# Introduction to the code

This depository contains implementations of the ADM method introduced in “The Auxiliary Dipole Method (ADM) for fast individualized coil placement for optimal TMS E-field targeting and dosaging,” available in biorxiv and under review in Neuroimage Journal. This document introduces the reader into the code requirements, as well as, provides some mathematical background on the method.

# System requirements

The ADM code requires 64-bit system and either windows, MacOS, or Linux operating systems. The code requires a Matlab installation. To run realistic head simulations a system with at least 8GB of ram is recommended. Finally, this code should work with most versions of Matlab, however, they have only been tested with Matlab 2010b, Matlab 2011a, and Matlab 2017-2019 versions. If you experience any problems please let me know the operating system and Matlab version, and error message so that I can provide a fix.

# Required data structures for running code

Each code requires a dipole representation of the coil, a tetrahedral mesh of the head, and observation points where the E-field is to be evaluated .

**Dipole data-structure (**, , , and **):** The dipole representation of the coil consists cartesian dipole locations , magnetic dipole vector, number of dipoles, and time derivative. For magnetic dipole models, the primary E-field is computed as

|  |  |  |
| --- | --- | --- |
|  |  |  |

where is the time derivative of the current pulse, nc is the number of dipoles, is the permeability of free-space, and and is the ith dipole cartesian location and dipole weight, respectively. For magnetic dipole models,

**Head mesh data-structure (** , and **):** The head mesh data structure is in the standard FEM mesh format, which consists of a tetrahedron to point array (), point array (), and conductivity array (). The point array has 3 dimensional entries with the cartesian location of one mesh point. Tetrahedron to point array has 4 dimensional entries **,** and the ith tetrahedron has vertices ,,, and . The conductivity of tetrahedron made of vertices is the ith entry of the array .

**ROIs data structures:** The following structures can be extracted from the mesh. The ROI is described by the tetrahedron ids () of the ROI (i.e. if is part of the ROI, then . A vector associated with each entry of . is the direction along which the E-field vector is calculated along in in its tetrahedron. Some functions do not require .

**Coil placement specific data structures:** The separation of the coil windings from the scalp (th\_hair); this is added to account for hair. The ADM implementation typically looks coil placements at scalp positions on a square patch centered about the ROI center of mass. The radius of the square patch is controlled by scth.

**Coil placement uncertainty quantification data structures:** To conduct UQ the coil positional uncertainty (or posuq) and coil orientation uncertainty (or anguq) needs to be specified.

# How to run the codes

Once the above data structures have been generated look for the ’testruncode.m’ and modify to pass the correct data structures. For more advanced users, ‘runcode.m’ is written in a modular fashion to enable users to derive their own modules from them. All of these codes work with electrical dipoles added documentation is forthcoming.