

## **Efficient Evaluation of Epithelial/Absorbed Power Density by Multi-Antenna User Equipment with SAM Head Model**

This repository contains the MATLAB code developed for the work presented in the paper “Efficient Evaluation of Epithelial/Absorbed Power Density by Multi-Antenna User Equipment with SAM Head Model”. (<https://ieeexplore.ieee.org/document/10578308>)

### **Main Files**

- SDP\_CVX\_PSO\_1.m
- SDP\_CVX\_PSO\_2.m

### **Auxiliary Functions**


- is\_in\_tri.m
- is\_in\_uv.m
- obj\_vt.m
- uv2xyz.m

### **Instructions**

1. **Run obj\_vt.m first:** This script takes a parameterized 3D OBJ file and the corresponding 2D parameter space texture map image file as inputs. The output uv2xyz\_info.mat is a necessary input for the function uv2xyz.m.
2. **Texture Map Requirements:** If using parameterization methods like Free Border Surface Parameterizations (e.g., Least Squares Conformal Maps, LSTM), ensure that the texture map image file uses at least two different colors to distinguish between the inside and outside of the border. The default color for the outside of the border is rgb (50, 50, 50).
3. **3D Surface Parameterization:** For detailed 3D surface parameterization methods, refer to CGAL’s Surface Mesh Parameterization documentation ([https://doc.cgal.org/latest/Surface\\_mesh\\_parameterization/index.html](https://doc.cgal.org/latest/Surface_mesh_parameterization/index.html)). We have found that this C++ tool provides better results.
4. **Electromagnetic Field Data:** The scripts SDP\_CVX\_PSO\_1.m and SDP\_CVX\_PSO\_2.m correspond to equations (1) and (2) in the paper, respectively. The required electromagnetic field data input format is the same as the format exported by SEMCAD V19.2. Refer to the SEMCAD V19.2 documentation for specific details.

Part of SEMCAD V19.2 documentation:

## 2.8.10.2 MATLAB Exporter

The  [icon matlab exporter](#) MATLAB Exporter creates .mat files. Field arrays are stored as MATLAB arrays called *Snapshot0*, *Snapshot1*, ... at each frequency (or time step). The grid is stored either as rectilinear, defined by the MATLAB arrays *Axis0*, *Axis1*, *Axis2* or as an unstructured grid.

- Rectilinear Grid

The arrays contain the following data:

- *Snapshot0*, ...: Each line of the matrix contains the value of the exported field extracted from one cell of the grid. The number of lines corresponds to the number of cells in the employed field sensor. When exporting from the *Overall Field Sensor*, the values are stored by reading along the x-axis first, then y, then z.
- *Axis0*, *Axis1* and *Axis2*: 1-D arrays containing, respectively, the *x*, *y* and *z* coordinates of the grid lines. The number of grid lines is always one more than the number of cells. This implies for *Overall Field* that the length of *Axis0* is  $n_x + 1$ , *Axis1* is  $n_y + 1$  and *Axis2* is  $n_z + 1$ .

The values in *Snapshot0* are always evaluated at the cell centers. Hence, the first entry in *Snapshot0* is geometrically located at  $\left[ \frac{Axis0(1)+Axis0(2)}{2}, \frac{Axis1(1)+Axis1(2)}{2}, \frac{Axis2(1)+Axis2(2)}{2} \right]$ , the second entry of *Snapshot0* is located at  $\left[ \frac{Axis0(2)+Axis0(3)}{2}, \frac{Axis1(1)+Axis1(2)}{2}, \frac{Axis2(1)+Axis2(2)}{2} \right]$ , and so on.