BEM-FMM TMS Modeling Toolkit v. 0.3 Description

September 5, 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 (Gimbutas et al 2019) 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 `TMS_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_coil.m`. Observe and close the figure
- 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_e.m`. Observe and close the figure
- Run `bem3_surface_field_l.m`. Observe and close the figure
- Run `bem3_surface_field_p.m`. Observe and close the figure
- Run `bem3_surface_field_the.m`. Observe and close the figure
- Run `bem3_surface_field_thl.m`. Observe and close the figure
- Run `bem4_define_planes.m`. Observe and close the figures
- Run `bem5_volume_XY/XZ/YZ.m`. Observe and close all three figures.

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

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 as described in the FMM 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), Partial Differential Equations or Antenna Toolbox (for model remeshing), and Statistics and Machine Learning Toolbox (for geometrical search of nearest neighbors used in the volumetric fields plots) Those toolboxes are not absolutely necessary, but the TMS 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) and the IT'IS Foundation's MIDA model (Iacono et al 2015), have been included with permission in the redistributable software package. A DropBox folder (DropBox 2020) contains the base code for Human Connectome Project subject 110411. An additional Dropbox folder contains data for all 18 head models referenced above.

Suggested citation: https://pubmed.ncbi.nlm.nih.gov/32235065/ ([16])

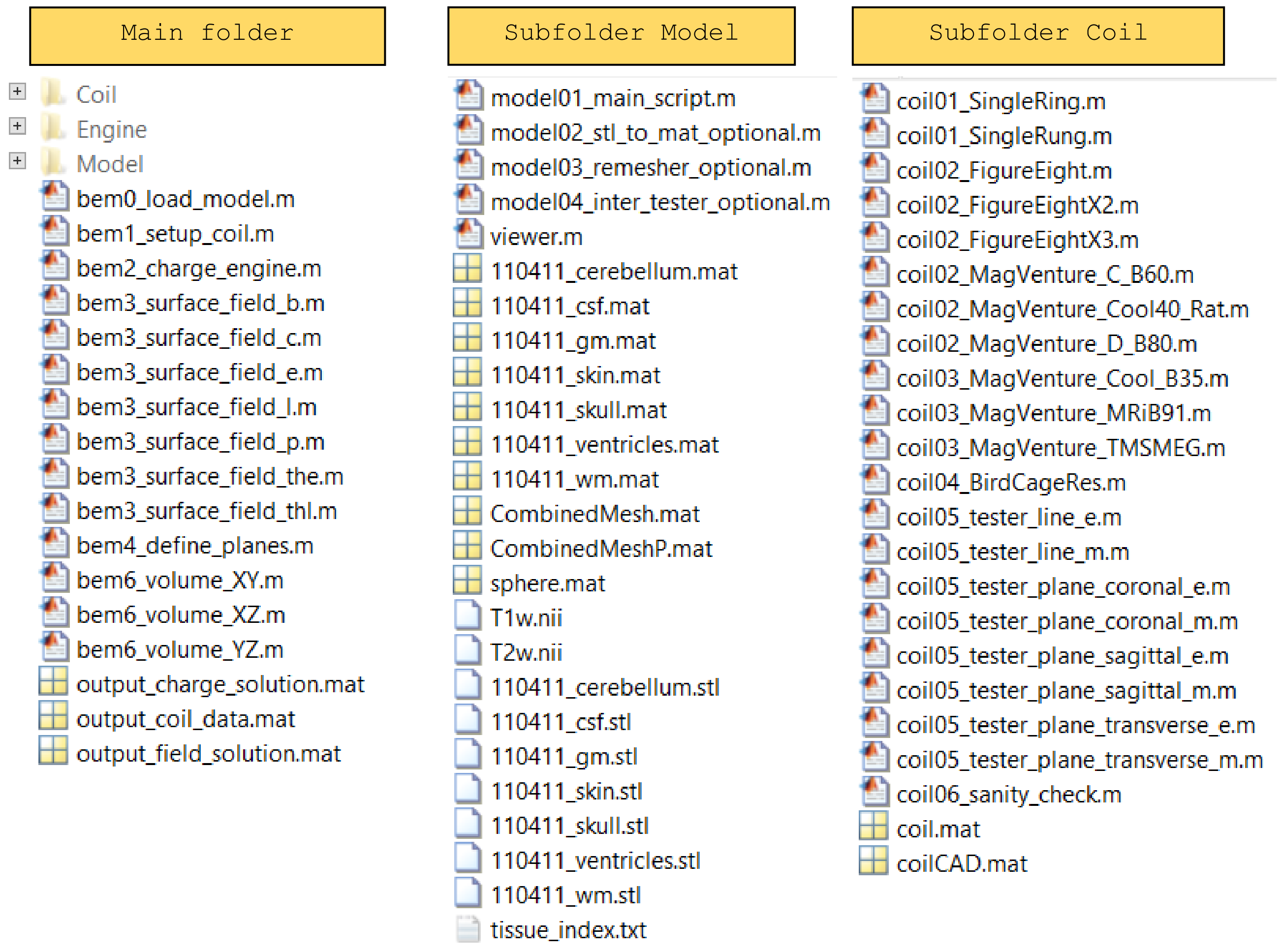
3. Toolkit Organization

A code version containing only Connectome Subject 110411 is available at a DropBox repository (DropBox 2020). For purposes of this walkthrough, we focus specifically on the toolkit containing Connectome Subject 110411.

The code contains a number of short MATLAB or MATLAB-compatible scripts organized within three subfolders – `Model`, `Coil`, and `Engine` – and a number of scripts located in the main folder, as shown in Fig. 1.

The main folder and three subfolders are organized as follows:

1. The main folder contains all major computational scripts which define coil/head position, perform computations, and output electric fields 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 `Coil` is devoted to coil definition, construction of the coil wire and CAD models, and, optionally, separate coil testing/optimization.
4. The subfolder `Engine` contains computational scripts and functions serving different purposes, including the BEM-FMM engine (from Aug. 2020).



Main folder

- Coil
- Engine
- Model
- bem0_load_model.m
- bem1_setup_coil.m
- bem2_charge_engine.m
- bem3_surface_field_b.m
- bem3_surface_field_c.m
- bem3_surface_field_e.m
- bem3_surface_field_l.m
- bem3_surface_field_p.m
- bem3_surface_field_the.m
- bem3_surface_field_thl.m
- bem4_define_planes.m
- bem6_volume_XY.m
- bem6_volume_XZ.m
- bem6_volume_YZ.m
- output_charge_solution.mat
- output_coil_data.mat
- output_field_solution.mat

Subfolder Model

- model01_main_script.m
- model02_stl_to_mat_optional.m
- model03_remesher_optional.m
- model04_inter_tester_optional.m
- viewer.m
- 110411_cerebellum.mat
- 110411_csf.mat
- 110411_gm.mat
- 110411_skin.mat
- 110411_skull.mat
- 110411_ventricles.mat
- 110411_wm.mat
- CombinedMesh.mat
- CombinedMeshP.mat
- sphere.mat
- T1w.nii
- T2w.nii
- 110411_cerebellum.stl
- 110411_csf.stl
- 110411_gm.stl
- 110411_skin.stl
- 110411_skull.stl
- 110411_ventricles.stl
- 110411_wm.stl
- tissue_index.txt

Subfolder Coil

- coil01_SingleRing.m
- coil01_SingleRung.m
- coil02_FigureEight.m
- coil02_FigureEightX2.m
- coil02_FigureEightX3.m
- coil02_MagVenture_C_B60.m
- coil02_MagVenture_Cool40_Rat.m
- coil02_MagVenture_D_B80.m
- coil03_MagVenture_Cool_B35.m
- coil03_MagVenture_MRiB91.m
- coil03_MagVenture_TMSMEG.m
- coil04_BirdCageRes.m
- coil05_tester_line_e.m
- coil05_tester_line_m.m
- coil05_tester_plane_coronal_e.m
- coil05_tester_plane_coronal_m.m
- coil05_tester_plane_sagittal_e.m
- coil05_tester_plane_sagittal_m.m
- coil05_tester_plane_transverse_e.m
- coil05_tester_plane_transverse_m.m
- coil06_sanity_check.m
- coil.mat
- coilCAD.mat

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

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.

4. Default Application Example

In the following computation example, we will consider Connectome subject model #110411 and the MRi-B91 TMS-MRI coil model targeting the hand area of the primary motor cortex, located above the precentral gyrus of the right hemisphere (the hand knob area). The tissue conductivity values are those of the SimNIBS TMS software package. Alternatively, some other sets (see, in particular Engwer et al., 2017 and Piastra et al., 2018) may be used. The coil is driven with a time-varying current of $\frac{dI}{dt} = 9.4e7\ Amperes/sec.$

5. Coil Selection and Analysis (subfolder `Coil`)

Coil selection (required). Start with subfolder `Coil`. Here, dedicated MATLAB scripts generate coil models (both wire and CAD), with *one* script per coil. The coil conductor centerline is defined manually using either an analytical formula or a set of points in three dimensions. After that, the corresponding script automatically generates the volumetric computational wire grid coil model and the coil CAD model. The default coil axis is the *z*-axis. Run scripts `coil01*` through `coil03*`. This generates several coil models, some of which are shown in Fig. 2.

Finally, run the script `coil03_MagVenture_MRiB91.m`. This will generate and save the MRI compatible TMS coil model MRi-B91 from MagVenture, Denmark shown in Fig. 2a. This model, centered at the origin of the Cartesian coordinate system, will be used in further analysis below.

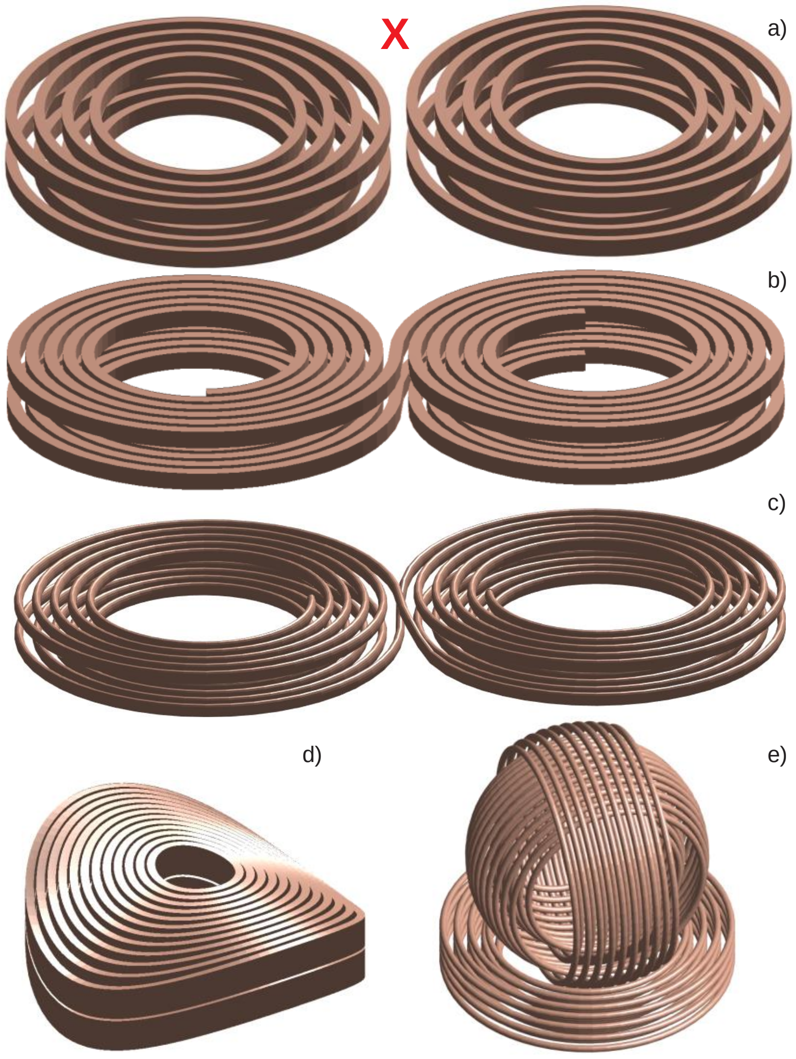


Fig. 2. Some solid CAD models created using the MATLAB-based coil geometry generator. Fig. 2a is a simplified MRi-B91 TMS-MRI coil model (MagVenture, Denmark) with elliptical conductors of a rectangular cross-section used in this example; Fig. 2b is a simplified MagPro C-B60 coil model (MagVenture, Denmark); Fig. 2c is a generic double figure-eight spiral coil model with an elliptical cross-section and two bootstrapped interconnections; Fig. 2d is a simplified Cool-40 Rat small animal coil model (MagVenture, Denmark); Fig. 2e is a three-axis multichannel TMS coil array radiator (Navarro de Lara et al, 2018). Note that the red “X” denotes the default coil model.

Examining fields of MRi-B91 coil in free space (optional). After the coil model MRi-B91 has been selected, a number of scripts in subfolder `Coil` will allow us to examine the coil's electric and magnetic fields. These are line plots (`coil05_tester_line_e/m.m`, Fig. 3a,b) as well as high-resolution 2D contour plots (`coil05_tester_plane_coronal_e/m.m` etc.) for any component of the electric and/or magnetic field in the coronal, sagittal, and transverse planes. The coil's time-varying current, dI/dt, must be specified at the beginning of every script. For the magnetic field, only the steady-state current, I_0, is required. Scripts `coil05_tester_plane_coronal_e/m.m` may be run to obtain Fig. 3c,d. Note that these scripts also define the observation plane window. This window will not be reused for head-coil computations of the main folder.

While performing mathematical (mesh generation, FMM computations) and graphical operations, these scripts call several functions from the subfolder `Engine`. A coil mesh generator script and a field computation script may be further combined into one script and augmented with a parametric loop to enable coil analysis and design (cf., for example, Makarov et al 2019).

***Changing coil geometry/optimizing coil fields (optional)*.** All geometry parameters are to be given in the respective coil scripts. This coil is constructed as many elliptical coaxial rings of a finite cross-section, where the ring axis is the *z*-axis. The script introduces the coil geometry by defining intersections of the conductor centerlines with the *xz*- and *yz*-planes, respectively. Parameters at the beginning of the script `coil03_MagVenture_MRiB91.m` define the conductor's characteristics. Rectangular (`flag 2`) and elliptical (`flag 1`) cross-sections are permitted.

Both the computational wire grid and the coil CAD model are generated by the function `meshcoil.m`. This function is specifically applicable to a particular coil geometry consisting of a (large) number of concentric loops; it creates the wire coil model all at once. The input are intersection points of the loop centerlines with *xz*- and *yz* planes. Either a Litz wire model (parameter `sk = 0`) or a skin-layer model (parameter `sk = 1`) may be used. In the former case, the current distribution across a conductor's cross-section is nearly uniform. In the latter case, the wire grid is situated close to the surface of the conductor. The density of the wire grid depends on the cross-section triangulation; it is controlled by parameter `M` – the number of cross-section subdivisions. The grid resolution in the direction of the conductor centerline is controlled by the original centerline discretization.

Function `meshwire.m` of subfolder `Engine` is more general than `meshcoil.m`. This function creates the wire mesh for an arbitrary *single* conductor. Either closed loops (`coil01_SingleRing.m`) or open conductors (`coil01_SingleRung.m`) may be generated. The computational wire grid coil model consists of straight, short, infinitely-thin current filaments or segments. The current filaments are defined as short straight lines joining centroids of triangles of the cross-sectional mesh, which are replicated along the conductor's centerline as many times as required. The cross-section is always perpendicular to the conductor's centerline, so the filaments are always parallel to the conductor's centerline.

In either case, the computational coil grid is the structure `strcoil` with the following fields:

```
Pwire – nodes of elementary wires inside the conductor
Ewire – edges (start & end points) of elementary wires inside the conductor
Swire - weights of elementary wire segments given total current of 1A
```

Weights are necessary to ensure that the total current through the conductor's cross-section is 1 A.

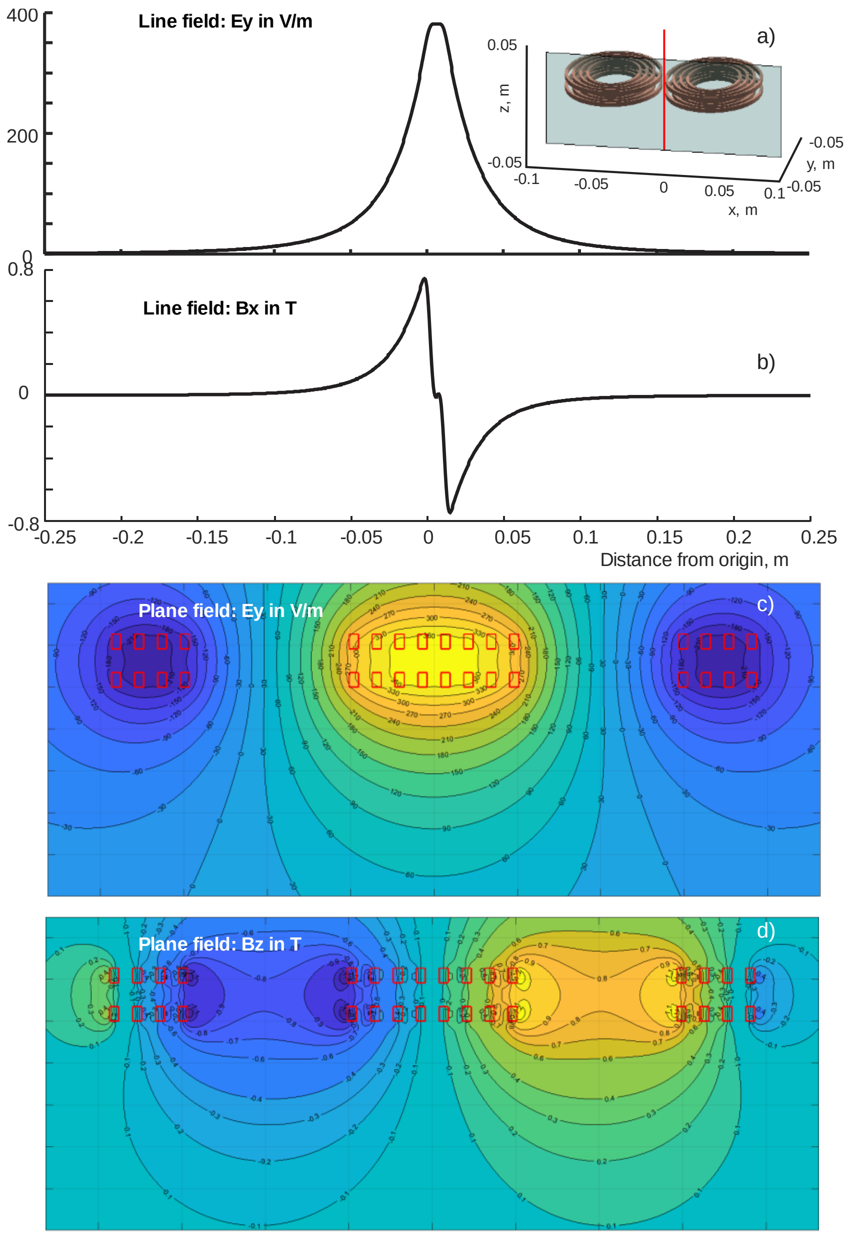


Fig. 3. Coil evaluation example. Electric (a,c) and magnetic (b,d) fields on a line or in a plane with 0.25 M observation points for the MRi-B91 TMS-MRI coil (MagVenture). Conductor cross-section is marked

in red. The coil is driven with a time-varying current of $\frac{dI}{dt} = 9.4e7\ Amperes/sec$. An equivalent definition would be a conductor current of 5 kA and a CW frequency of 3 kHz.

A solid CAD model (as opposed to a wire mesh model) for the coil conductor can be constructed using the function `meshsurface.m` of subfolder `Engine`. This script creates a structured triangular surface mesh (comprising array of nodes `P` and array of facets `t`) for the conductor's side surface. The coil CAD model should properly define the normal vectors of the triangular surface patches and the corresponding triangle orientation. Once converted to `*.stl` format using MATLAB's built-in function `stlwrite`, this coil model may be used in FEM-based software packages (e.g. ANSYS Electronics Desktop). Fig. 4 shows a detailed concept of the combined wire/CAD coil model using in this software.

At present, the coil geometry modeler is restricted to predominantly flat or moderately bent conductor loops or nearly planar curves. H-coils (see, for example, Deng, etal., 2013) with sharp conductor bends in all three planes may be constructed if necessary, but only using the circular conductor cross-section.

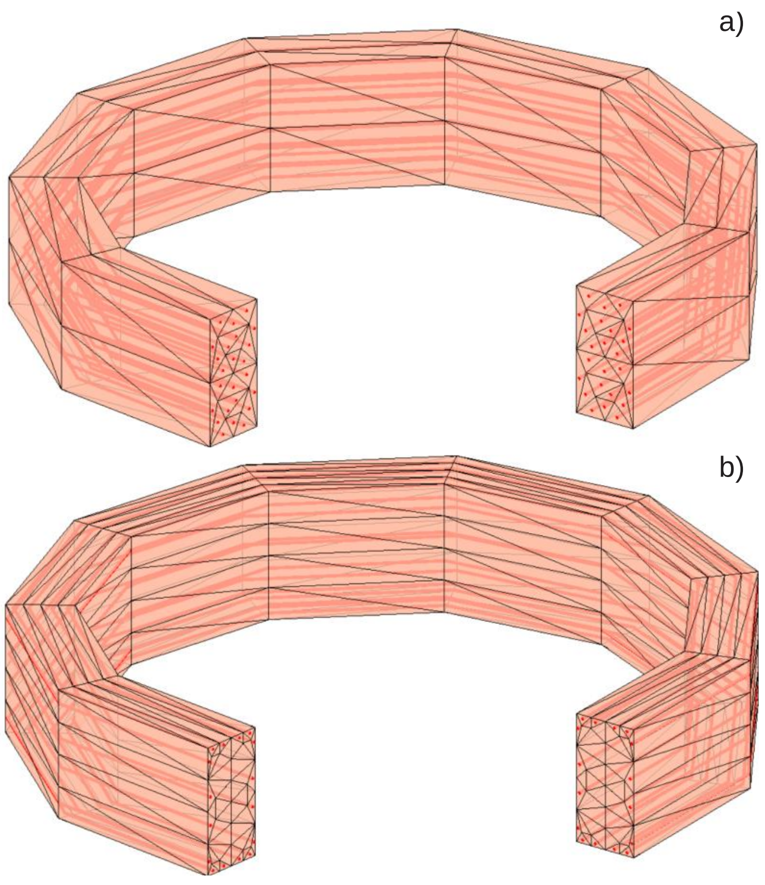


Fig. 4. Concept and construction of the coil model. Filaments of current (red) within conductor's surface CAD model are shown. a) – Uniform current distribution (Litz wire); b) – modeling the skin effect (a solid conductor at a high frequency).

6. Head Model Import, Remeshing, and Analysis (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`) containing arrays of vertices $\boldsymbol{P}$, 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. 5a below). Repeat this last operation for every brain compartment in the folder.

Built-in head models. For computational studies that do not involve MRI data collection, the Dropbox location provides 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 Dropbox location 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. The final model included is the FDA's MIDA model (Iacono et al 2015) with 11M facets and 117 tissues, which will be discussed in greater detail in a future publication.

Any other surface model obtained from 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).

Additionally, 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. 5a.

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.

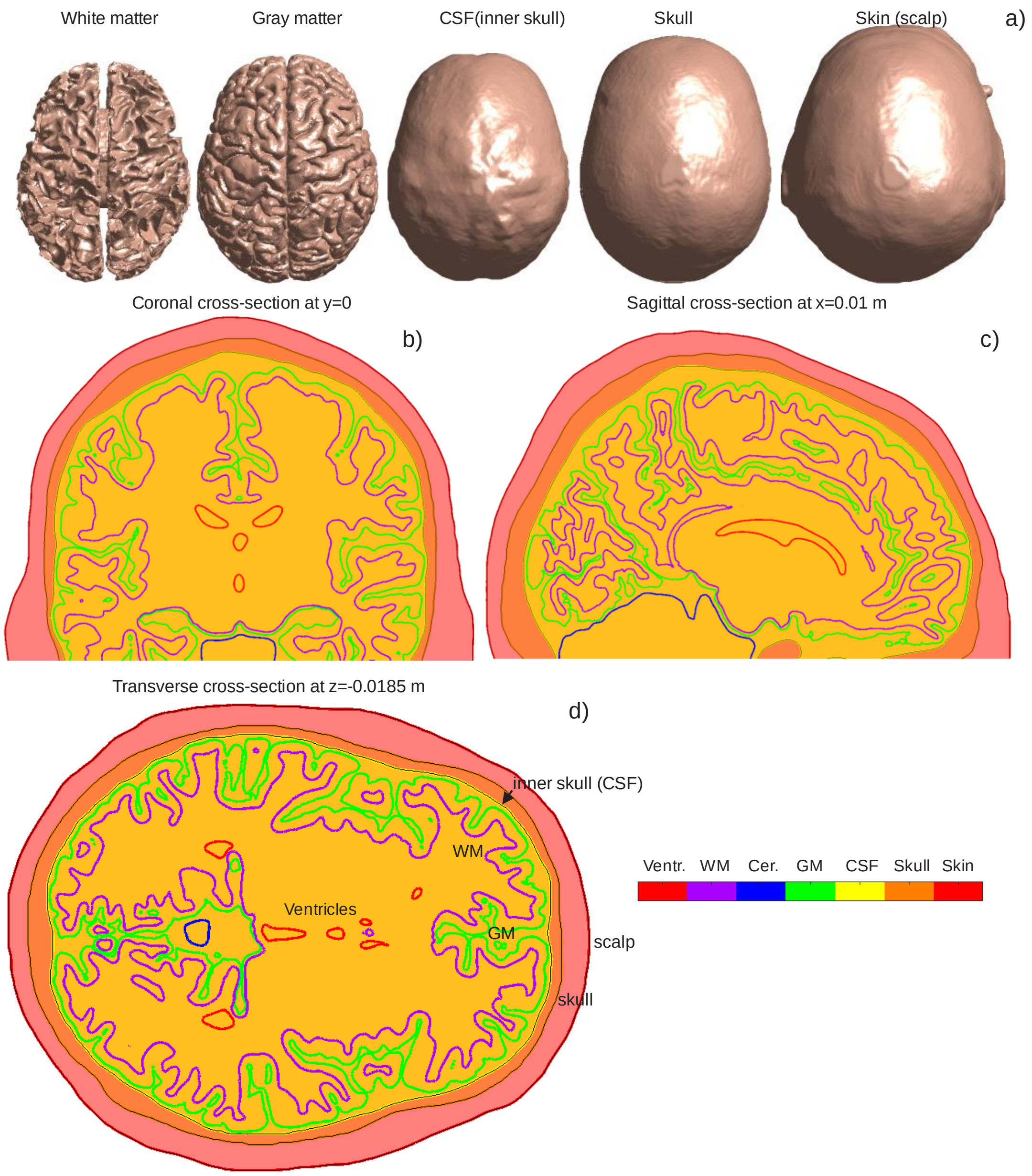


Fig. 5. a): Brain compartments of the default package head model # 110411: white matter (WM), gray matter (GM), cerebrospinal fluid (CSF or inner skull), skull, and skin; b-d) – Head cross-sections in three principal planes.

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 1.8 M. Fig. 6 shows the corresponding surface meshes for the gray matter shell along with the original segmentation. The red circle labels the targeted stimulation area close to the precentral gyrus crown. Note that the remeshing procedure may require significant time.

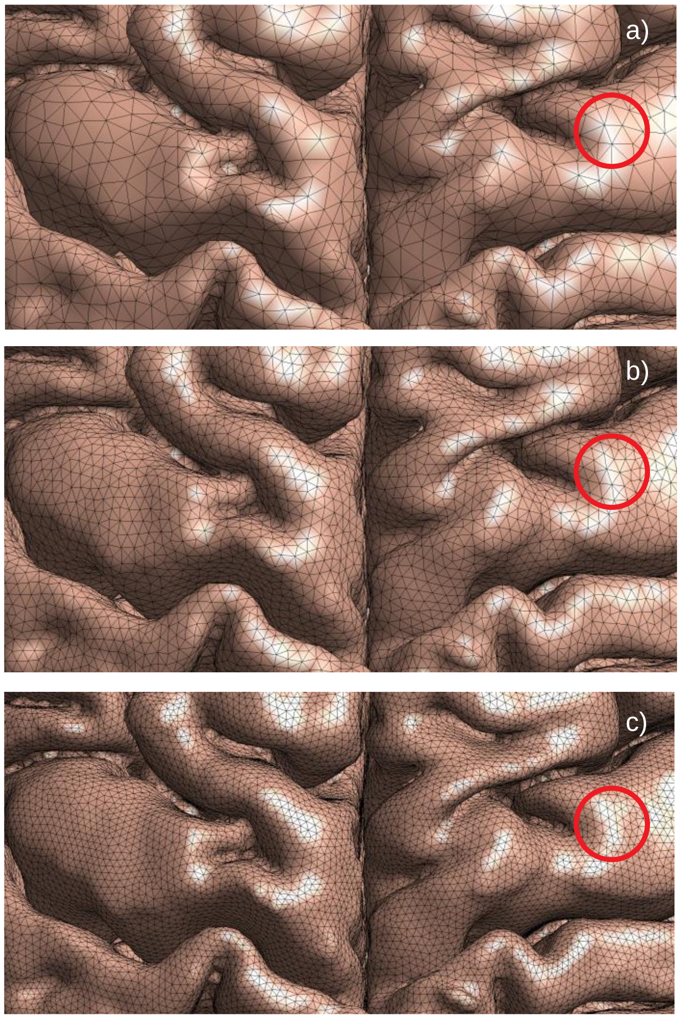


Fig. 6. a) – Coarser model with average cortical edge length of 1.9 mm and average cortical nodal density of 0.32 nodes per mm^2; b) – default meshing with average cortical surface mesh edge length of 1.5 mm and average cortical nodal density of 0.55 nodes per mm^2; c) – finer model with average cortical edge length of 0.99 mm and average cortical nodal density of 1.2 nodes per mm^2. The red circle labels the targeted stimulation area close to the precentral gyrus crown.

Creating combined head mesh. The combined mesh for the entire head is created by appending individual meshes. This is done 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 default (minimum) number of neighbors is 4. Integrals are computed in parallel, using 15 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`.

The script `model01_main_script.m` reads from a tissue index file (always 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.

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. For the MIDA model, however, the interior and exterior boundaries of every tissue are explicitly segmented. This means, for example, that the MIDA model's white matter and gray matter 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. 7 below illustrates the results of this operation.

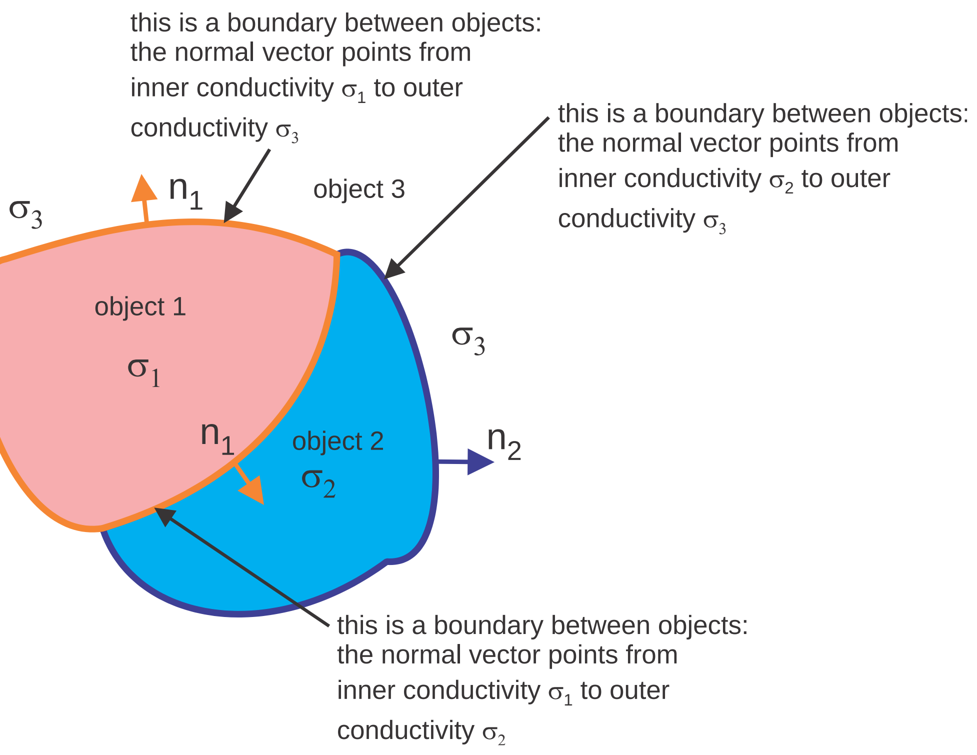

this is a boundary between objects: the normal vector points from inner conductivity σ1 to outer conductivity σ3
this is a boundary between objects: the normal vector points from inner conductivity σ2 to outer conductivity σ3
σ3
n1
object 3
object 1
σ1
σ3
n1
object 2
σ2
n2
this is a boundary between objects: the normal vector points from inner conductivity σ1 to outer conductivity σ2

Fig. 7. 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.

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 coil positioning. Run the script and observe the generated results. Change the line definition if desired.

7. Computational Workflow (main folder)

The computations are performed in the main folder. The scripts of the main folder should be executed *sequentially*.

Go to the main folder and execute the script `bem0_load_model.m` first. This script imports head model data into the MATLAB workspace and sets the MATLAB path. It also imports the previously computed solution if available. Next (and before running the simulations), the coil position above the head as in Fig. 8 must be defined.

Coil positioning/tissue properties. Run the script `bem1_setup_coil.m`. This script

(i) initializes the coil's time-varying current, dI/dt;

(ii) defines the steady-state current, I_0, which is necessary to compute the magnetic field;

(iii) defines coil position above the head by proper rotation and translation. The corresponding steps include coil rotation about its axis, tilt, and translation;

(iv) determines coil centerline or another observation line;

(v) displays the combined head-coil geometry (skin, skull, GM, or WM shell) as shown in Fig. 8; and

(vi) displays (nearest) intersection points between the tissues and the coil's centerline; also displays the corresponding distances from the bottom of the coil to the intersection points.

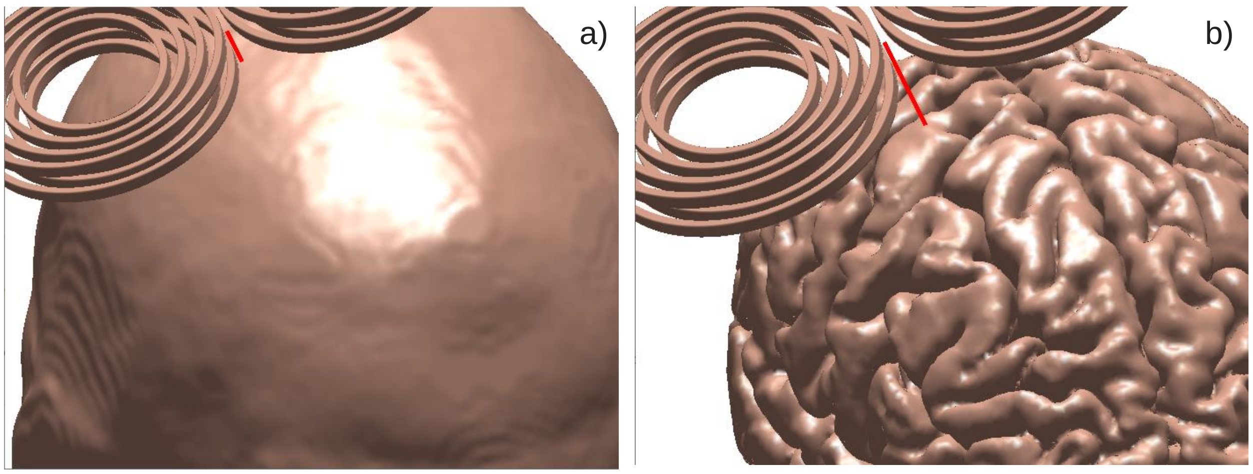


Fig. 8. Output of the script `bem1_setup_coil` for the same coil-model configuration: a) – Coil position above the skin shell. The distance from the coil bottom to the skin shell along the coil centerline is 10.8 mm; b) – Coil position above the GM shell. The distance from the coil bottom to the GM shell along the coil centerline is 26.2 mm.

Coil position adjustment may be performed by running the script `bem1_setup_coil` multiple times. When performing mathematical (e.g. coil rotation) and graphical operations, this script calls several functions from subfolder `Engine`.

Specifically, coil positioning is done in three steps:

```
coilaxis        = [0 0 1];                   %   Transformation 1: rotation axis
theta           = 0;                         %   Transformation 1: angle to rotate
Nx = +0.45; Ny = 0.0; Nz = 1.0;              %   Transformation 2: New coil centerline
MoveX = +42e-3; MoveY = 0; MoveZ = 79.5e-3;%   Transformation 3: New coil position
```

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

(i) computes the primary field of the coil on every head interface (face nodes then face centers by interpolation) 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, Saad 2003);

(iii) computes the surface field (only the continuous E-field contribution) based on the known surface charge distribution and;

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

Fig. 9a shows the typical convergence rate (relative residual of the iterative solution). Usually, 14 iterations are enough for a reliable result ([16]). Fig. 9b is the same result when the charge conservation law is ignored; and Fig. 9c is the same result when the neighbor potential electrostatic integrals are replaced by the crude center-point approximation.

While the convergence without the charge conservation law might appear acceptable for this particular example, this solution typically converges to an incorrect result where charges accumulate at the sharp boundaries of the bottom of the head.

As for the near-field integration accuracy of the default example, three double potential integrals for three neighbor triangular patches (default value is given in the script `model01_main_script.m`) are computed precisely. 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 a larger storage.

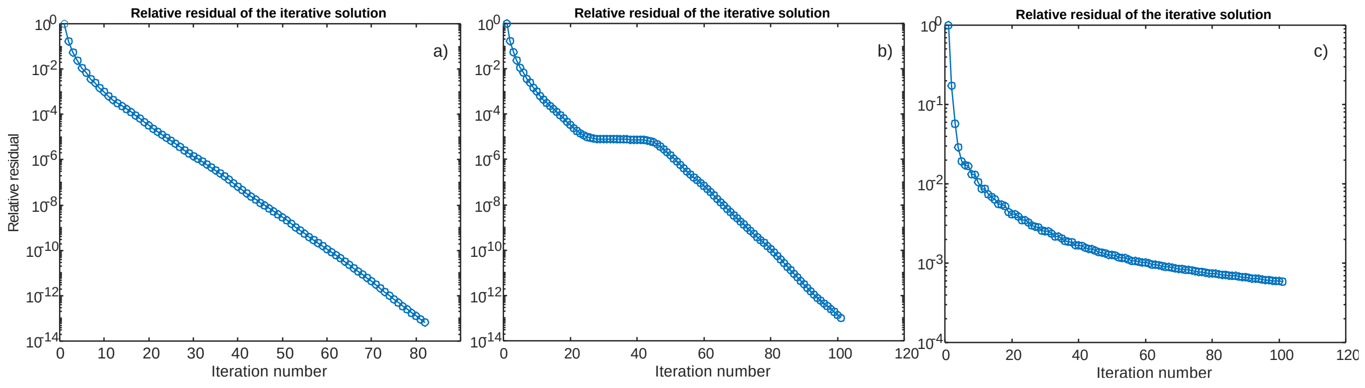


Fig. 9. a) –Typical convergence rate of the solution for the default example; b) – The same result when the charge conservation law is ignored; and c) –The same result when the neighbor potential electrostatic integrals are replaced by the center-point approximation.

Surface charge averaging. For practical purposes, it might be convenient to introduce weighted surface charge averaging (i.e., to low pass filter the surface charge density). The default option averages 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`.

8. Fields Output (main folder)

Fields just inside/outside tissue interfaces. The script `bem3_surface_field_e` displays the generally discontinuous electric field just inside or outside any head compartment. It can be the total, normal, or the tangential field. The tangential field component is continuous through the interface, but the normal (and thus the total) field component is almost always discontinuous.

The script `bem3_surface_field_b` performs a similar operation for the magnetic field, which indeed remains continuous across boundaries.

Scripts `bem3_surface_field_c.m` and `bem3_surface_field_p.m` perform the same operation for surface charge density and the continuous surface potential. Fig. 10 shows the output of `bem3_surface_field_b.m` and `bem3_surface_field_e.m` for the gray matter interface. The total electric field just inside the gray matter shell is plotted.

The script `bem3_surface_field_l` displays the resulting Lorentz force density just inside or outside any head compartment. It can be the total, normal, or tangential force.

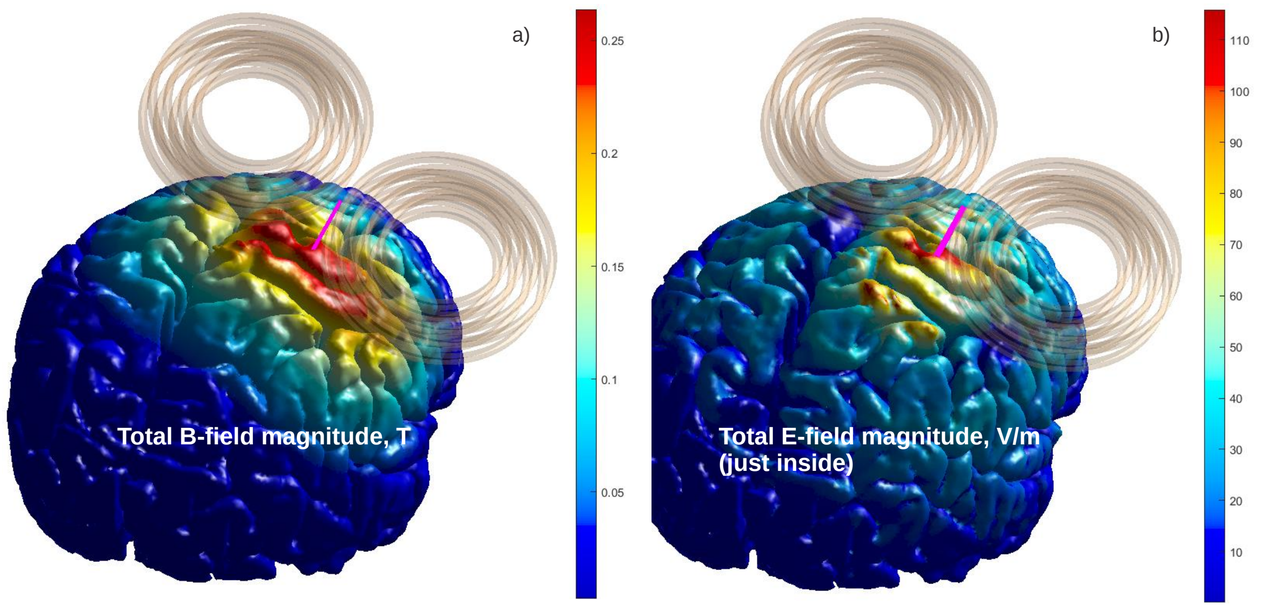


Fig. 10. a) – Magnitude of total magnetic field on the gray matter shell; b) – Magnitude of total electric field just inside the gray matter shell (the top of the cortical layer).

Normal fields just inside/outside tissue interfaces. Due to the geometry and electrophysiological characteristics of cortical neurons, the electric field component perpendicular to the cortical layer might be significant for neurostimulation. The script `bem3_surface_field_the.m` displays the *normal* field component just inside or just outside any brain compartment, respectively. Those components are directly obtained from the already-known surface-charge solution without extra computations ([16]). The script `bem3_surface_field_thl.m` displays the corresponding Lorentz force density.

To better visualize the focal area for the normal component at the interfaces, a small blue ball is drawn at the center of every facet where the absolute field value is in the range 80-100% of the maximum field value observed just inside (or outside) the shell. The parameter of 80% is the field threshold margin; it is assigned in the script `bem1_setup_coil.m` as

```
margin = 0.80;
```

and can be changed at any time if necessary.

Fig. 11 shows the corresponding display for the normal fields just inside the GM and WM shells, respectively, for the default example. One can see that the normal coil field just inside the WM shell appears to be quite focal in this particular case; however, the focal area is located not directly underneath the coil.

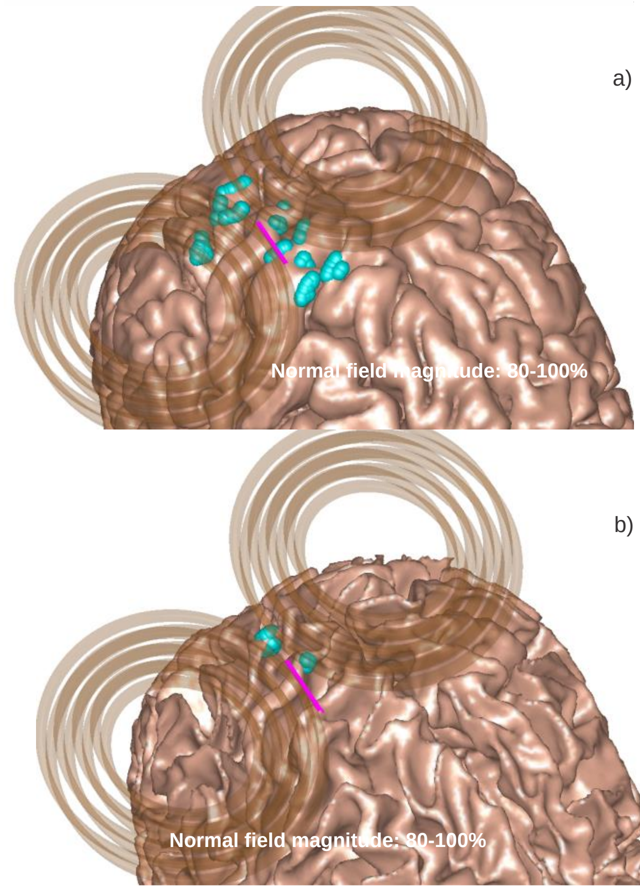

a)
Normal field magnitude: 80-100%
b)
Normal field magnitude: 80-100%

Fig. 11. Focal area of the normal field just inside GM (a) and WM (b). The blue spheres indicate facets whose normal field magnitudes are 80-100% of the corresponding maximum field magnitude.

Observation plane definitions, segmentation cross-section precomputation. Next, run the script `bem4_define_planes.m`. This script defines the observation planes for the 3D display and pre-computes segmentation (triangular surface mesh) cross-sections that will be used in the field output plots. If NIfTI data are available (e.g. `T1w.nii`), subsequent visualizations will superimpose the mesh cross-sections and observation planes onto the corresponding NIfTI slices.

Volumetric fields in principal planes. The scripts `bem5_volume_XY/XZ/YZ.m` compute and output the electric field and the electric current density (any of its Cartesian components or a magnitude) in the three principal planes. The plane position and its size are specified in the script `bem4_define_planes.m`. It is possible to plot the field only within the selected brain compartments. Examples are given in [16].

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 versus 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 is 2-5; higher numbers might provide better field accuracy but will simultaneously slow down the computations.

9. Advanced: Control of Numerical Accuracy

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

1. `RnumberE`, found in `Model\model01_main_script.m`: Number of neighbor potential electrostatic double surface integrals (electric field + electric potential) to be computed precisely. The default number is 4. The maximum number is unlimited, but it is subject to memory constraints. Numbers above 16 do not affect the overall solution accuracy significantly.

2. `prec`, found in `Engine\bemf4_surface_field_electric.m` and in all other FMM-related scripts: Intrinsic volumetric FMM precision (Gimbutas et al., 2019). The default value is 0.1-0.01 (10-1%). Values smaller than 0.01 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 14. The maximum number is unlimited but is subject to speed constraints. Numbers above 20 do not affect the overall solution accuracy.

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

5. `R`, found in `bem5_volume_XY/XZ/YZ.m` (in the main folder): Dimensionless (vs. average triangular face size) radius R of an integration sphere within which precise integration of the surface charge density is performed when computing volumetric fields. The default value is 4. The maximum number is unlimited, but it is subject to speed constraints.

10. Test of Numerical Accuracy

In application to TMS problems, the BEM-FMM algorithm was previously tested and validated against analytical and FEM numerical solutions by Makarov et al 2018 and Htet et al 2019; a very detailed and rigorous independent comparison study was further performed by Gomez et al 2019, [5].

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

[3] DropBox Repository Aug. 2020: TMS Modeling Toolkit v1.0. Online: https://tmscorelab.github.io/TMS-Modeling-Website/

[4] 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.

[5] Gomez LJ, Dannhauer M, Koponen LM, Peterchev AV. Conditions for numerically accurate TMS electric field simulation. *Brain Stimul*. 2020 Jan-Feb;13(1):157-166. doi: 10.1016/j.brs.2019.09.015

[6] 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.

[7] 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.

[8] 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*. 2019:16:1-13. doi: https://dx.doi.org/10.1088/1741-2552/aafbb9.

[9] 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*. 2019: 6(5):1-13. doi: https://doi.org/10.1088/2057-1976/ab4c76.

[10] 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

[11] 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

[12] 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.

[13] 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.

[14] 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.

[15] Makarov SN, Navarro de Lara L, Noetscher GM, Nummenmaa A. Modeling Primary Fields of TMS Coils with the Fast Multipole Method. *biorxiv*. 2019. doi: https://doi.org/10.1101/514919.

[16] Makarov SN, Wartman WA, Daneshzand M, Fujimoto K, Raij T, Nummenmaa A. A software toolkit for TMS electric-field modeling with boundary element fast multipole method: an efficient MATLAB implementation. *J Neural Eng*. 2020;17(4):046023. Published 2020 Aug 4. doi:10.1088/1741-2552/ab85b3

[17] 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.

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

[19] 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.

[20] 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/

[21] 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.

[22] 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/