BEM-FMM EEG/MEG Forward Problem Modeling Toolkit (Module) v. 2.3
Description

September 2, 2020

Contents

The toolkit is intended for academic use only. The software platform is MATLAB 2019a or newer (Windows/Linux). While the Windows implementation is stable and fast, the Linux implementation of the method may require extra recompilation of the FMM distributables ([4]) as described in the FMM software manual.

1. Quick start sequence for code evaluation (Windows, MATLAB)

Watch the corresponding video trailer first.

Download the software – folder `EEGMEG_Package`. Open MATLAB. From the main folder, switch the working directory to the `Model` subfolder.

- Run `model01_main_script.m` to perform necessary model precomputations.

Switch to the main folder, then run all the scripts of the main folder strictly sequentially to analyze all fields in the default example:

- Run `bem0_load_model.m`
- Run `bem1_setup_dipoles.m`. Observe and close two figures
- Run `bem2_charge_engine.m`. Observe and close the figure
- Run `bem3_surface_field_b.m`. Observe and close the figure
- Run `bem3_surface_field_c.m`. Observe and close the figure
- Run `bem3_surface_field_p.m`. Observe and close the figure
- Run `bem4_define_planes.m`. Observe and close the figures
- Run `bem5_volume_b_XY/XZ/YZ.m`. Observe and close all figures
- Run `bem5_volume_e_XY/XZ/YZ.m`. Observe and close all figures
- Run `bem5_volume_p_XY/XZ/YZ.m`. Observe and close all figures

2. Use, System Requirements, and Third-Party Components

The toolkit is intended for academic use only. The default software platform is MATLAB 2019a/b or newer (Windows/Linux). The toolkit runs "as is" under Windows and does not require any extra compilation. While the Windows implementation is stable and relatively fast (but not yet optimized for maximum speed), the Linux implementation of the method may require extra recompilation of the FMM distributables (Gimbutas et al., 2019) as described in the corresponding software manual (Gimbutas et al., 2019).

The following toolboxes (usually supplied with the MATLAB Academic Package) are required: Image Processing Toolbox (for NIfTI data processing if necessary), Partial Differential Equations or Antenna Toolbox (for model remeshing if necessary), and Statistics and Machine Learning Toolbox (for geometrical search of nearest neighbors used in the volumetric field plots). Those toolboxes are not absolutely necessary, but the toolkit must be modified to operate without them, and its performance will somewhat degrade. The FMM engine (Gimbutas et al., 2019) and example setups with SimNIBS segmentation (Saturnino et al 2019) of Human Connectome Project subjects 101309, 110411, 117122, 120111, 122317, 122620, 124422, 128632, 130013, 131722, 138534, 149337, 149539, 151627, 160123, and 198451 (Van Essen et al 2012-2019), as well as example setups with the SimNIBS Ernie model (Thielscher et al 2015), have been included with permission in the redistributable software package.

3. Toolkit Organization

The toolkit contains a number of short MATLAB or MATLAB-compatible scripts organized within two subfolders – `Model` and `Engine` – and a number of scripts located in the main folder, as shown in Fig. 1. The folders are organized as follows:

1. The main folder contains all major computational scripts which define cortical dipole assembly, perform computations, and output fields and potential both on surfaces and in volume. If NIfTI data are available, surface meshes and fields can be registered against NIfTI slices using the built-in NIfTI viewer.
2. The subfolder `Model` contains the head model that will be used for analysis. It also contains tools for remeshing (coarsening or refining) the head model and for performing necessary precomputations, such as double potential integrals for neighbor facets.
3. The subfolder `Engine` contains computational scripts and functions serving different purposes, including the BEM-FMM engine.

All scripts can be changed/modified and rearranged to organize parametric loops if necessary. The scripts of the main folder can be executed at any time for the default configuration.

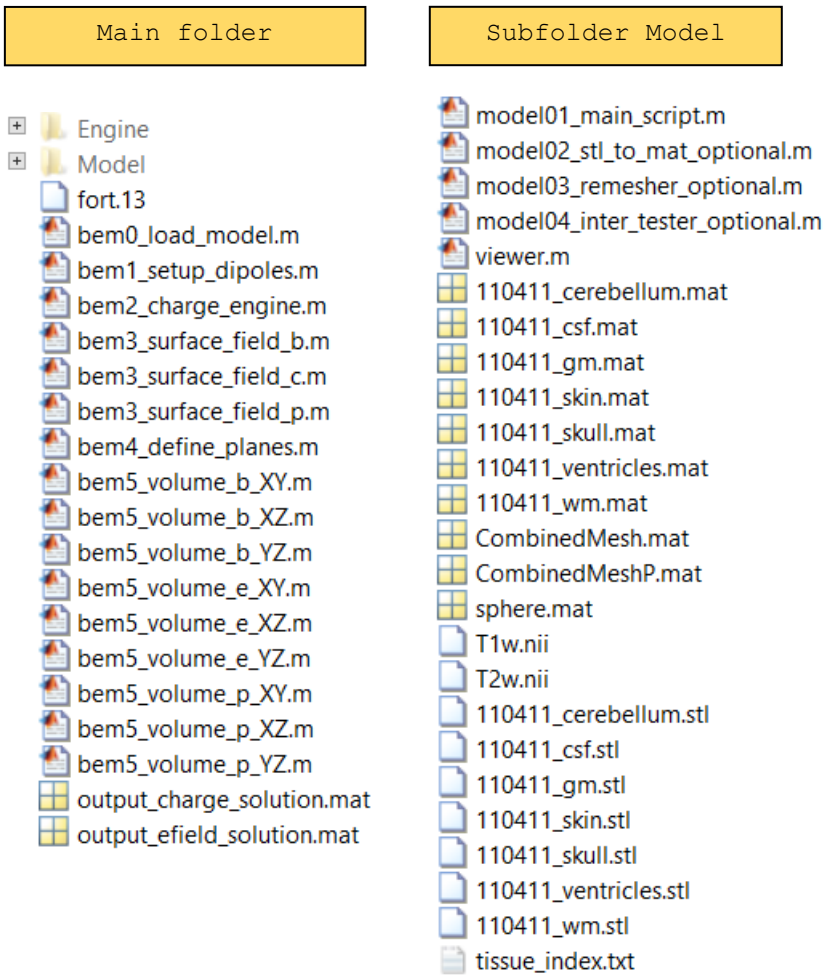

Fig. 1. Low-level organization chart of the toolkit.

4. Computational Workflow Overview (main folder)

The computations are performed in the main folder. The scripts of the main folder should initially be executed sequentially but may subsequently be executed in any order since all the data have already been computed.

The first script, `bem0_load_model.m`, imports head model data into the MATLAB workspace and sets the MATLAB path. It also imports the previously computed solution, if available.

The second script `bem1_setup_dipoles.m` initializes a group of closely-spaced dipoles – an equivalent cortical dipole layer. The dipole density may be varied.

BEM-FMM engine. The next script to be executed is `bem2_charge_engine.m`. This script

(i) computes the primary (or "incident") field of the cortical dipole(s) on every head interface (face nodes) using the FMM with facet subdivision;

(ii) computes the iterative solution of the BEM integral equation for the induced surface charge density using the FMM, precomputed near-field potential integrals, and MATLAB GMRES (generalized minimum-residual method, Saad 2003); and

(iii) displays the time for every iteration step in the MATLAB command window and plots the entire convergence history when completed.

As for the near-field integration accuracy, the double potential integrals for three neighbor triangular patches (default value is given in the script `model01_main_script.m`) are computed precisely using the sild-angle approach. For non-neighbor triangles, the center-point approximation is used for the double potential integrals and FMM. The number of neighbor integrals can be increased at the expense of larger memory usage.

Surface charge averaging (optional). For practical purposes, it might be convenient to introduce weighted surface charge averaging (i.e., to low pass filter the surface charge density). One option is to average over the target facet and its three immediate topological neighbor triangles. After the solution is obtained, we might substitute in the script `bem2_charge_engine.m`

```
c = (c.*Area + sum(c(tneighbor).*Area(tneighbor), 2))./(Area + sum(Area(tneighbor), 2));
```

This rule can be modified if necessary.

When performing mathematical (FMM) operations, the scripts of this folder call original and derived FMM functions from the subfolder `Engine`.

Fields at interfaces. After the computations have been completed, the scripts `bem3_surface_field_b/c/p.m` display the magnetic field, charge distribution, and electric potential/voltage at the interfaces. `bem3_surface_field_b` shows the magnetic field as measured by a "magnetometer" surface located at a user-defined distance from the skin.

Comparisons with analytical solutions. Such a comparison is performed in two separate folders: `Version 2.3_example1_hom_sphere` and `Version 2.3_example2_4layer_sphere`. There, the scripts `bem4_comparison_p_hom.m` performs comparisons with the analytical solution for a homogeneous sphere. The scripts `bem4_comparison_b/p_inhom.m` compare with the analytical solutions for a four-layer sphere. All analytical solutions have been programmed and tested in MATLAB.

The script `bem4_comparison_p_hom.m` computes the electric potential of a finite-length vertical dipole in a homogeneous sphere. We have used Frank's analytical solution (Frank 1952) here.

The script `bem4_comparison_b_inhom.m` computes the magnetic field of an arbitrary point dipole in an arbitrary multilayer sphere following the known analytical solution (Sarvas 1987). For a vertical dipole, the magnetic field outside the sphere vanishes, no matter how complicated the multilayer sphere composition is.

The script `bem4_comparison_p_inhom.m` computes the electric potential of an arbitrary point dipole in a four-layer sphere. We have used the analytical solution of Zhang and Mosher (Zhang 1995, Mosher et al., 1999). Note that in a review (Mosher et al., 1999), r should replace r_q in Eq. (17).

Surface data precomputation and NIfTI data import. The next script is `bem4_define_planes.m`. This script defines three principal observation planes and prepares mesh cross-sections that will be used in the field output plots.

If NIfTI data are available (e.g. `T1w.nii`), they will be included in subsequent visualizations which will superimpose mesh cross-sections and/or fields onto the corresponding NIfTI slices.

Volumetric fields in principal planes. The scripts `bem5_volume_b/e/p_XY/XZ/YZ.m` compute and output the magnetic field (any of its Cartesian components or a magnitude), the electric field (any of its Cartesian components or a magnitude), and the electric potential in the three principal planes. The plane position and its size are specified in the script `bem4_define_planes.m`.

The volumetric field computations require more time since the potential integrals are no longer precomputed and must be calculated at the time of execution, depending on the position of a given observation point relative to the nearest interface(s). The critical numerical parameter here is the dimensionless (vs. average triangle size) radius, R, of an integration sphere within which integration of the surface charge density is performed. Its default value ranges between 2 and 5; higher numbers (e.g., $R = 10$) may provide better field accuracy but simultaneously slow down the computations.

5. Head Model Import and Processing Overview (subfolder `Model`)

Two acceptable model formats. The head model files should always be located in the dedicated folder `Model` with contents shown in Fig. 1. The primary set are *.stl (stereolithography) files for every individual brain compartment in the form of a faceted shell. The *.stl files use triangular facets with normal vectors facing out of the shell. This is the standard output of the SimNIBS segmentation pipeline and other relevant software packages. The number of shells may be arbitrary. The script `model02_stl_to_mat_optional.m` converts *.stl files, either binary or ASCII, to equivalent MATLAB data files (using MATLAB's built-in function `stlread` or otherwise) containing arrays of vertices P, facets t, and normal vectors $\boldsymbol{n}$. Every MATLAB data file can further be inspected and visualized using the function `viewer.m` from the same subfolder (as shown in Fig. 2a below). Repeat this last operation for every brain compartment (or spherical shell) in the folder.

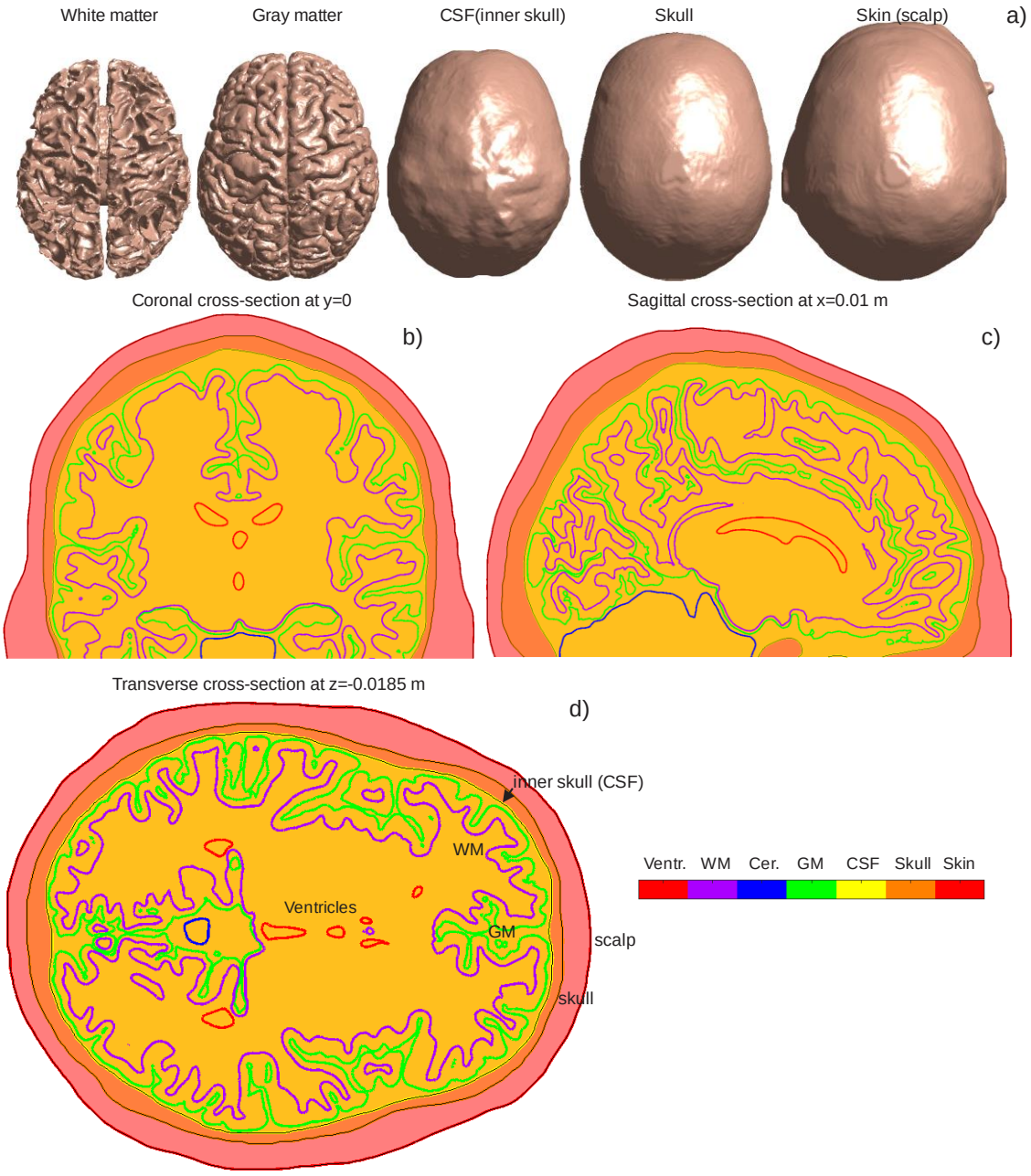

Fig. 2. a): Brain compartments of the default package head model # 110411: white matter (WM), gray matter (GM), cerebrospinal fluid (CSF or inner skull), skull, and skin. Cerebellum and ventricles are not shown. b-d): Head cross-sections in three principal planes.

Built in head models. For computational studies that do not involve MRI data collection, the present package is augmented with 16 realistic head models for 16 Connectome Project (Van Essen et al., 2012-2019) subjects with isotropic voxel resolution of 0.7 mm. These are subjects #101309, 110411, 117122, 120111, 122317, 122620, 124422, 128632, 130013, 131722, 138534, 149337, 149539, 151627, 160123, and 198451. The datasets have been converted to surface models with the help of the SimNIBS 2.1 pipeline; every model includes seven brain

compartments (skin, skull, CSF or cerebrospinal fluid, GM or gray matter, WM or white matter, ventricles, cerebellum). Every model has been checked and confirmed against the original NIfTI images and with regard to mesh manifoldness (Htet et al., 2019b). The default average cortical surface mesh edge length is 1.4 mm, the cortical nodal density is 0.55 nodes per mm^2, and the total number of facets is 0.9 M.

In addition to the Connectome Project head models, the package also includes the default example model of the SimNIBS 2.1 pipeline, the Ernie model. This model is comparable in complexity to the Connectome models, with 0.9 M facets and seven tissue meshes.

Any other surface model obtained from the SimNIBS pipeline may be used in *.stl or *.mat (MATLAB) format. In particular, fifty CAD models, known as the Population Head Model Repository or PHM (Lee et al 2016, Lee et al 2018), have been made available from the website of the IT'IS Foundation, Switzerland (IT'IS Foundation 2016).

Alternatively, the models described in detail in (Htet et al., 2019b) may be downloaded independently from the MATLAB Central link (Collection of Sixteen High-Quality Human Head CAD Models, 2019). The default head geometry in the folder `Model` is subject 110411 with the following seven 2-manifold watertight enclosed brain compartments: white matter (WM), gray matter (GM), cerebrospinal fluid (CSF or inner skull), skull, skin, cerebellum, and ventricles. These brain compartments, with the exception of the cerebellum and ventricles, are shown in Fig. 2.

Processing NIfTI Data. NIfTI data (if available) should be located in the same subfolder `Model` as shown in Fig. 1. For example, the Connectome Project database contains T1 and T2 NIfTI data for every subject, which were made available with permission. The default application example uses subject 110411.

Model remeshing. A CM2 SurfRemesh® remeshing program from Computing Objects, France is included in the MATLAB package. This software enables creation of coarser and/or finer surface representations while minimizing the surface deviation error from the master mesh. MATLAB script `model03_remesher_optional.m` performs automated remeshing to any required maximum edge length, which should be given at the beginning of the script.

For example, the remeshing program generates a coarser model with the average cortical edge length of 1.9 mm and the average cortical nodal density of 0.32 nodes per mm^2 when the maximum edge length is chosen as 3 mm; the total number of facets is 0.4 M. On the other hand, the same program generates a finer model with the average cortical edge length of 0.99 mm and average cortical nodal density of 1.2 nodes per mm^2 when the maximum edge length is chosen as 1 mm; the total number of facets is then 1.8 M. Fig. 3 shows the corresponding surface meshes for the gray matter shell along with the original segmentation. The red circles label sample areas of interest close to the precentral gyrus crown. Note that the remeshing procedure may require significant time. It may also fail in some cases.

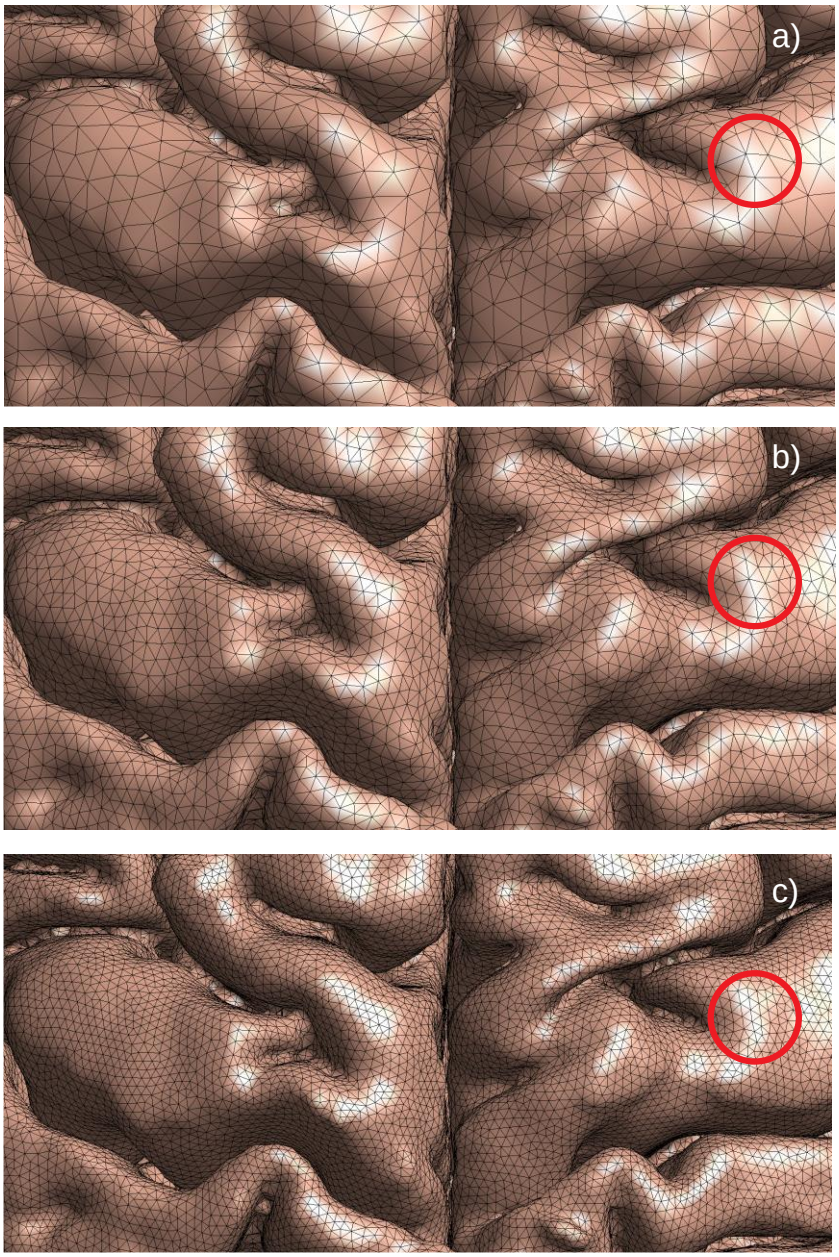

Fig. 3. a) – Coarser model with average cortical edge length of 1.9 mm and average cortical nodal density of 0.32 nodes per mm^2; b) – default meshing with average cortical surface mesh edge length of 1.5 mm and average cortical nodal density of 0.55 nodes per mm^2; c) – finer model with average cortical edge length of 0.99 mm and average cortical nodal density of 1.2 nodes per mm^2. The red circle labels an area of interest close to the precentral gyrus crown.

Creating combined head mesh. The combined mesh for the entire head is created by appending individual meshes. This operation is performed by running the script `model01_main_script.m`. The combined mesh is stored in the MATLAB data file `CombinedMesh.mat`. An additional data file, `CombinedMeshP.mat`, is generated in the same folder. This file contains precomputed double surface electrostatic integrals over triangles necessary for accurate BEM-FMM

simulations. The minimum number of neighbors for accurate EEG/MEG computations is 16. The integrals are computed in parallel, using 20 cores by default. The `numThreads` variable of `model01_main_script.m` may be adjusted depending on the computer configuration. Run the script `model01_main_script.m` to generate the data.

Defining tissue properties. The script `model01_main_script.m` reads from an editable tissue index file (named `tissue_index.txt`) in the `Model` subfolder to determine which `*.mat` tissue files to assemble into the final model and what conductivity values should be assigned to each of those tissues. Each line of a tissue index file provides the following information: tissue name (for reference in subsequent scripts), tissue source file, tissue conductivity, and enclosing tissue. It then assigns initial conductivity information to each facet of each tissue: the facet's interior conductivity (in the opposite direction of the facet's normal vector), the facet's exterior conductivity (in the direction of the facet's normal vector), and the conductivity contrast across the facet.

Treating duplicated facets. This script also checks the combined mesh for duplicate facets and for facets whose centroids are too close to be treated with the BEM-FMM algorithm. For the Connectome models, there should be none of these complications, because tissues of these models surround and enclose each other without touching – they are hollow shells, where each shell segments a boundary between exactly two tissue types. Other models, however, do not follow this meshing scheme. For example, the interior and exterior boundaries of every tissue of the MIDA model (Iacono et al 2015) are explicitly segmented. This means that the MIDA model's white matter and gray matter, for example, both independently segment their mutual boundary, producing a large number of duplicate facets. These duplicate facets would produce singularities that invalidate simulation results, so they are resolved as follows.

For each pair of duplicate facets, one is designated the facet to be deleted, and the other is designated the facet to be kept. The outer conductivity of the facet to be kept is set equal to the inner conductivity of the facet to be deleted, and associated conductivity contrast information is updated for the facet to be kept. The facet to be deleted, and all associated information, is then removed from the model. Fig. 4 below illustrates the results of this operation.

Tissue intersection marker points. The script `model04_inter_tester.m` finds intersection points between selected tissue meshes and an arbitrary ray. These data may be useful for further processing. Run the script and observe the generated results. Change the line definition if desired.

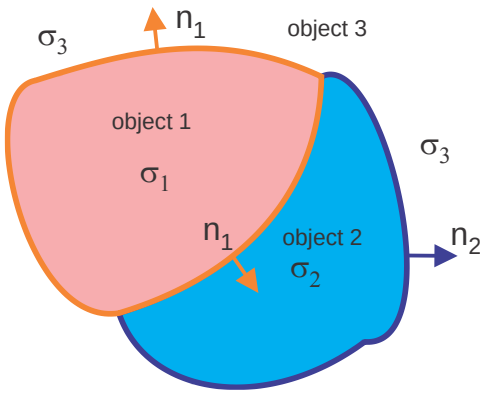

Fig. 4. Object 3 (with interior conductivity σ_3) surrounds and encloses both Object 1 (with interior conductivity σ_1) and Object 2 (with interior conductivity σ_2), so Object 1 and Object 2 initially list σ_3 as the exterior conductivity for all facets in their respective meshes. Because Object 1 and Object 2 have each explicitly segmented their mutual interface, that interface initially contains coincident facets contributed by both objects. In this example, Object 2's copies of the interface facets have been removed, and Object 1's copies of the facets remain. Object 1's facets at the interface still list σ_1 as their interior conductivity, but have changed their exterior conductivity from σ_3 to σ_2.

6. Control and Test of Numerical Accuracy

For a given surface mesh resolution, the numerical accuracy of the method is controlled by the following parameters:

1. `RnumberE/RnumberP`, found in `Model\model01_main_script.m`: Number of neighbor potential electrostatic double surface integrals (separately for electric field and electric potential) to be computed precisely. The default number varies between 16 and 128. The maximum number is unlimited, but is subject to memory constraints. Numbers above 128 do not affect the overall solution accuracy significantly.

2. `prec`, found in `Engine\bemf4_surface_field_electric.m` and in all other FFM implementations: Intrinsic FMM precision (Gimbutas et al., 2019). The default value is 0.01 (1%) which may be adequate for surface-based charge distributions. Values smaller than 0.001 do not affect the overall solution accuracy significantly.

3. `iter`, found in `bem2_charge_engine.m` in the main folder: Number of GMRES iterations used. The default value is 15. The maximum number is unlimited but is subject to speed constraints. Numbers above 20-30 do not affect the overall solution accuracy significantly.

4. `tneighbor`, found in `Model\model01_main_script.m`: Number of neighbor facets for optional averaging of the computed surface charge density after the solution had been obtained. The default number is 3 for manifold meshes.

5. `R`, found in `bem5_volume_b/e/p_XY/XZ/YZ.m` (in the main folder): Dimensionless (vs. average triangular face size) radius R of an integration sphere within which precise integration is

performed. In `bem5_volume_b/e/p_XY/XZ/YZ.m`, it is the integration of the induced surface charge density on nearby triangles given an observation point in close proximity to them. The default value ranges between 2 and 5. The maximum number is unlimited but is subject to speed constraints.

7. Application Example #1. Finite-Length Dipole in a Homogeneous Sphere

In this example (folder `Version 2.3_example1_hom_sphere`), some unnecessary scripts have been removed and some other scripts have been simplified. The major goal of the example is to become familiar with code functionality and execution flow. The main computational script `bem2_charge_engine.m` should execute in approximately 2 seconds (using a 2.4 GHz multicore server).

Assigning sphere conductivity and computing potential integrals. There is only one "tissue" mesh in this folder (subfolder `Model`): a sphere named `meshsphere6_92.mat`. It has a radius of 92 mm and approximately 50,000 facets. The average triangle quality (twice the ratio of the inradius to the circumradius) is 0.75. The average edge length (mesh resolution) is 2.3 mm. Run `viewer.m` and inspect the sphere mesh and its properties.

The assigned sphere conductivity value is 0.43 S/m. This value is given in the editable tissue index file (named `tissue_index.txt`) in the same subfolder `Model`.

Now, run the script `model01_main_script.m`. Reduce the number of cores for parallel computations (the `numThreads` variable) if necessary. This script has to be executed only *once*.

Defining dipole parameters. After that, go to the main folder of the example. Run the first script, `bem0_load_model.m`. Next, open and run the second script `bem1_setup_dipole.m`, which initializes a single vertical cortical dipole with a finite length of 2 mm and with a source/sink current of 1 μA:

```
%%   Single-dipole example
I0 = 1e-6;              %   source current, A
%  Vertical finite-length dipole (2 mm length)
strdipolePplus      = [0.000 0.000 0.076];    %   in m
strdipolePminus     = [0.000 0.000 0.074];    %   in m
```

Running simulations. The next script to be executed is `bem2_charge_engine.m`. This script

(i) computes the primary (or "incident") field of the cortical dipole on the sphere surface using the FMM;

(ii) computes the iterative solution of the BEM integral equation for the induced surface charge density using the FMM, precomputed near-field potential integrals, and MATLAB GMRES (generalized minimum-residual method); and

(iii) displays the time for every iteration step in the MATLAB command window and plots the entire convergence history when completed.

Visualizing surface fields. The next scripts to execute are `bem3_surface_field_c/p.m`. These scripts display the surface charge distribution and the continuous surface electric potential.

Comparing analytical and numerical solutions. The next script to execute is `bem4_comparison_p_hom.m`. This script computes an analytical solution for the electric potential of a finite-length vertical dipole in the homogeneous sphere and then compares it with the numerical result. We use Frank's analytical solution (Frank 1952) here. This analytical solution is not yet optimized; it executes very slowly (around 10 min on a multicore server). Therefore, the default analytical solution has been precomputed for convenience.

For the present example, the script generates two metrics of error for the electric potential on the sphere surface: the relative 2-norm error and the relative difference measure (RDM) following the definition given in (Engwer et al 2017), i.e.

$$Error(\varphi_{num}, \varphi_{analyt}) = \|\varphi_{num} - \varphi_{analyt}\| / \|\varphi_{analyt}\| \quad (1)$$

$$RDM(\varphi_{num}, \varphi_{analyt}) = 100\% \times \left\| \frac{\varphi_{num}}{\|\varphi_{num}\|} - \frac{\varphi_{analyt}}{\|\varphi_{analyt}\|} \right\| \quad (2)$$

Both errors are below **0.1%**. They are **0.032%** and **0.030%**, respectively. This result is to be expected since the dipole separation distance from the surface (17 mm) significantly exceeds the mesh resolution (average edge length) of 2.3 mm. In this case, the numerical solution is very accurate.

Visualizing volumetric fields. Run the script `bem4_define_planes.m` next. The following scripts `bem5_volume_b/e/p_XY/XZ/YZ.m` will plot the volumetric fields.

The three scripts for the electric field have an extra plotting option: to plot the field *only* within the sphere but not outside. This option is turned *on* by default. To plot the electric fields both inside and outside the sphere, set variable `in` on line 52 to a vector of logical 1's.

8. Application Example #2. Point and Finite-Length Dipoles in a Multi-Layer Sphere

In this example (folder `Version 2.3_example1_hom_sphere`), the code functionality will also be tested. The goal of this example is twofold. First, we estimate code accuracy in a more realistic scenario of a multi-layer sphere with ~0.24 M facets. Second, we estimate code and method accuracy for a challenging yet rather typical situation when the dipole distance to the nearest head compartment mesh (gray or white matter) is approximately *equal* to the mesh resolution (average edge length). The main computational script `bem2_charge_engine.m` should execute in approximately 20-30 sec (using a 2.4 GHz multicore server). for 1-5000 dipoles. However, for a large number of dipoles, the dipole plotting graphics in MATLAB becomes very slow. It should be therefore turned off.

Assigning conductivities of spherical compartments, assembling combined geometry, and computing potential integrals. There are six "tissue" meshes in this folder (subfolder `Model`):

- five spheres `meshsphere6_92/86/80/78/73.mat`. These spheres will represent skin, skull, CSF, GM (gray matter), and WM (white matter). They have radii of 92, 86, 80, 78, and 73 mm. Every sphere has approximately 50,000 facets; the average triangle quality (twice the ratio of the inradius to the circumradius) is 0.75. The average edge length (mesh resolution) of the combined model with five spheres is 2.0 mm.
- one sphere mesh `meshsphereo_97.mat` with the radius of 97 mm, which will serve as an observation surface for the magnetic field.

The assigned sphere conductivity values are 0.43, 0.01, 1.79, 0.33, and 0.33 S/m. These values are again given in the editable tissue index file (`tissue_index.txt`) in the same subfolder `Model`. Note that the last two conductivities are made equal to each other. This means that the smallest (WM) sphere has a conductivity contrast of zero, and thus does not influence the solution. The entire model, along with the assigned conductivity values, thus becomes equivalent to a *four-layer sphere model* considered in prior relevant and comprehensive studies (Engwer et al 2017, Piastra et al 2018).

Now, run the script `model01_main_script.m`, which combines the five meshes together, and computes the necessary potential integrals. Reduce the number of cores for parallel computations (the `numThreads` variable) if necessary. This script has to be executed only *once*.

Defining dipole parameters. Next, go to the main folder of the example. Run the first script, `bem0_load_model.m`. After that, open and run the second script `bem1_setup_dipole.m`, which by default initializes a single horizontal cortical dipole of a very short yet finite length of 0.04 mm and with a source/sink current of 1 μA:

```
%%   Single-dipole example
I0 = 1e-6;              %   source current, A

%  Horizontal short dipole (0.04 mm length)
strdipolePplus      = [+0.00002 0.000 0.0755];  %   in m
strdipolePminus     = [-0.00002 0.000 0.0755];  %   in m
```

Given this very short length, we treat this dipole as a point dipole. True point dipoles (with infinitesimally short length) cannot be used by the program. The geometry of the problem is shown in Fig. 5 below. The script `bem1_setup_dipole.m` provides additional figures displaying the dipole position in three-dimensional space and surrounding meshes.

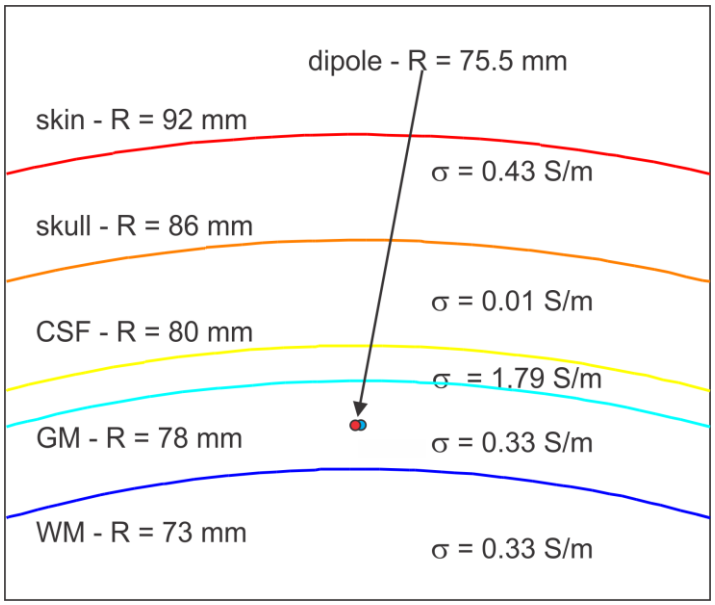

Fig. 5. Problem geometry and dipole position. Since the innermost sphere ("white matter") has a conductivity contrast of zero, a four-layer sphere geometry is effectively considered.

Running simulations. The next script to be executed is `bem2_charge_engine.m`. This script

(i) computes the primary (or "incident") field of the cortical dipole on all surfaces of the multi-layer model using the FMM;

(ii) computes the iterative solution of the BEM integral equation for the induced surface charge density using the FMM, precomputed near-field potential integrals, and MATLAB GMRES (generalized minimum-residual method); and

(iii) displays the time for every iteration step in the MATLAB command window, and plots the entire convergence history when completed.

Visualizing surface fields. The next scripts to execute are `bem3_surface_field_c/p.m`. These scripts display the surface charge distribution and the continuous surface electric potential for *any* of the model compartments. The compartment identifier is variable `tissue_to_plot`.

Comparing analytical and numerical solutions for magnetic field. The next script to execute is `bem4_comparison_b_inhom.m`. This script computes the magnetic field of the point dipole on the observation surface separated by 5 mm from the skin and compares it with the known analytical solution (Sarvas 1987).

The script generates two error metrics for the magnetic field on the observation sphere surface: the relative 2-norm error and the relative difference measure (RDM) following the definition given previously, i.e.

$$Error(\boldsymbol{B}_{num}, \boldsymbol{B}_{analyt}) = \|\boldsymbol{B}_{num} - \boldsymbol{B}_{analyt}\| / \|\boldsymbol{B}_{analyt}\| \quad (3)$$

$$RDM(\boldsymbol{B}_{num}, \boldsymbol{B}_{analyt}) = 100\% \times \left\| \frac{\boldsymbol{B}_{num}}{\|\boldsymbol{B}_{num}\|} - \frac{\boldsymbol{B}_{analyt}}{\|\boldsymbol{B}_{analyt}\|} \right\| \quad (4)$$

The errors are **4.0%** and **4.0%**, respectively. This result is to be expected since the dipole separation distance from the nearest GM surface (2.5 mm) now approaches the mesh resolution (average edge length of the model) of 2.0 mm. This circumstance increases the error.

A parametric loop could be organized to rotate the horizontal dipole about, say, the x-axis and find the average error. Such a loop with 1,000 dipole positions and the rotation angle changing from 0 to 360 degrees predicts similar average error values of **4.1%** in Eqs. (3) and (4), respectively.

Comparing analytical and numerical solutions for electric potential. The next script to execute is `bem4_comparison_p_inhom.m`. This script computes electric potential of the point dipole on the skin (outer sphere) surface and compares the solution with the analytical result of Zhang and Mosher (Zhang 1995, Mosher et al., 1999). Both errors given by Eqs. (1) and (2) appear to be **2.8%** and **2.4%**, respectively. The error increase is again to be expected since the dipole separation distance from the nearest GM surface (2.5 mm) approaches the mesh resolution (average edge length) of 2.0 mm.

A parametric loop could again be organized to rotate the horizontal dipole about, say, the x-axis and find the average error. Such a loop with 1,000 dipole positions and the rotation angle changing from 0 to 360 degrees predicts similar error values of **2.9%** and **2.4%**, respectively.

Visualizing volumetric fields. Run the script `bem4_define_planes.m` next. The following scripts `bem5_volume_b/e/p_XY/XZ/YZ.m` will plot the volumetric fields.

Replacing the horizontal dipole by a vertical dipole. As an exercise, you may return to the script `bem1_setup_dipole.m` and replace the horizontal dipole by a vertical dipole of the same strength, length, and position by uncommenting the lines

```
% Vertical short dipole (0.04 mm length)
strdipolePplus      = [0.000 0.000 0.07552];    %   in m
strdipolePminus     = [0.000 0.000 0.07548];    %   in m
```

and commenting the previous dipole definition. Then, repeat the above computation sequence for the horizontal dipole and observe how the results change. The script `bem4_comparison_b_inhom.m` will now generate `Inf/NaN` errors since the analytical solution predicts the magnetic field at the observation sphere to be exactly zero. On the other hand, the numerical solution does generate a certain nonzero magnetic field. This field is about **60 times weaker** than that for the equivalent horizontal dipole, but it still exists. This is a numerical error.

Next, the script `bem4_comparison_p_inhom.m` predicts the potential errors (1) and (2) of **2.8%** and **2.8%**, respectively, at the skin surface. This result is to be expected since the dipole separation distance from the nearest GM surface (2.5 mm) approaches the mesh resolution (average edge length) of 2.0 mm.

A parametric loop could be organized to rotate the vertical dipole about, for example, the x-axis and find the average error. Such a loop, with 1,000 dipole positions and rotation angle changing from 0 to 360 degrees, predicts somewhat smaller average error values of **2.9%** and **2.9%**, respectively.

The scripts `bem5_volume_b_XY/XZ/YZ.m` show the strength of the magnetic field blockage for the vertical dipole. The next group of scripts (for the electric field) show quite interesting behavior of the field in the skull volume. Finally, the electric potential in three principal planes could be examined.

Results for a cluster of vertical cortical dipoles. Now, run the script `bem1_setup_dipoles.m` and inspect the suggested problem geometry for a cluster of 20 dipoles between GM and WM sphere shells. Each dipole is 1 mm long. This geometry is obtained by creating a line that starts at the centroid of every GM facet of interest and continues in the direction of its inner normal vector. The positive pole of a dipole is located on this line at the distance of 2.5 mm from the facet center; the negative pole is located 1 mm further. All GM facets within an enclosing sphere could be chosen. The same method will be applied to create an equivalent cortical dipole layer for a realistic subject.

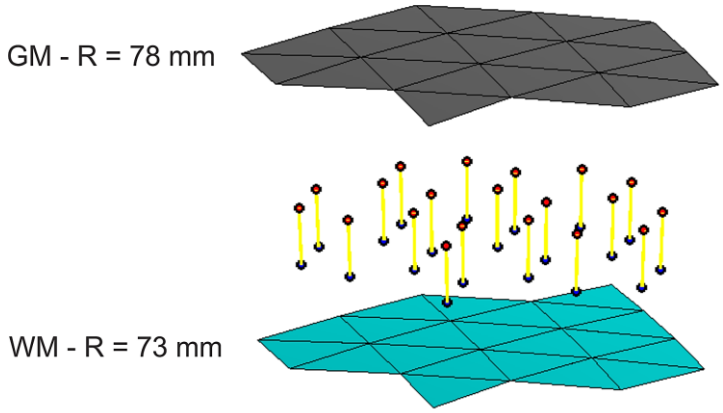

Fig. 6. Problem geometry for a cluster of 20 dipoles; each is 1 mm long.

Now, run the computational sequence again and go directly to the script `bem4_comparison_p_inhom.m`. You will see that the electric-potential error as compared to the analytical solution is actually relatively small. Despite the finite dipole length, it becomes **2.7%** and **2.6%** for Eqs. (1) and (2), respectively. Those values are even less than those for the single vertical point dipole (**2.8%** and **2.8%**) considered above! If the cluster of shorter dipoles (0.1 mm long) were considered, the error becomes **2.8%** and **2.7%** for Eqs. (1) and (2), respectively. This exercise underscores the fact that the error for the dipole cluster does *not* generally exceed the single-dipole error.

In the following field scripts, the scale may need to be adjusted for every figure to obtain the proper resolution.

9. Application Example #3. Equivalent Cortical Dipole Layer for Connectome Subject 110411

In this example, we consider a realistic head model – Connectome subject 110411 – with seven head compartments (Htet et al., 2019b). The default average cortical surface mesh edge length is approximately 1.4 mm, the cortical nodal density is 0.55 nodes per mm^2, and the total number of facets is 0.9 M. The main computational script `bem2_charge_engine.m` should execute in less than 1-2 min (using a 2.4 GHz multicore server) for 1-50,000 dipoles. However, for a large number of dipoles, the dipole plotting graphics in MATLAB becomes very slow. It should be turned off.

Assigning conductivities of spherical compartments, assembling combined geometry, and computing potential integrals. The corresponding example folder is `Version 2.3_example3_subject110411`. There are seven tissue meshes (skin, skull, CSF, GM, WM, cerebellum, ventricles) in this folder (subfolder `Model`); six of them are shown in Fig. 2. The meshes are inspected with the function `viewer.m` from the same subfolder.

The assigned conductivity values are the standard values used in the SimNIBS package (Saturnino et al 2019). These values are again given in the editable tissue index file (`tissue_index.txt`) in the same subfolder `Model`.

Now, run the script `model01_main_script.m`, which combines all partial meshes together and computes necessary potential integrals. Reduce the number of cores for parallel computations (the `numThreads` variable) if necessary. This script has to be executed only *once*.

Defining dipole parameters. Next, go to the main folder of the example. Run the first script, `bem0_load_model.m`. After that, open and run the second script `bem1_setup_dipoles.m`, which by default initializes a dipole cluster between WM and GM interfaces as shown in Fig. 7a,b. This geometry is obtained by creating a line that starts at the centroid of every GM facet of interest and continues in the direction of its inner normal vector. The positive pole of a dipole is located on this line at the distance of 1 mm from the facet center; the negative pole is located 1 mm further. All GM facets within an enclosing sphere are chosen. The corresponding code reads (every dipole has a source/sink current of 1 μA):

```
d  = 0.3e-3;            %   finite-dipole length, m (change to 1e-3)
s  = 2e-3;              %   spacing from GM, m
R  = 0.007;             %   radius of the enclosing sphere for the cluster in m
Ctr= 1e-3*[31 0 56];    %   position of enclosing sphere in m
```

It generates 468 dipoles within the sphere.

The individual dipole strength is computed/normalized to obtain the desired strength of a cortical equivalent dipole layer. For example, one can use the Okada-Murakami constant of $1\ \mathrm{nA \cdot m/mm^2}$ (Murakami and Okada 2006, Murakami and Okada 2015) to model a largely

invariant realistic current dipole density. When the dipole length is d and the cortical area per one dipole is A, this given us an expression for the dipole current I_0 in the form

$$\frac{I_0 d}{A} = 1 \text{ nanompere} \cdot \text{m/mm}^2 \rightarrow I_0 = 10^{-3}\frac{A}{d} \text{ [amperes]} \quad (5)$$

Running simulations. The next script to be executed is `bem2_charge_engine.m`. This script

(i) computes the primary (or "incident") field of the cortical dipoles on all surfaces of the multi=layer model using the FMM;

(ii) computes the iterative solution of the BEM integral equation for the induced surface charge density using the FMM, precomputed near-field potential integrals, and MATLAB GMRES (generalized minimum-residual method); and

(iii) displays the time for every iteration step in the MATLAB command window and plots the entire convergence history when completed.

Visualizing surface fields. The next scripts to execute are `bem3_surface_field_b/c/p.m`. These scripts display the magnetic field distribution, the surface charge distribution, and the continuous surface electric potential for *any* of the model compartments. The compartment identifier is variable `tissue_to_plot`.

Plotting volumetric fields. Magnetic and electric fields, and the electric potential are plotted exactly in the same way as described in Examples 1 and 2. The plane size and its position are given at the beginning of every script; the plane position is now linked to the center of the dipole cluster. The plot resolution is controlled by variable `Ms`; the field threshold (for graphics) is controlled by variable(s) `th`; the number of plot levels is controlled by variable `levels`.

References

[1] Collection of Sixteen High-Quality Human Head CAD Models. 2019. MATLAB Central Onl: https://www.mathworks.com/matlabcentral/fileexchange/69517-collection-of-sixteen-high-quality-human-head-cad-models

[2] Engwer C, Vorwerk J, Ludewig J, Wolters CH. A Discontinuous Galerkin Method to Solve the EEG Forward Problem Using the Subtraction Approach. *SIAM J. Sci. Comput*. 2017;39(1): B138–B164. doi. 10.1137/15M1048392.

[3] Frank E. Electric potential produced by two point current sources in a homogeneous conducting sphere. *J. Appl. Phys.* 1952;23:1225–28. https://doi.org/10.1063/1.1702037.

[4] Gimbutas Z, Greengard L, Magland J, Rachh M, Rokhlin V. fmm3D Documentation. Release 0.1.0. 2019. Online: https://github.com/flatironinstitute/FMM3D

[5] Gomez L, Dannhauer M, Koponen L, & Peterchev A.V. Conditions for numerically accurate TMS electric field simulation. *bioRxiv 505412* 2018b. doi: https://doi.org/10.1101/505412.

[6] Gomez LJ, Dannhauer M, Koponen LM, Peterchev AV. Conditions for numerically accurate TMS electric field simulation. Brain Stimul. 2019 Oct 3. pii: S1935-861X(19)30378-X. doi: 10.1016/j.brs.2019.09.015.

[7] Greengard L, Rokhlin V. A fast algorithm for particle simulations. *J. Comput. Phys.* 1987;73(2):325-348. doi: 10.1016/0021-9991(87)90140-9.

[8] Hasgall PA, Di Gennaro F, Baumgartner C, Neufeld E, Lloyd B, Gosselin MC, Payne D, Klingenböck A, Kuster N. *IT'IS Database for thermal and electromagnetic parameters of biological tissues*. Version 4.0, May 15, 2018. doi: 10.13099/VIP21000-04-0. Onl: www.itis.ethz.ch/database.

[9] Htet AT, Saturnino GB, Burnham EH, Noetscher G, Nummenmaa A, Makarov SN. Comparative performance of the finite element method and the boundary element fast multipole method for problems mimicking transcranial magnetic stimulation (TMS*). J Neural Eng*. 2019a:16:1-13. doi: https://dx.doi.org/10.1088/1741-2552/aafbb9.

[10] Htet AT, Burnham EH, Noetscher GM, Pham DN, Nummenmaa A, MakarovSN. Collection of CAD human head models for electromagnetic simulations and their applications. *Biomedical Physics & Engineering Express*. 2019b: 6(5):1-13. doi: https://doi.org/10.1088/2057-1976/ab4c76.

[11] Human Connectome Project. *S1200 Reference Manual. April 10 2018*. Onl: https://www.humanconnectome.org/storage/app/media/documentation/s1200/HCP_S1200_Release_Reference_Manual.pdf

[12] Iacono MI, Neufeld E, Akinnagbe E, Bower K, Wolf J, Oikonomidis I, Sharma D, Lloyd B, Wilm B, Wyss M, Pruessman K, Jakab A, Makris N, Cohen E, Kuster N, Kainz W, and Angelone LM. MIDA: A Multimodal Imaging-Based Detailed Anatomical Model of the Human Head and Neck. *PLoS One* 2015; 10(4). doi: 10.1371/journal.pone.0124126

[13] Lee E, Duffy W, Hadimani R, Waris M, Siddiqui W, Islam F, Rajamani M, Nathan R, Jiles D. Investigational Effect of Brain-Scalp Distance on the Efficacy of Transcranial Magnetic Stimulation Treatment in Depression. *IEEE Trans. Magn.*, 2016;52(7):1-4. doi. 10.1109/TMAG.2015.2514158.

[14] Lee EG, Rastogi P, Hadimani RL, Jiles DC, Camprodon JA. Impact of non-brain anatomy and coil orientation on inter- and intra-subject variability in TMS at midline. *Clin Neurophysiol.* 2018 Sep;129(9):1873-1883. doi: 10.1016/j.clinph.2018.04.749. Appendix A. Supplementary data.

[15] Makarov SN, Noetscher GM, Raij T, Nummenmaa A. A Quasi-Static Boundary Element Approach with Fast Multipole Acceleration for High-Resolution Bioelectromagnetic Models. *IEEE Trans. Biomed. Eng.* 2018;65(12):2675-2683. doi: 10.1109/TBME.2018.2813261.

[16] Murakami S, Okada Y. Contributions of principal neocortical neurons to magnetoencephalography and electroencephalography signals. *J Physiology*. 2006;575:925–936. doi: 10.1113/jphysiol.2006.105379. PMID: 16613883.

[17] Murakami S, Okada Y. Invariance in current dipole moment density across brain structures and species: Physiological constraint for neuroimaging. *NeuroImage*. 2015 May; 111:49–58. doi: 10.1016/j.neuroimage.2015.02.003. PMID: 25680520.

[18] Mosher JC, Leahy RM, Lewis PS. EEG and MEG: forward solutions for inverse methods. *IEEE Trans Biomed Eng.* 1999. Mar;46(3):245-59. PMID: 10097460.

[19] Piastra MC, Nüßing A, Vorwerk J, Bornfleth H, Oostenveld R, Engwer C, Wolters CH. The Discontinuous Galerkin Finite Element Method for Solving the MEG and the Combined MEG/EEG Forward Problem. *Frontiers in Neuroscience*. 2018;12(Article 30):1-18. doi: 10.3389/fnins.2018.00030

[20] Rokhlin V. Rapid Solution of Integral Equations of Classical Potential Theory. *J. Computational Physics*, 1985;60(2):187–207. doi: 10.1016/0021-9991(85)90002-6.

[21] Saad Y. *Iterative Methods for Sparse Linear Systems*. 2nd Ed., Society for Industrial and Applied Mathematics. 2003. ISBN 978-0-89871-534-7.

[22] Sarvas J. Basic mathematical and electromagnetic concepts of the biomagnetic inverse problem. *Phys. Med. Biol.* 1987; 32(1):11-22. PMID: 3823129.

[23] Saturnino GB, Puonti O, Nielsen JD, Antonenko D, Madsen KH, Thielscher A. SimNIBS 2.1: A Comprehensive Pipeline for Individualized Electric Field Modelling for Transcranial Brain Stimulation. In: Makarov S, Noetscher G, Horner M. Eds. *Brain and Human Body Modeling*. Springer Nature. NY 2019. ISBN 9783030212926.

[24] The Population Head Model Repository. 2016. IT'IS Foundation website. doi: 10.13099/ViP-PHM-V1.0. Online: https://www.itis.ethz.ch/virtual-population/regional-human-models/phm-repository/

[25] Thielscher A, Antunes A, and Saturnino GB. Field modeling for transcranial magnetic stimulation: a useful tool to understand the physiological effects of TMS? *IEEE EMBS 2015*, Milano, Italy.

[26] Van Essen DC, Ugurbil K., Auerbach E, Barch D, Behrens TE, Bucholz R, Chang A, Chen L, Corbetta M, Curtiss SW, Della Penna S, Feinberg D, Glasser MF, Harel N, Heath AC, Larson-Prior L, Marcus D, Michalareas G, Moeller S, Oostenveld R, Petersen SE, Prior F, Schlaggar BL, Smith SM, Snyder AZ, Xu J, Yacoub E. The Human Connectome Project: A data acquisition perspective. *NeuroImage*, 2012;62(4):2222–2231. doi: 10.1016/j.neuroimage.2012.02.018. PMID: 22366334. Online: http://www.humanconnectomeproject.org/

[27] Zhang Z. A fast method to compute surface potentials generated by dipoles within multilayer anisotropic spheres. *Phys. Med. Biol.* 1995 March;40(3):335–349. PMID: 7732066.