COMETS v. 2.0

A Matlab toolbox for simulating electric field due to tDCS

User's Manual

Manual Written by

Chany Lee

Contact: cometstool@gmail.com

Computational NeuroEngineering Lab
Hanyang University, Seoul, Korea
http://CoNE.hanyag.ac.kr

Quick Guide

Operation conditions

- Microsoft Windows environment (tested on Windows 7 and Windows 10)
- MATLAB v.2015 or higher version

Program installation and start

- Download COMETS v2.0 and related documents from the official website (http://www.cometstool.com).
- Unzip compressed file to a folder.
- Run Matlab and change working folder to the folder of COMETS v2.0.
- Type *Comets* in the command window of Matlab.

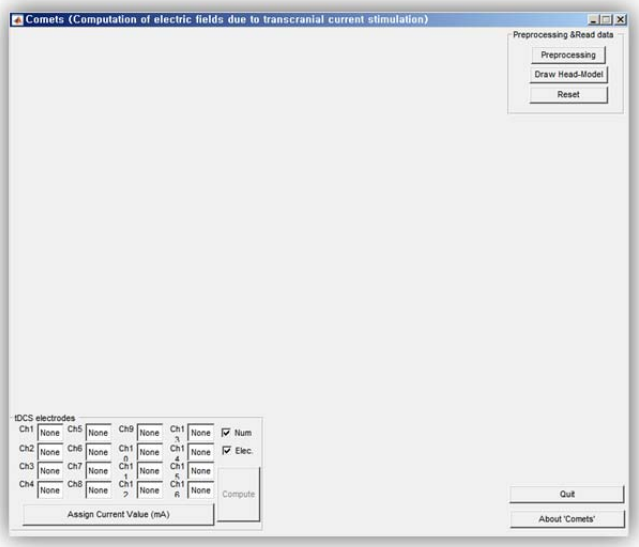


Fig.1. Initial window of COMETS2

Data import and preprocessing

- Press *Preprocessing* button, and you can find the popup file browser.
- Select testmodel.node. Comets v2.0 reads testmodel.node and testmodel.ele, simultaneously.
- If you select other files named as xxx.node, Comets v2.0 reads xxx.node and xxx.ele. Two files should be saved in the same folder.
- Details of input data can be found in the “DATA FORMAT” section of this manual.

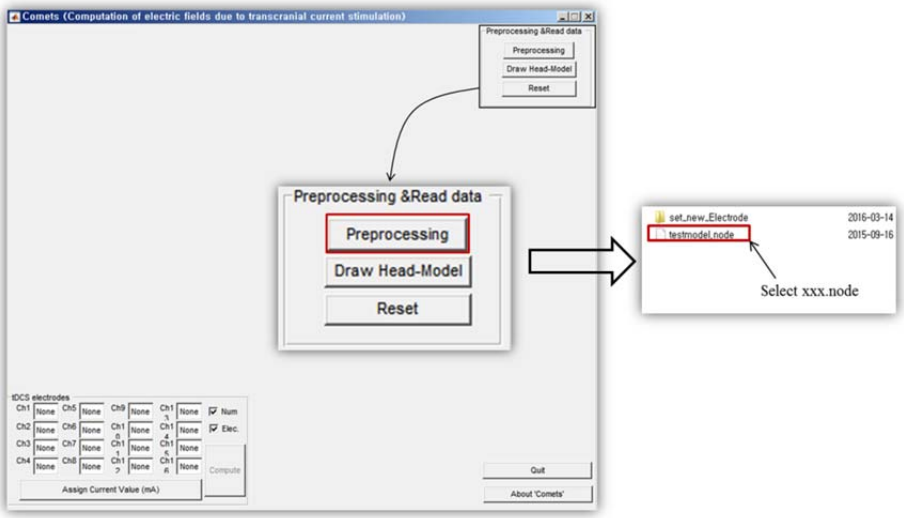


Fig. 2. Selecting a file for preprocessing.

• File selection is the start of preprocessing.

• During the preprocessing, messages are printed in the command window of Matlab. Elapsed time depends on the size of the head model data.

• After the preprocessing, a message box informs *Preprocessing Completed* and disappears in a few seconds.

Loading scalp surface

• Press *Draw Head-Model* button.

• Extracted scalp surface from input files are shown in main panel as in Fig. 3.

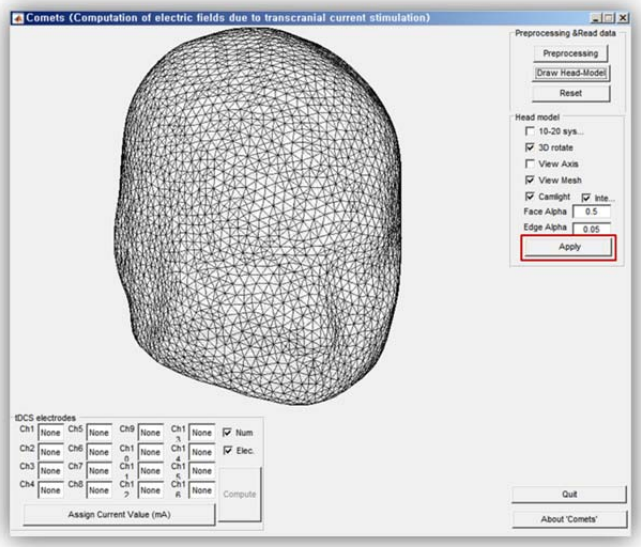


Fig. 3. Drawing of scalp in the main window.

• Press *Apply* button as indicated in Fig. 3. You can manipulate the view options in the *Head Model* box (green box in Fig 4).

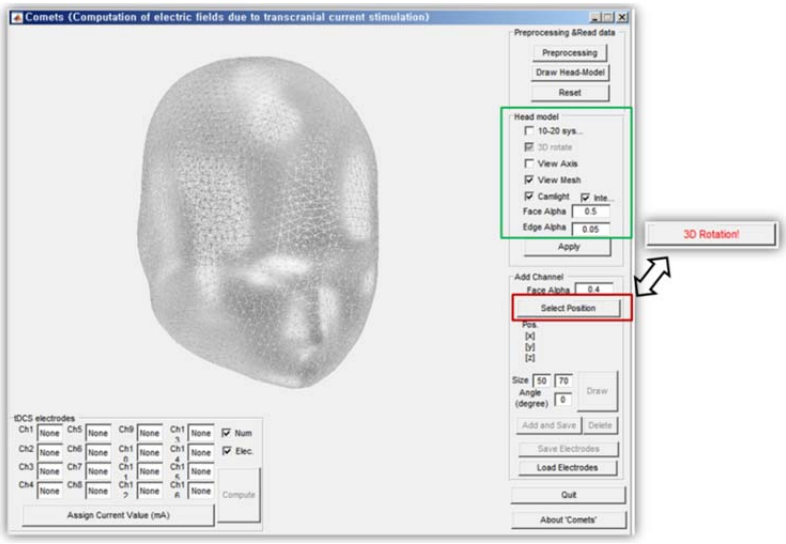
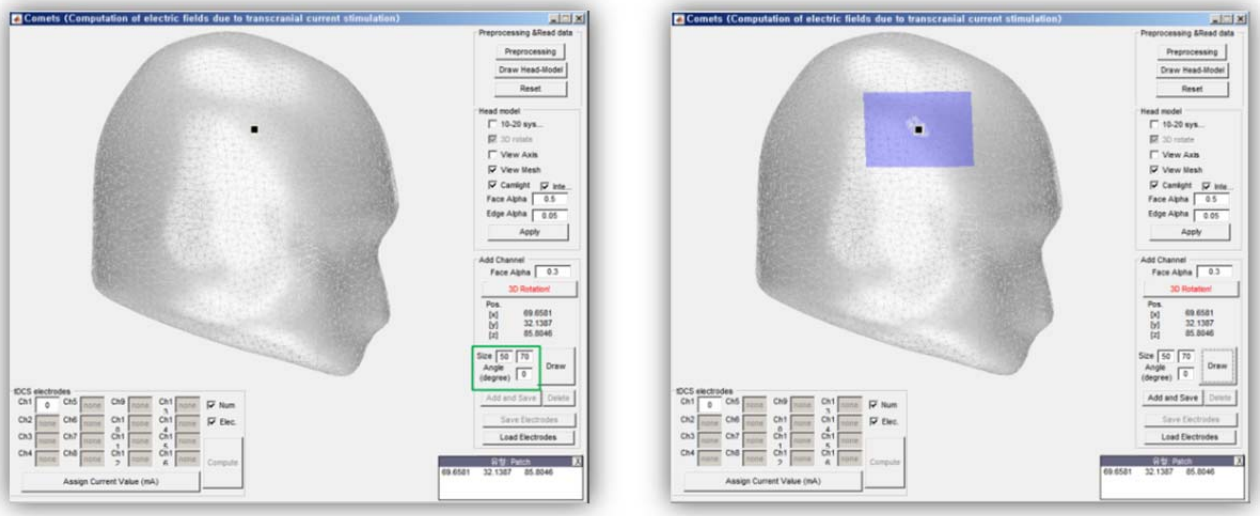


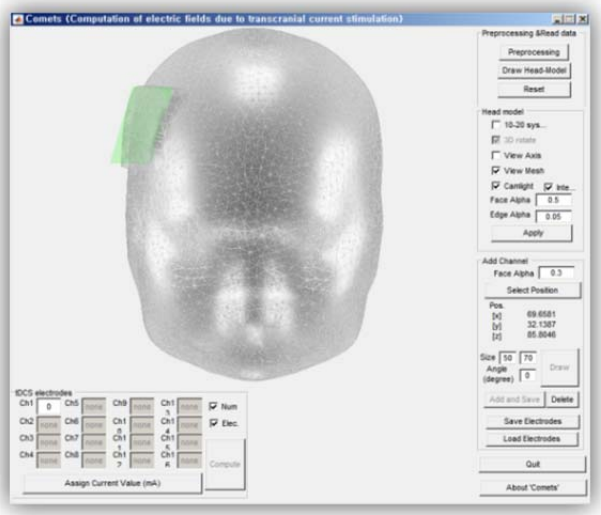
Fig. 4. View options.

Electrode generation

- If *Select Position* is written in the red box of Fig. 4, you can rotate the scalp surface.
- If *3D Rotation* is written in the red box of Fig. 4, you can select locations of the electrodes.
- Rotate the head to an adequate angle, and then press *Select Position*.
- Select a node by clicking left button of mouse as shown in Fig. 5(a). This selected node will be the center of an electrode.
- Type the size and angle of an electrode in the green box of Fig. 5(a). Units of length and angle are millimeters and degrees, respectively.
- Press *Draw* button. You can see a blue rectangle roughly indicating the electrode shape (Fig. 5(b)).
- Adjust the size and angle of the electrode, and press *Draw* button. You can do this repeatedly.
- Press *Add and Save* button. After a few seconds, the rectangular sponge electrode is generated on the scalp (Fig. 5(c)).



(a) (b)



(c)

Fig. 5. Procedure for electrode generation: (a) A black point is the center of an electrode. (b) A blue rectangle shows the rough shape of the electrode. (c) Generated electrode.

• Make the other electrode in the same way.

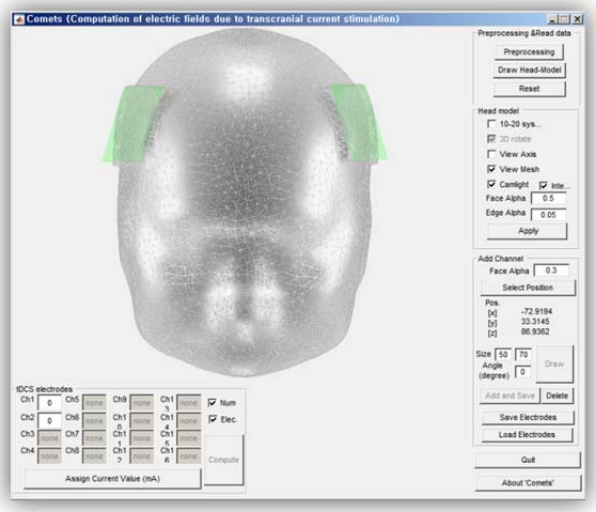


Fig. 6. Two generated electrodes.

• If an error message is printed in the Matlab command window as you press *Add and Save* button, it means that the generation of electrodes failed. This 'rare' failure generally results from the coordinates and connectivity of nodes in the original head model. In this case, please try to slightly move the position of the center point, or slightly change the size or angle of the electrode.

• If you want to delete previously generated electrode, just press *Delete* button.

Calculation of electric potential, electric field intensity, and current density

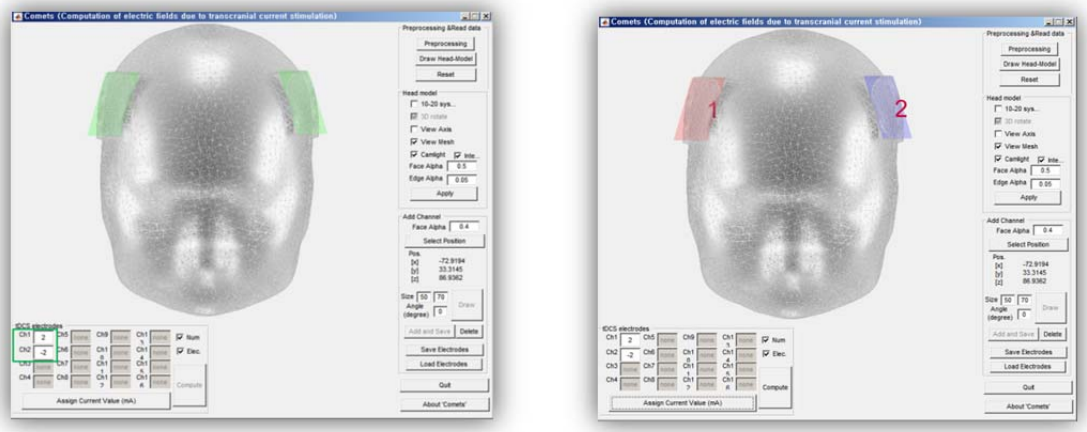
• In the green box of Fig. 7(a), the amount of current injection should be described. Textbox for *Ch1* and *Ch2* are activated. Two values should have the identical absolute value but should have opposite signs.

• The unit of current is mA.

• More than two electrodes can be generated, but this version of COMETS cannot solve such multi-electrode problems.

• After typing the amount of current, press *Assign Current Value* button. Then, you can see the numbered electrodes (Fig. 7(b)).

• Press *Compute* button. Some messages will be printed in the command window.



(a) (b)

Fig. 7. Setting the amount of current injection.

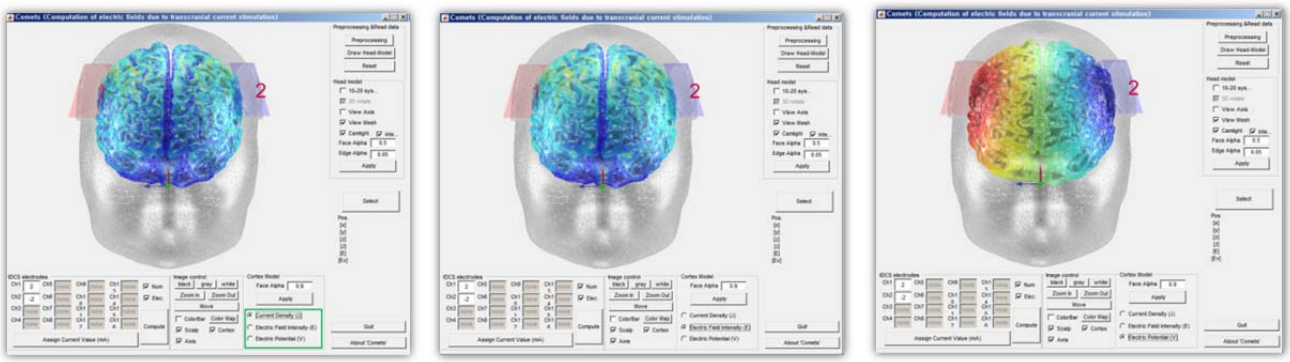
Observation of results

• After the calculation, results automatically appear in the main window of COMETS.

• Default value is the *Current Density*. *Electric Field Intensity* and *Electric Potential* are also available.

• Please try different image control options to change the visualization. *Black*, *gray*, and *white* represent background colors. You can click tick boxes to show or hide electrode number (*Num*), electrode shape (*Elec.*), color bar (*ColorBar*), scalp surface (*Scalp*), cortical surface (*Cortex*), and axis (*Axis*). You can make your own color map by clicking *Color Map*. Please refer to the help document of 'colormapeditor' for more information on editing color maps.

• You can probe the exact physical quantities by selecting points on the cortical surface, after pressing *Select* button. If you press *Apply* button in the head model box, you cannot use this 'probe' function (This is a known bug. We will try to fix this ASAP).



(a) (b) (c)

Fig. 8. Results of tDCS analysis. (a) Current density. (b) Electric field intensity. (c) Electric potential.

• For advanced users, a file named 'cortexfield.tec' is automatically generated in the main folder of COMETS2. '.tec' is

supported by a commercial software package, TECPLOT (http://www.tecplot.com/), which can be used for visualization of finite element data. This file is composed of three parts: header, node data, and element data. In the header, the numbers of nodes and elements are specified. Node data have 6 columns. First three columns are coordinates of nodes. Electric current density, electric field intensity, and electric potentials are written in the 4-th, 5-th, and 6-th columns, respectively. Elements data has information on tetrahedrons. Users can use other visualization software by slightly modifying this *.tec file. Fig. 9 shows an example of the visualization using TECPLOT.

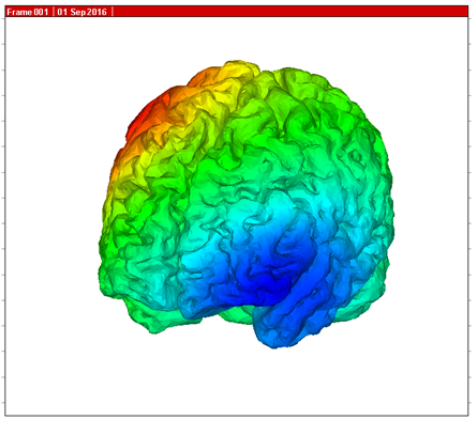


Fig. 9. A plot using TECPLOT.

APPENDIX

Structure of COMETS2

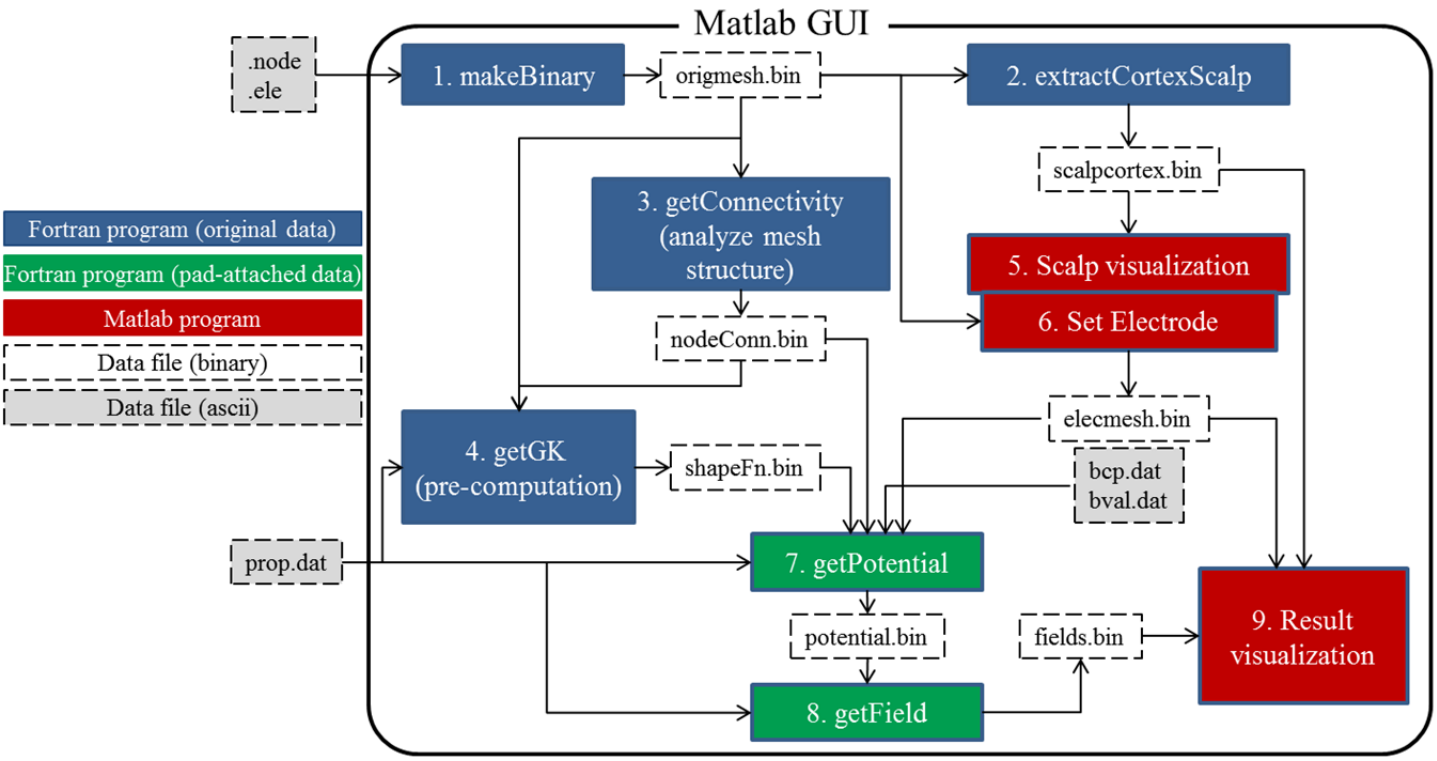
Data format

Reading and writing binary file

Generating head model

References

Structure of COMETS2



DATA FORMAT

xxx.node, xxx.ele

COMTES2 begins with these two files. Finite element mesh data are composed of nodes (points) and elements (tetrahedrons). 'xxx.node' contains coordinates of nodes, and 'xxx.ele' includes information of tetrahedrons. These files can be generated by TETGEN. The detailed information on the file formats can be found at the home page of TETGEN (http://wias-berlin.de/software/tetgen/). To generate an individual head model, please refer to *Generating head model* section.

origmesh.bin

This binary file just contains data from 'xxx.node' and 'xxx.ele.' 'Int' type is 4-byte and 'Real' type is 8-byte.

Data type	dimension	value
Int	1 X 1	nnode_orig
Int	1 X 1	nele_orig
Real	3 X nnode_orig	nodes_orig
Int	4 X nele_orig	elements_orig
Int	1 X nele_orig	regions_orig

nnode_orig: the number of nodes.
nele_orig: the number of elements
nodes_orig: (x, y, z) coordinates of nodes
elements_orig: elements data of head model
regions_orig: region numbers of head model
*Suffix '_orig' means that this variable is about original head model before electrode attachment.

nodeConn.bin

This file contains how many nodes are connected to each node and what nodes are connected to each node. In FEM, node connectivity is directly related to the structure of the stiffness matrix.

Data type	dimension	value
Int	1 X nnode_orig	nnodeConn_orig
Int	1 X nnodeConn_orig(1)	nodeConn_orig (1:nnodeConn_orig (1),1) node list connected to node1
Int	1 X nnodeConn_orig (2)	nodeConn_orig (1:nnodeConn_orig (2),2) node list connected to node2
…	…	…
Int	1 X nnodeConn_orig (# nodes)	nodeConn_orig (1:nnodeConn_orig (nnode_orig),nnode_orig) node list connected to node(nnode_orig)

nnodeConn_orig: the number of nodes directly connected to each node
ex) nnodeConn_orig(1001): the number of nodes directly connected to 1001st node
nodeConn: the list of nodes connected to each node
ex) nodeConn_orig(1:nnodeConn_orig(1001),1001): the list of nodes directly connected to 1001st node

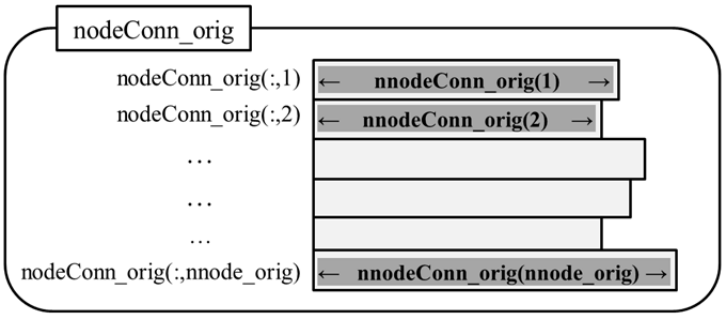


Fig. A1. Structure of nodeConn_orig.

shapeFn.bin

In this file, variables for direct sparse solver (DSS) of intel math kernel library (MKL) are saved. More information about MKL can be found in webpage at https://software.intel.com/en-us/intel-mkl.

Data type	dimension	value
Int	1 X 1	nGK_orig
Real	1 X nGK_orig	GK_orig
Int	1 X nGK_orig	columns_orig
Int	1 X (nnode_orig+1)	rowIndex_orig

nGK_orig: the number of nonzero entries of global stiffness matrix for FEM
GK_orig: Global stiffness matrix. This contains upper triangular matrix of global stiffness matrix because of symmetry.
columns_orig: Column information of GK_orig
ex) columns_orig(i) contains column number of GK_orig(i).
rowIndex_orig: GK_orig index of first entry for each row
ex) rowIndex_orig(i) contains index of first entry of i-th row in GK_orig

scalpcortex.bin

Data type	dimension	value
Int	1 X 1	nnode_scalp
Int	1 X 1	nele_scalp
Real	3 X nnode_scalp	Nodes on scalp
Int	3 X nele_scalp	Element data of scalp
Int	1 X nnode_scalp	trace_scalp
Int	1 X 1	nnode_cortex
Int	1 X 1	nele_cortex

Real	3 X nnode_cortex	Nodes on cortex
Int	3 X nele_cortex	Element data of cortex
Int	1 X nnode_cortex	trace_cortex

nnode_scalp: the number of nodes on scalp.
nele_scalp: the number of elements (triangles) on scalp.
Nodes on scalp: coordinates of nodes.
Element data of scalp: triangles of scalp mesh.
trace_scalp: node numbers of head mesh. ex) trace_scalp(i) is node number of head mesh corresponding to i-th node of scalp.
nnode_cortex: the number of nodes on cortex which is defined as a surface between brain and CSF.
nele_cortex: the number of elements on cortex.
Nodes on cortex: coordinates of nodes.
Element data of cortex: triangles of cortex.
trace_cortex: node numbers of head mesh.
*Variables with suffixes '_scalp' and '_cortex' are surface mesh data.

elecmesh.bin

This file contains data of head mesh after electrode generation.

Data type	dimension	value
Int	1 X 1	nnode_scalp
Int	1 X 1	nele_scalp
Real	3 X nnode_scalp	Nodes on scalp
Int	3 X nele_scalp	Element data of scalp
Int	1 X nnode_scalp	trace_scalp
Int	1 X 1	nnode
Int	1 X 1	nele
Real	3 X nnode_cortex	Nodes in head
Int	3 X nele_cortex	Element data of head
Int	1 X nnode_cortex	Regions of head
Int	1 X 1	nElecSeedNodes
Int	1 X nelecSeedNodes	Seed nodes for electrodes
Int	1 X 1	nElectrode
Int	1 X nElectrode	nInterfaceFaces
Int	1 X nInterfaceFaces(1)	Interfacial face of 1-st electrode
…	…	…
Int	1 X nInterfaceFaces(nElectrode)	Interfaial face of nElectrode-th electrode

nnode_scalp: the number of nodes on scalp. After electrode generation, outermost surface is assumed as scalp.
nele_scalp: the number of elements on scalp.
Nodes on scalp: coordinates of nodes.
Element data of scalp: triangles of scalp mesh.
trace_scalp: node numbers of head mesh.
nnode: the number of nodes in whole head after electrode generation.
nele: the number of elements in whole head after electrode generation.
Nodes in head: coordinates of nodes.
Element data of head: tetrahedrons in head mesh
Regions of head: region numbers of head model.
nElecSeedNodes: the number of generated nodes
Seed nodes for electrodes: the list of added nodes for electrode generation.
nElectrode: the number of electrode. In COMETS2, nElectrode is 2.
nInterfaceFaces: the numbers of triangles between electrode and original scalp.
Interfacial face of i-th electrode: the list of triangles between electrode and original scalp.

prop.dat

This file contains material properties (electrical conductivities) for finite element analysis.

prop.dat

```
nProp
prop(1)
prop(2)
…
prop(nProp)
```

nProp: the number of properties. This number is identical to the number of tissues considered in finite element analysis.
prop(i): electric conductivity of i-th tissue.
Ex) An element with region number 2 has electric conductivity of prop2.

bcp.dat, bval.dat

These files are used for boundary condition of FEM. ‘bval.dat’ is unchanged. ‘bcp.dat’ contains the list of nodes at which the boundary conditions are imposed.

bcp.dat

```
nBC
node1 -1
…
nodeM -1
node(M+1) -2
…
node(M+N) -2
```

bval.dat

```
2
1
-1
```

nBC: the number of nodes on boundary condition. nBC equals M+N.
node(1)~node(M): list of nodes of electrode1
node(M+1)~node(M+N): list of nodes of electrode2
-1, -2: ‘-1’ and ‘-2’ mean electrode1 and electrode2, respectively.

potential.bin

Data type	dimension	value
Real	1 X nnode	potentials

potentials: electric potential of whole head

field.bin

Data type	dimension	value
Real	1 X nnode_cortex	magnitude of E field
Real	1 X nnode_cortex	magnitude of current density

magnitude of E field: magnitude of electric field intensity at cortical nodes.
magnitude of current density: magnitude of current density at cortical nodes.

Examples of reading and writing a binary file

Matlab

```
fid = fopen(filename,'w');
fwrite(fid, e0.nnode, 'int32');
fwrite(fid, e0.nele, 'int32');
fwrite(fid, nodes, 'double');
fwrite(fid, elements, 'int32');
fclose(fid);

fid = fopen(filename,'r');
E_Field_Cortex = fread(fid, [nnode_cortex 3], 'double');
fclose(fid);
```

Intel fortran for Windows (Intel Fortran compiler 10.0)

```
open(unit=1,file=filename, form='binary', status='old')
read(1) nGK_orig
read(1) GK_orig
read(1) columns_orig
read(1) rowIndex_orig
close(1)
open(1,file=filename,status='replace',form='binary')
write(1) nnode, nele
write(1) nodes
write(1) elements
write(1) regions
close(1)
```

Intel fortran for Linux (Intel parallel studio 2013, LinuxMint17)

```
open(1,file=filename,status='replace',form='unformatted', access='stream')
write(1) nnode, nele, nface
write(1) nodes
write(1) elements
write(1) regions
write(1) faces
close(1)
open(1,file=filename,form='unformatted', access='stream')
read(1) nnode, nele, tempi
read(1) nodes
read(1) elements
close(1)
```

Generating Head Models

As mentioned above, COMETS2 starts with ‘xxx.ele’ and ‘xxx.node’ files. Hence, any methods for generating ‘xxx.ele’ and ‘xxx.node’ are acceptable. In this section, we introduce a procedure to generate head model using Curry7.

Curry7 (+Meshlab) + Iso2mesh

Using Curry7 (http://compumedicsneuroscan.com/), boundary element model (BEM) composed of triangular surfaces can be extracted from MRI data. This software can adjust the number of surfaces by changing the mesh resolution. Also, it provides the manual segmentation tool which is useful for modifying the error of auto-segmentation. Four surfaces can be extracted (scalp, outer skull, inner skull, and brain). After completing segmentation, the output files are named as ‘.sxx,’ and each file contains information of coordinates of nodes and a list of triangular surfaces. You can use other software to segment the tissues.

Additionally, smoothing was conducted with software called MeshLab (http://meshlab.sourceforge.net/). Though other software also has the function of smoothing, MeshLab is highly recommended since it is easy to use and has various smoothing functions which helps the user modifying the surface to desired one by changing smoothing parameters.

Using boundary element model, finite element model which is filled with tetrahedral volume can be created by the software called Iso2mesh (http://iso2mesh.sourceforge.net/) which is a Matlab-based mesh generator. This software uses Tetgen (http://wias-berlin.de/software/tetgen/) for creating volume elements so you must download it. Finite element mesh generation requires four steps:

1. Load information of nodes and faces of each surface to Matlab by reading ‘.sxx’ files.
2. Use ‘mergemesh’ function to combine the nodes and faces of all surfaces into global nodes and faces. For example, nodes and faces information of all surfaces are merged into global nodes and global faces.
3. Use ‘s2m’ function to create volumetric mesh. There are five input parameters in this function: *nodes*, *faces*, *keepratio*, *maxvol* and *method*. Input global nodes and faces derived from step 2 into *nodes* and *faces* parameter part. Select ‘cgalmesh’ for *method*, *keepratio* is automatically fixed as 1. Finally, decide a value of *maxvol* which is related to maximum tetrahedral element volume. This parameter determines a total number of elements. Proper values for parameters can be selected by users.
4. If you execute ‘s2m’ function, ‘surf2mesh’ function which is included in ‘s2m’ function will also be executed. Significant difference of input parameters between two functions is that there is a region parameter in ‘surf2mesh’ function. Users must find the representative points of each tissue from MRI data. If the points are exact, it is enable to distinguish one tissue from the others and each element assign their own region number which is corresponding to type of tissues.

When those steps are finished, Tetgen files (.node, .ele, .face) are generated. You can check the quality of mesh and attribution of regions with Tetview.

SIMNIBS

First of all, users should read and follow the installation guide and other documents at SIMNIBS webpage (http://simnibs.de/). SIMNIBS is executable only in OSX and Linux environments. Unless you have Mac., you have to install a distribution of Linux. According to documents in SIMNIBS webpage, Ubuntu 14.04, Ubuntu 16.04, and CentOS 6 were successfully tested for SIMNIBS. In addition, LinuxMint 17.2 was also tested by us.

If SIMNIBS and other required software are installed, finite element meshes can be obtained by the following procedures.

1. Convert MRI data to ‘nii’ data

    1.1 ‘dcm’ to ‘nii’

    In a linux console window,

    >>mri_convert –i *inputfile* –o *outputfile*

    Ex)

>>mri_convert –i ../data/leej001.dcm –o ./LJ.nii.gz

1.2 PAR/REC to ‘nii’

1.2.1 Run MRIcroGL (downloadable at http://www.mccauslandcenter.sc.edu/mricrogl/home) in Windows

1.2.2 Select ‘Import->Convert DICOM to NIFTI’ menu. Then, new window named as ‘dcm2nii’ come up.

1.2.3 Select ‘File->PAR/REC to NIFTI’

*Sometimes, after conversion using MRIcroGL (or other converting software), dimension scale changes.

2. Mesh generation (‘nii’ to ‘msh’ or ‘nii.gz’ to ‘msh’)

   Copy the file generated in step1 to linux or Mac. In a console window, type

   >>mri2mesh –all *subject_name subject_file*

   Ex)

   >>mri2mesh –all LJ LJ.nii.gz

   After several hours, ‘*subject_name*.msh’ file can be found.

3. ‘msh’ to ‘ele’ and ‘node’

   Copy ‘*subject_name*.msh’ file to Windows. Then, go ‘mshReader’ folder which is a subfolder of COMETS2 and run ‘ReadSIMNIBS.exe.’ Select a ‘msh’ file and press *Convert* button. After conversion, ‘*subject_name*.node’ and ‘*subject_name*.ele’ files are generated in the same folder of input file the window of ‘ReadSIMNIBS.exe’ disappears.

SIMNIBS usually generates mesh data with a large number of nodes and elements (approximately 5 times larger than the default model of COMETS2). Hence, computation generally requires much more time and much larger RAM when handling a mesh from SIMNIBS. Basically, SIMNIBS generates head models with 5 tissues such as white matter, gray matter, CSF, skull, and scalp. So, please keep in mind that ‘prop’dat’ file should be edited adequately.

If you have any questions, please send an e-mail to cometstool@gmail.com

Please sometimes visit the download page of COMETS2 if you would like to use the latest version of COMETS2.