ for c	< 0.10
αa, $\alpha \in \mathbb{R}$	< 0.10

formulas above (Section 4.3) requires $O(L^4)$ operations, which makes it a bottleneck of the algorithm, especially operator T that operates on twice as many coefficients. Several approaches have been proposed to accelerate them:

a) using a FFT by taking advantage of the convolutional structure of the translation formula [8];

b) using a sequence of rotation, translation along the $z-$axis, and another rotation [7];

c) using a plane wave representation for faster summation [3].

These accelerations reduce the complexity from $O(L^4)$ to between $O(L^2 \log L)$ and $O(L^3)$ but are significantly more complex than the brute-force convolution (41) and thus faster only for high values of L; e.g. for $L \geq 20$ in the case of the FFT approach according to our tests.

The timings above unfortunately make the size of the problem for which it is advantageous to apply FMM very big. Even in a very simple case of two far groups of elements interacting 100 times, we find that we need more than 200 elements in each group for the FMM to be faster than calculating the pairwise interactions directly. For real geometries not all cells are well separated, so we can estimate that FMM only starts to be competitive for problems with more than 10^4 elements.

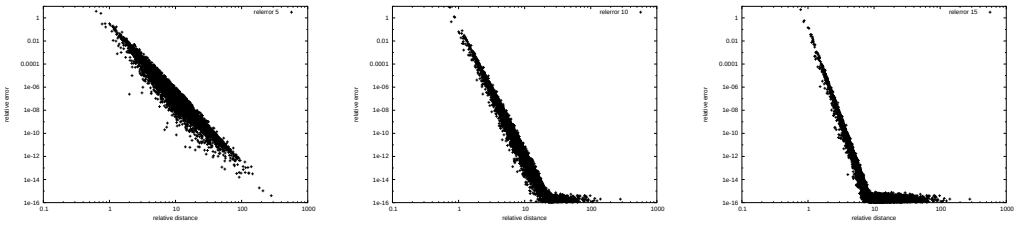
4.6 Tree structure

We have tested two basic approaches to generating the tree structure. The *median tree* approach consists of constructing a binary tree in a top-down fashion. In each step a set of elements belonging to a parent node is split into two halves using the best dividing plane parallel to one of the coordinate axes, and assigned to two children nodes. The splitting is stopped for nodes with less than K_c elements. The center of the bounding box of all the elements contained in a node is chosen as the expansion center for the node.

The tree constructed this way has the advantage of being ideally balanced and enabling the interactions to be treated at almost the ideal scale. Unfortunately, besides its larger depth, there is another important drawback: As the cell centers can take any value, we need to calculate (and store) the translation operators for any two interacting cells, which soon becomes prohibitive, especially for operator T. For example, for our three-sphere head model with $N = 1126$ elements in total and $K_c = 16$ elements per leaf node, there would be 38294 distinct operator T values to be computed.

For this reason, we have adopted a classical adaptive *octtree* structure. Its construction starts with a calculation of the bounding box of all elements. At each level, a parent box is divided into eight identical subboxes. Each of the eight children then receives elements whose center (center of gravity) falls into its box. The subdivision is stopped for nodes with less then K_c elements. Empty branches are pruned. Expansion centers are put into geometrical centers of each box.

An octtree may not be well balanced. However, it is relatively shallow, for example for our spherical head model with $N = 71686$ elements only 5 levels are needed with $K_c = 100$.



$L = 5$ $L = 10$ $L = 15$

Figure 4: The relative error to approximate $1/\|\mathbf{r}' - \mathbf{r}\|$ using (29) as a function of the relative distance for $L = 5, 10, 15$.

Its major advantage is that the expansion centers are guaranteed to lie on a Cartesian grid with known spacing. There are therefore only a limited number of vectors joining centers of interacting cells and thus an acceptable number of distinct translation operators to precompute. For our model with $N = 71686$ only 3776 operators T are needed.

There are two boxes associated with each cell. First, there is the cell box delimiting the cell itself. Its sizes are exactly half of the sizes of the parent box and it contains the centers of all elements belonging to the cell. Then there is a bounding box that encloses all the elements (not just their centers), which is often smaller than the cell box but can occasionally also be bigger. We use bounding boxes to determine whether two cells are well-separated.

4.7 Choice of parameters

The choice of λ and L is guided by time and accuracy considerations.

Kernel approximation accuracy.

It is easy to establish that the truncation error of (29) is proportional to $(\|\mathbf{r} - \mathbf{C}\|/\|\mathbf{r}' - \mathbf{C}\|)^{L+1}$ which is bounded by $\lambda^{-(L+1)}$. In Figure 4 we show the relative accuracy of the approximation of $1/\|\mathbf{r}' - \mathbf{r}\|$ using (29) for various values of L as a function of the relative distance $\|\mathbf{r}' - \mathbf{C}\|/\|\mathbf{r} - \mathbf{C}\|$ for 10^4 random points. The dependence corresponds nicely to the prediction formula, taking into account the 64 bit floating precision.

Time/accuracy trade-off.

Both λ and L influence the precision of the FMM approximation and the overall time consumption. The parameter λ (29) influences the size of the cell neighborhood to be treated locally; for a 3D problem we expect the number of near cells to grow as λ^3. For octtree-type cells the minimum useful value of λ is $\lambda_{\min} = \sqrt{3} \approx 1.73$ hence local-interaction

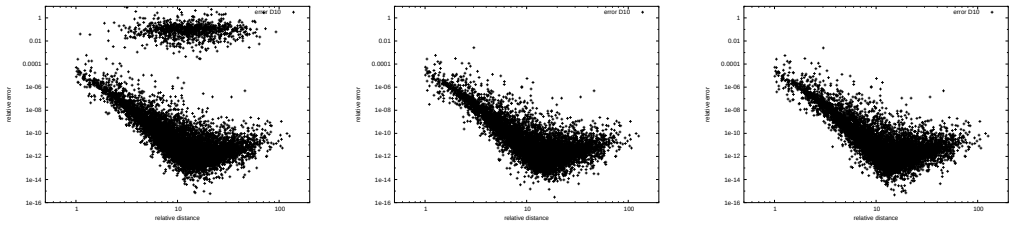


Figure 5: The relative error to approximate elements of $\mathsf{D}_{i'j}$, for 10^4 random triangles and $L = 10$, as a function of the relative distance using three quadrature rules (QR): 16 point symmetric triangle QR (left), 25 point product Gauss QR (middle), 144-point product Gauss QR (right).

neighborhood has $3^3 = 27$ cells. The influence of parameter L on the computation of the expansion coefficients, resp. translation operators is $O(L^2)$, resp. $O(L^4)$.

We have experimented various values of λ and L in our three-layer sphere models. The optimal value of λ was always between $\lambda_{\min}$ and 3. In order to limit the amount of memory needed to store precalculated local interaction coefficients we therefore decided to set $\lambda = 2$, in agreement with [9].

There is no consensus about what accuracy is fundamentally required to calculate the MEG/EEG BEM interactions for the inverse (source identification) problem, due to the measurement and modeling errors. We have therefore decided to tune the accuracy using our spherical head models with known analytical solution to the forward problem, requiring the difference between the FMM and non-FMM implementations to be less than 1% of the error with respect to the analytical solution. This corresponds to calculating the BEM interactions with relative accuracy about 10^{-4} which in turn required us to set $L = 10$.

Accuracy of the operator approximation

In order to choose the appropriate numerical quadrature procedure to implement (30,33), we evaluated the relative error of approximating S_{ij}, $\mathsf{D}_{i'j}$, $\mathsf{N}_{i'j'}$ as a function of relative distance between the elements (triangles) and the quadrature rule (QR) used for 10^4 randomly generated triangles of different sizes, shapes and orientations. The relative distance between triangles T_1 and T_2 was measured as the ratio $\min_{\mathbf{X}\in T_1} \|\mathbf{X}-\mathbf{C}\| / \max_{\mathbf{X}\in T_2} \|\mathbf{X}-\mathbf{C}\|$ with $\mathbf{C}$ being the center of expansion. For the direct computation, a 16 point QR is used (Section 2.4), but this QR is inadequate when dealing with the integration of spherical harmonics, especially for operator $\mathcal{D}$. Observe in Figure 5 the extra cloud of points in the middle figure, corresponding to interactions that should sum to zero. We decided to use the $5 \times 5 = 25$ point tensor product Gauss quadrature that performs adequately; increasing further the integration order does not improve accuracy.

4.8 Interaction between several trees

In a standard FMM there is only one tree that interacts with itself. In our case we have to deal with multiple trees for two reasons: 1) Each surface is handled separately, so that per-surface information such as conductivities does not have to be stored at each element. 2) We have two types of variables associated with each surface: potential V at its vertices (P1) and flow p at its faces $P0$. For any external surface (interface with air) only the P1 tree is built, since the flow p is known to be zero there (See also Section 2.2). For example, for a shell model with N_s spherical surfaces, there are N_s P1 trees and $(N_s - 1)$ P0 trees. The up-sweep phase is performed separately for each tree, hence in our example case there are $2N_s - 1$ up-sweeps.

A tree $\mathcal{A}$ will interact (through down-sweeps) with all other trees that correspond to surfaces delimiting a common volume with $\mathcal{A}$. Each down-sweep corresponds to one non-empty block in the system matrix A (Section 2.4). For our example, there will be $(12N_s - 15)$ down-sweeps.

All elements (vertices and faces) have a globally unique identification number that becomes an index of the corresponding x_i or y_i variable (for V resp. p (8)). The vertex indices go from 0 to $N_v - 1$, the triangle indices from N_v to $N_v + N_t - 1 = N - 1$.

To treat the multiple tree interactions efficiently:

a) The S values are shared globally (across all trees) to calculate the N values using (15).

b) The other precalculated direct interactions (D, N) are also shared globally to take advantage of the symmetry of the system matrix A in (16).

c) The outer expansion coefficients c are calculated directly from coefficients a (37). Evaluating coefficients a (30) and b (33) also shares many intermediate results, e.g. the values of Lagrange polynomials.

d) The expansion centers of all trees lie on a common grid so that the that the R, S and T operators can be shared. They currently also share the top-level box, which further simplifies some operations.

e) P0 tree nodes contain a coefficients. P1 tree nodes contain b and c coefficients that can be translated simultaneously in the up-sweep phase.

f) The outer expansions calculated during the up-sweep phase can be used to evaluate the interaction of this tree with all other trees. Therefore, all the up-sweeps should be completed before all the down-sweeps. Contrariwise, there is no need to store the values during the down-sweep, they can be calculated on the fly.

4.9 To summarise

The distinguishing features of our FMM algorithm can be summarised as follows.

a) It is tailored for symmetric BEM, and thererefore has to cope with the interactions between two different discretizations (P0/P1) of several surfaces through four different operators ($\mathcal{N}, \mathcal{D}, \mathcal{D}^*, \mathcal{S}$), see section 2 and [5, 21].

b) It can cope with interactions between different element trees.

c) It uses an efficient caching of intermediate results, avoiding the repetition of expensive calculations.

d) First an interaction (work) plan is established and the necessary values precalculated. The work plan is then executed at each iteration, eliminating overhead.

e) The octtree structure is pruned so that there are no empty branches.

f) The decision of whether two cells are sufficiently far apart is based on actual bounding boxes of the contained patches as opposed to the bounding boxes of the cells themselves.

g) It is implemented in a modular and efficient fashion in the functional language Ocaml[2].

5 Experiments

The superior accuracy of the symmetric BEM was already demonstrated in [5]. The purpose of this section is to demonstrate that this accuracy is not compromised by the FMM acceleration proposed and that it permits us to treat on a single computer in reasonable time far larger problems then the direct (non-accelerated) implementation.

We have used the same spherical head models as in [5] since analytical solutions are available for them. They consist of 3 concentric spheres with radii 0.87, 0.92, and 1.0, delimiting volumes with conductivities 1.0, 0.0125, 1.0 and 0.0, from inside towards outside. Different resolution meshes were used with $642, 2562$ and 10242 vertices per surface, with corresponding total number of unknowns for the symmetric BEM equal to $4486, 17926$ and 71686, respectively. A dipolar source was placed at distance 0.425 from the center.

The experiments were performed on a computer with a 1.6 Ghz 64 bit AMD Opteron processor with 5 GB of physical memory.

5.1 Single-sphere head models

The first series of experiments was performed on a simplified version of the head models containing one surface only, for a meaningful comparison with timings reported by Tissari and Rahola [12] using precorrected-FFT method. The programs were run with maximum number of elements per cell $K_c = 200$, expansion order $L = 10$, minimum relative distance $\lambda = 1.7$, and MINRES relative stopping threshold $\varepsilon = 10^{-6}$. The elapsed time, memory requirements and number of iterations are shown in Tables 2 and 3. The results of Tissari and Rahola are reported with their precision control parameters $p = 3, 4$ since they seem

[2] `http://caml.inria.fr`

Table 2: The elapsed CPU time and number of iterations needed to solve the forward problem for the single-sphere models as a function of the number of unknowns for the direct (no FMM) and FMM-accelerated symmetric BEM methods. Time needed to solve the problem using the double layer direct and precorrected-FFT method reported by Tissari and Rahola [12] for parameters $p = 3/p = 4$ are marked with *.

	Time [s]				Iterations	
Unknowns	direct sym.	direct dl.*	FMM	prec.-FFT*	FMM	prec.-FFT*
362		15		32/165		≤ 7
642	63		69		12	
1002		123		98/539		≤ 7
2252		714		270/1336		≤ 7
2562	1309		812		17	
5762		4760		724/4012		≤ 7
9002		12715		1223/6706		≤ 7
10242	26060		6325		26	
12962		27685		1934/9860		≤ 7

Table 3: The memory requirements to solve the forward problem for the single sphere head models as a function of the number of unknowns for the direct (no FMM) and FMM-accelerated symmetric BEM methods in MB (the actual memory usage fluctuates due to the garbage collector). Memory requirements for the double layer direct and precorrected-FFT method reported by Tissari and Rahola [12] for parameters $p = 3/p = 4$ are marked with *.

Unknowns	direct sym.	direct dl.*	FMM	prec.-FFT*
362		20		22/25
642	65		21	
1002		34		32/40
2252		98		66/71
2562	288		95	
5762		539		126/169
9002		1288		202/268
10242	2546		881	
12962		2649		296/389

to correspond to the precision required. Non-accelerated direct method results for both our methods and [12] are also shown.

According to the timings for the direct problem, their computer and implementation seem to be comparable to ours for the same number of unknowns, even though they use the double-layer formulation. It apparently has the advantage of involving a well-conditioned matrix with easy preconditioning, never requiring more than 6 or 7 iterations of the optimizer [12]. This emphasizes the assembly time with respect to the matrix-vector product evaluation time.

Table 4: The elapsed CPU time and number of iterations needed to solve the forward problem for the three-sphere models as a function of the number of unknowns for the direct (no FMM) and FMM-accelerated symmetric BEM methods. Relative error with respect to the analytical solution is also reported. We put between parentheses the extrapolated value for the largest problem, that could not be solved by the direct method due to lack of memory.

	Time [s]		Iterations		Rel. error [%]	
Unknowns	direct sym.	FMM	direct sym.	FMM	direct sym.	FMM
4486	2628	3030	238	238	0.989	0.989
17926	33928	70378	384	625	0.252	0.245
71686	*(542575)*	453600	N/A	900	N/A	0.090

Our FMM algorithm is always faster than the precorrected-FMM for $p = 4$ and they are comparable even for $p = 3$. On the other hand, the FMM algorithm seems to need more memory. Note that in this case (low number of iterations), the FMM version is faster and uses less memory than the direct version even for moderately sized problems and the subquadratic time complexity shows nicely.

The only other published FMM implementation we have found for symmetric BEM is by Of et al. [11]. After correcting for their use of only 7-point integration rules (the degree of their spherical harmonic expansion was not reported), their timings are comparable to ours.

5.2 Three-sphere head models

The programs were run with $K_c = 300$, $L = 10$, $\lambda = 1.7$ and $\varepsilon = 10^{-6}$. The elapsed time, memory requirements, number of iterations and achieved accuracy are shown in Tables 4 and 5.

We observe that no FMM acceleration takes place for the smallest mesh ($N = 4486$). For the middle one ($N = 17926$), FMM brings a memory saving but is still slower. For the largest mesh, FMM enables us to produce a valid result, which could not be calculated by the direct method for lack of memory.

6 Discussion

6.1 Difficulty of the three sphere model

There are two basic reasons why the FMM performs much better for the single-sphere model compared to the three-sphere model: First, the conditioning of the system is one order of magnitude worse, because of the low conductivity of the middle layer corresponding to the skull. Therefore, many iterations are needed, which emphasizes the matrix-vector product evaluation time versus the assemble time. Second, because the three surfaces are close to

Table 5: The memory requirements to solve the forward problem for the three sphere head models as a function of the number of unknowns for the direct (no FMM) and FMM-accelerated symmetric BEM methods in MB. We put between parentheses the extrapolated value for the largest problem, that could not be solved by the direct method due to lack of memory.

Unknowns	direct sym. [MB]	FMM
4486	91	123
17926	1390	1106
71686	*(22229)*	4400

one another, there are many more elements to be treated locally and the subquadratic time complexity only manifests itself for very large problems. Nevertheless, the FMM enables us to solve middle-sized problems, which barely fit in memory for current computers.

6.2 Implementation

Having implemented several of the low-level routines in both Ocaml and C++, we can claim that the time overhead of using Ocaml is acceptable — less than 50% on the average. The memory overhead fluctuates between 25% $\sim$ 50%, depending on the garbage collector settings.

The most critical point in implementing the FMM turned out to be memory management. Most of the memory is used to store the precalculated local interactions. By timing and counting the most costly basic operations (Section 4.5), we can establish that there is little hope of accelerating the FMM for our problem significantly more, unless computationally more efficient expansions and their translations are established. Replacing spherical harmonics by pseudoparticles [30] or Cartesian polynomials [31] might be an alternative to consider.

7 Conclusion

We have developed a fast multipole method to accelerate the symmetric BEM for the forward MEG/EEG problem that is as accurate as the symmetric BEM with direct assembly. With increasing size of the problem it gets faster and uses less memory than the direct method.

1 Appendix: Integral operators

We recall here the four integral operators involved in the symmetric BEM, as defined in [5, 20, 21].

$$\begin{aligned}
\big(\mathcal{N}f\big)(\mathbf{r}) &= \int_{\partial\Omega} \partial_{\mathbf{n},\mathbf{n}'} G(\mathbf{r}-\mathbf{r}') f(\mathbf{r}')\,\mathrm{d}s(\mathbf{r}') \\
\big(\mathcal{D}f\big)(\mathbf{r}) &= \int_{\partial\Omega} \partial_{\mathbf{n}'} G(\mathbf{r}-\mathbf{r}') f(\mathbf{r}')\,\mathrm{d}s(\mathbf{r}') \\
\big(\mathcal{D}^* f\big)(\mathbf{r}) &= \int_{\partial\Omega} \partial_{\mathbf{n}} G(\mathbf{r}-\mathbf{r}') f(\mathbf{r}')\,\mathrm{d}s(\mathbf{r}') \\
\big(\mathcal{S}f\big)(\mathbf{r}) &= \int_{\partial\Omega} G(\mathbf{r}-\mathbf{r}') f(\mathbf{r}')\,\mathrm{d}s(\mathbf{r}')
\end{aligned}$$

If we restrict $\mathbf{r}$ to be on the surface Γ and use the notation Θ for $\partial\Omega$ the operators above define mappings from functions defined on Θ to functions defined on Γ. For example, for the operator $\mathcal{S}$ we have:

$$\big(\mathcal{S}_{\Gamma\,\Theta} f\big)(\mathbf{r}) = \int_{\Theta} G(\mathbf{r}-\mathbf{r}') f(\mathbf{r}')\,\mathrm{d}s(\mathbf{r}') \quad \mathbf{r}\in\Gamma$$

2 Appendix: Representation Theorem

The Boundary Element Method is based on the fundamental Representation Theorem. For details on its application to the EEG problem refer to [21].

Theorem 1 (Representation Theorem) *Let $\Omega \subseteq \mathbb{R}^3$ be a bounded open set with a regular boundary $\partial\Omega$. Let $u : (\mathbb{R}^3 \backslash \partial\Omega) \to \mathbb{R}$ be a harmonic function ($\Delta u = 0$ in $\mathbb{R}^3\backslash\partial\Omega$), satisfying the $\lim_{r\to\infty} r\,|u(\mathbf{r})| < \infty$ and $\lim_{r\to\infty} r\frac{\partial u}{\partial r}(\mathbf{r}) = 0$, and denote $p(\mathbf{r}) \stackrel{def}{=} \partial_{\mathbf{n}} u(\mathbf{r})$. Then*

$$\begin{aligned}
-p &= +\mathcal{N}[u] - \mathcal{D}^*[p] \qquad \text{for } \mathbf{r}\notin\partial\Omega \\
u &= -\mathcal{D}[u] + \mathcal{S}[p] \\
-p^{\pm} &= +\mathcal{N}[u] + \big(\pm\frac{\mathcal{I}}{2} - \mathcal{D}^*\big)[p] \qquad \text{for } \mathbf{r}\in\partial\Omega \\
u^{\pm} &= \big(\mp\frac{\mathcal{I}}{2} - \mathcal{D}\big)[u] + \mathcal{S}[p]
\end{aligned}$$

where $\mathcal{I}$ denotes the identity operator.

3 Appendix: Spherical harmonics

There are several definitions of spherical harmonics and Legendre polynomials differing mainly in normalization and sign conventions. Ours follows Epton and Dembart [8]. The

spherical harmonic $Y_n^m(\theta,\phi)$ is

$$Y_n^m(\theta,\phi) = \sqrt{\frac{(n-|m|)!}{(n+|m|)!}}(-1)^m P_n^{|m|}(\cos\theta)\, e^{im\phi}$$

where $P_n^m(x)$ are the associated Legendre polynomials

$$P_n^m(x) = \frac{1}{2^n n!}\frac{(n+m)!}{(n-m)!}(1-x^2)^{-m/2}\frac{d^{n-m}}{dx^{n-m}}(x^2-1)^n$$

Given a vector $\mathbf{x}$ with Cartesian coordinates (x,y,z), we define its spherical coordinates as (r,θ,ϕ) by

$$\mathbf{x} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} r\ \cos\phi\ \sin\theta \\ r\ \sin\phi\ \sin\theta \\ r\ \cos\theta \end{pmatrix}$$

The inner and outer spherical harmonics are

$$\begin{aligned} O_n^m(\mathbf{x}) &= \tfrac{(-1)^n i^{|m|}}{A_n^m}\tfrac{Y_n^m(\theta,\phi)}{r^{n+1}} \\ I_n^m(\mathbf{x}) &= i^{-|m|}A_n^m r^n Y_n^m(\theta,\phi) \\ \text{with}\quad A_n^m &= \tfrac{(-1)^n}{\sqrt{(n-m)!\,(n+m)!}} \end{aligned}$$

Acknowledgements

Sponsored by the Czech Ministry of Education under Project LN00B096.

References

[1] J. Phillips, R. Leahy, J. Mosher, and B. Timsari, "Imaging neural activity using MEG and EEG," *IEEE Engineering in Medicine and Biology Magazine*, vol. 16, no. 3, pp. 34–42, 1997.

[2] M. Hämäläinen, R. Hari, R. J. IImoniemi, J. Knuutila, and O. V. Lounasmaa, "Magnetoencephalography— theory, instrumentation, and applications to noninvasive studies of the working human brain," *Reviews of Modern Physics*, vol. 65, no. 2, pp. 413–497, Apr. 1993.

[3] H. Cheng, L. Greengard, and V. Rokhlin, "A fast adaptive multipole algorithm in three dimensions," *J. Comput. Phys.*, no. 155, pp. 468–498, 1999. [Online]. Available: http://amath.colorado.edu/courses/7400/2003fall/005/papers/ChengGreengardRokhlin.pdf

[4] M. Clerc, R. Keriven, O. Faugeras, J. Kybic, and T. Papadopoulo, "The fast multipole method for the direct E/MEG problem," in *Proceedings of ISBI*. Washington, D.C.: IEEE, NIH, July 2002. [Online]. Available: http://www.biomedicalimaging.org/

[5] J. Kybic, M. Clerc, T. Abboud, O. Faugeras, R. Keriven, and T. Papadopoulo, "A common formalism for the integral formulations of the forward EEG problem," *IEEE Transactions on Medical Imaging*, 2004, accepted for publication.

[6] M. Clerc, A. Dervieux, O. Faugeras, R. Keriven, J. Kybic, and T. Papadopoulo, "Comparison of BEM and FEM methods for the E/MEG problem," in *Proceedings of BIOMAG 2002*, Aug. 2002.

[7] R. K. Beatson and L. Greengard, "A short course on fast multipole methods," in *Wavelets, Multilevel Methods and Elliptic PDEs*, M. Ainsworth, J. Levesley, W. Light, and M. Marletta, Eds. Oxford University Press, 1997, pp. 1–37. [Online]. Available: http://www.math.canterbury.ac.nz/~mathrkb/pdfs/beatson+greengard/beatsongreengard.pdf

[8] M. A. Epton and B. Dembart, "Multipole translation theory for the three-dimensional Laplace and Helmholtz equations," *SIAM J. Sci. Comput.*, vol. 16, no. 4, pp. 865–897, July 1995.

[9] B. Dembart and E. Yip, "The accuracy of fast multipole methods for Maxwell's equations," *IEEE Comput. Sci. Eng.*, vol. 5, no. 3, pp. 48–56, 1998. [Online]. Available: http://dx.doi.org/10.1109/99.714593

[10] J. Rahola, "Experiments on iterative methods and the fast multipole method in electromagnetic scattering calculations," CERFACS, Tech. Rep. TR/PA/98/49, 98.

[11] G. Of, O. Steinbach, and W. L. Wendland, "A fast multipole boundary element method for the symmetric boundary integral formulation," in *Proceedings of IABEM*, Austin, TX, USA, 2002. [Online]. Available: http://cavity.ce.utexas.edu/iabem2002/fullpapers/of.pdf

[12] S. Tissari and J. Rahola, "A precorrected-FFT method to accelerate the solution of the forward problem in magnetoencephalography," *Phys. Med. Biol.*, no. 48, pp. 523–541, 2003.

[13] J. Sarvas, "Basic mathematical and electromagnetic concepts of the biomagnetic inverse problem," *Phys. Med. Biol.*, vol. 32, no. 1, pp. 11–22, 1987.

[14] O. Faugeras, F. Clément, R. Deriche, R. Keriven, T. Papadopoulo, J. Roberts, T. Viéville, F. Devernay, J. Gomes, G. Hermosillo, P. Kornprobst, and D. Lingrand, "The inverse EEG and MEG problems: The adjoint space approach I: The continuous case," INRIA, Tech. Rep. 3673, May 1999. [Online]. Available: http://www.inria.fr/RRRT/RR-3673.html

[15] J. C. de Munck, "A linear discretization of the volume conductor boundary integral equation using analytically integrated elements," *IEEE Trans. Biomed. Eng.*, vol. 39, no. 9, pp. 986–990, Sept. 1992.

[16] J. C. Mosher, R. B. Leahy, and P. S. Lewis, "EEG and MEG: Forward solutions for inverse methods," *IEEE Transactions on Biomedical Engineering*, vol. 46, no. 3, pp. 245–259, Mar. 1999.

[17] N. G. Gencer and I. O. Tanzer, "Forward problem solution of electromagnetic source imaging using a new BEM formulation with high-order elements," *Phys. Med. Biol.*, vol. 44, no. 9, pp. 2275–2287, 1999.

[18] M. S. Hämäläinen and J. Sarvas, "Realistic conductivity geometry model of the human head for interpretation of neuromagnetic data," *IEEE Trans. Biomed. Eng.*, vol. 36, no. 2, pp. 165–171, Feb. 1989.

[19] C. G. Bénar and J. Gotman, "Modeling of post-surgical brain and skull defects in the EEG inverse problem with the boundary element method," *Clinical Neurophysiology*, no. 113, pp. 48–56, 2002.

[20] J.-C. Nédélec, *Acoustic and Electromagnetic Equations.* Springer Verlag, 2001.

[21] J. Kybic, M. Clerc, T. Abboud, O. Faugeras, R. Keriven, and T. Papadopoulo, "Integral formulations for the EEG problem," INRIA, Tech. Rep. 4735, Feb. 2003. [Online]. Available: http://www-sop.inria.fr/rapports/sophia/RR-4735.html

[22] J. Kybic and M. Clerc, "Symmetric BEM and multiscale fast multipole method for the E/MEG problem," in *4th International Symposium on Noninvasive Functional Source Imaging within the human brain and heart*, Chieti, Sept. 2003.

[23] D. R. Wilton, S. M. Rao, A. W. Glisson, D. H. Schaubert, O. M. Al-Bundak, and C. M. Butler, "Potential integrals for uniform and linear source distributions on polygonal and polyhedral domains," *IEEE Trans. Antenn. Propag.*, vol. 32, no. 3, pp. 276–281, Mar. 1984.

[24] A. S. Ferguson, X. Zhang, and G. Stroink, "A complete linear discretization for calculating the magnetic field using the boundary element method," *IEEE Trans. Biomed. Eng.*, vol. 41, no. 5, pp. 455–459, May 1994.

[25] J. N. Lyness and D. Jespersen, "Moderate degree symmetric quadrature rules for the triangle," *J. Inst. Maths Applics*, vol. 15, pp. 19–32, 1975.

[26] R. Cools, D. Laurie, and L. Pluym, "Algorithm 764: Cubpack++ — a C++ package for automatic two-dimensional cubature," *ACM Trans. Math. Software*, no. 23, pp. 1–15, 1997. [Online]. Available: http://www.cs.kuleuven.ac.be/~nines/research/CUBPACK/

[27] S. Tissari and J. Rahola, "Error analysis of a new Galerkin method to solve the forward problem in MEG and EEG using the boundary element method," CERFACS, Tech. Rep. TR/PA/98/39, 1998, toulouse, France.

[28] G. Fischer, B. Tilg, R. Modre, F. Hanser, B. Messnarz, and P. Wach, "On modeling the Wilson terminal in the Boundary and Finite Element Method," *IEEE Trans. Biomed. Eng.*, vol. 49, no. 3, pp. 217–224, Mar. 2002.

[29] R. Barret, M. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. van der Vonst, *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods.* Philadelphia: SIAM, 1994, available from netlib.

[30] J. Makino, "Yet another fast multipole method without multipoles — pseudoparticle multipole method," *J. Comput. Phys.*, vol. 151, no. 2, pp. 910–920, 1999, academic Press Professional, Inc.

[31] D. Apalkov and P. Visscher, "Fast multipole method for micromagnetic simulation of periodic systems," *IEEE Transactions on Magnetics*, vol. 39, no. 6, pp. 3478–3480, 2003.



Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399