.MSH BRAIN IMAGE MESH SEPARATION AND IMAGE GENERATION IN .PNG FORMAT FOR DISPLAY OF POST PROCESSING VIEW – NORME.

Function names used –
electric_field.py
image_elec_field.sh
visual_check2.geo

electric_field.py
This is the main code written in python environment and uses the other two codes (image_elec_field.sh and visual_check2.geo) for separation of the different mesh structures namely - White Matter, Grey Matter, Cerebrospinal fluid (CSF), Skull and skin – indexed 1,2,3,4,5 respectively and display their respective electric fields generated.

The algorithm used and a short explanation of the mesh modules are given below –
1) Adding the image processing software simnibs to the current python code for use of GMSH library (gmsh_numpy).
2) Importing the imported libraries used in the code – os, numpy, gmsh_numpy, glob and rankdata (from scipy.stats).
3) Sets the current path of subject dataset for use by the python compilor and assigns it to the variable (file_path_name) for use in the code.
4) This part of the code extracts all the .msh files (the subject files) in the current path and stores the file name along with the entire path name in a text file with the name filename.txt, for looping through the subject files during mesh separation.
#5) Here the subject name is extracted from the filename.txt file and the mesh is read via gmsh.
6) The main mesh separating function is called here and the .msh file is written in the current directory after separation.
#7) In this section the bash script – image_elec_field.sh is called which in turn calls the visual_check2.geo function that is used to customize the mesh viewing options and same the separated mesh files at different angles in .PNG format. The .mesh file is saved in the following views – front view, below (bottom view), left side, right side, front, back view.
8) This is the main mesh separating function – separating_meshes_func(new_tail, r).

A brief description of the mesh variables are given below –

MSH ASCII file format – This is the format of the file used for mesh separation. This file contains the information regarding the file ($meshformat) and several other options regarding the nodes ($nodes), elements ($elements), region names ($physicalname), periodicity relations ($periodic) and several post processing datasets ($NodeData, $ElementData, $elementNodeData)
Merge char – expression: Merges a file names char – expression.
read_msh – reads the .msh file.
$Elements –
Node_number_list: list of node numbers of the nth element. The order of the nodes is given in node ordering.
elm_type: defines the geometric type of the n-th element. (Here since, we are just extracting the

surfaces for separating the meshes for image generation, triangle geometric type is used, and hence element type is 2 – 3 node triangles).
elm_number: this is the index number of the nth element in the mesh. The element number must be a positive (non - zero) integer. It must be kept in mind that the element number need not necessarily be a dense or an ordered sequence.
elm_nr: referring to the total number of elements in the mesh (Alias to number of elements).
number_of_elements: this refers to the number of elements in the mesh and should be equal to elm_nr.
Tags: This gives the number of integer tags that are associated with the n - th element. By default the first tag (tag1) is the number of physical entity to which the element belongs. The second tag (tag2) is the number of elementary geometrical entity to which the element belongs.
$Nodes -
node_coord : This is a 4 column matrix, which stores the coordinates of the element nodes. The 4th column of the matrix is '0' if the element type is 2 (triangle) and used here for surface element extraction.
nr : stores the number of nodes in the element list and is the alias to the number of nodes.
node_number : the nodes in the mesh are numbered from 1 to the number of nodes in the element list. This variable stores this value of the node.
$ElementData – Here the post processing values are set for the separated meshes
Number_of_string_tags: gives the number of string tags. By default the first string tag refers to the post processing view and is set to 1 here. (Remains constant during the mesh separation and thus is not re-assigned)
String_tags : refers to the name of the post processing view.(Remains constant during the mesh separation and thus is not re-assigned).
Number_of_real_tags: Gives the number of real tags that follow. By default the first real tag is associated with the dataset. (Remains constant during the mesh separation and thus is not re-assigned).
Number_of_integer_tags: gives the number of integer tags that follow. By default the first integer tag refers to the time step index (starting at 0), the second as the number of field components of the data in the view (1, 3, or 9), the third as the number of entities (node or elements – here elements) in the view and the fourth as the partition index for the view data (0 for no partition).
Elm_number: refers to the element numbers and begins with 1 till the number of elements in the mesh.
Value: refers to the element values in the mesh.

name - this stores the name of the file.
fn - this variable stores the filename and the format of the file.

The list of variables used in the electric_field.py function is given below –

path : gets the entire path of the current file directory.
file_path_name : stores the entire path of the current directory.
concat_file : gets all the files with the .MSH file extension.
filename : stores all the .msh files (subject list).
filenames.txt : .TXT file that stores the subject ID list along with the entire path leading to the

directory.
head: stores the directory path of the subject.
tail: stores the subject ID (.MSH) format along with the format type.
new_tail: stores the subject ID sans the file format.
m : reads the .MSH file into the python function.
gm : stores the extracted mesh file in .MSH format.
cmd: stores the path to the bash code which calls the .geo for customizing the image view option.
p : call to the bash function.
out : stores the output from the bash code.
err : stores the error values from the bash code.
index_gm : stores the tag value for mesh separation ranging from 1 to 5.
node_number_list_orig : stores the values of the original mesh node list.
node_number_list_new0 : stores the values of the node list after extraction of the surface via the tag value.
idx_surface : stores the surface node list values of the extracted mesh.
node_number_list_new: stores the surface node list of the extracted mesh.
node index : stores the unique values of the node list of the extracted mesh.
node_number_list_new_new : ranks the node numbers of the extracted nodes of the mesh from 1 to the length of the node number.
elmtype : stores the value of the type of element for the extracted mesh.
tagtype : stores the value of the tag1 for the extracted mesh.
tagtype2 : stores the value of the tag2 for the extracted mesh.
Integertag0: stores the integer tag value of first view.
Integertag1: stores the integer tag value of second view.
Integertag2: stores the integer tag value of third view.
Integertag3: stores the integer tag value of fourth view.
Elmnumber0: stores the number of elements for the first view (starts from 0 till the number of elements in the mesh).
Elmnumber1: stores the number of elements for the second view (starts from 0 till the number of elements in the mesh).
Elmnumber2: stores the number of elements for the third view (starts from 0 till the number of elements in the mesh).
Elmnumber3: stores the number of elements for the fourth view (starts from 0 till the number of elements in the mesh).
Value0: stores the element values for the first viewing option.
Value1: stores the element values for the second viewing option.
Value3: stores the element values for the third viewing option.
Value4: stores the element values for the fourth viewing option.