

The Matlab-SimBio Interface

Felix Lucka

November 3, 2011

This this will become a proper documentation to the Matlab-SimBio Interface (SBI) in the future.

1 What Can this Toolbox do for me?

- Access SimBio forward computation in an easy fashion.
- Functions to work with SimBio FEM head models:
 - Reading and writing all the files used for SimBio: `.v`, `.geo`, `.knw`, `.elc`, `.pts`, `.msr`,...
 - All model related data is available as matlab data types, additional information is computed (surfaces, adjacency matrices...).
 - Functions to create source spaces: Voxel based (future), surface based and grid based.
- Elementary plotting functions using Matlab and sophisticated ones using SCIRun (future).
- Tools for the validation of inverse methods: Generation of reference source configurations and simulated data with several constraints, computation of validation measures (future).
- and other stuff...

2 Preparations

Compiling SimBio: You need to have a binary of SimBio that runs on your system. Information how to get it, can be found [here](#).

Setting Up the Directory Structure for the Interface: At the moment, the toolbox relies on a fixed directory structure to work properly. When you download the toolbox, these are named SimBioInterface and SimBioData. Put them somewhere where it is convenient for you:

1. SimBioInterface is the root folder of the matlab toolbox: This folder contains all the matlab code and scripts. After running `startup.m`, its path (and all the subdirs) is added to the matlab path and is stored in the variable `SBI_path`. Make sure not to change the directory structure and names. Whenever the toolbox moves to a different directory to perform a certain task, it will jump back to `SBI_path`.
2. SimBioData is the root folder of the model data directory: This folder contains all the data related to concrete head models (when you download it, it will only contain the example data). The file `startup.m` (which is called automatically by matlab whenever you start matlab from `SBI_path`) tries to identify the path of this directory over the hostname of your machine and store it in the variable `data_path`. Please edit `startup.m` by adding the machines you use and the corresponding paths. The data directory itself contains a folder called `bin/` and folders which each contain one head model. Place a link or a copy of your SimBio binary in `bin/`. At the moment, the whole toolbox aims to use the Venant approach for forward modeling, so place `ipm_linux_opt_Venant` there (you can actually use other forward approaches and don't need to stick to that fixed notation, but it's not documented here). The names of the folders containing the head models identify these head models in a unique way, i.e., they match the names that will be used by SBI, and if the vista

format is used, the corresponding vista file needs to have the same name (with the `.v` ending). If the `.geo` format is used, the `.geo` and `.knw` files need to be named `ca_main.geo` and `ca_perm.knw`.

Compiling the Vista Readers The toolbox relies on some `.mex` files to read and write the vista format. Call `startup.m` if it is not called automatically. It will have a look if the mex files for your system are compiled. Otherwise call `sb_compile_vista`. If you do not succeed, try to compile other mex files first. I had [this](#) error on my ubuntu system.

3 Getting Started

Once you did all the above steps, open the file `TestAllFunctionsScript.m` in the scripts directory of `SBI_path`. It checks if all the functions of the toolbox work with all three possible FEM mesh types: Tetrahedra, hexahedra and geometry adapted hexahedra (node shifted). It also gives you a good overview over the current features of the toolbox. The meshes it uses are low resolution, unrealistic ones, such that it runs faster. However, the functions are not aimed to work as fast as possible with such meshes, but rather aimed at larger ones.

4 The General Structure

Get Functions The data of the model is stored in files in fixed locations. The most easy way to access it is to use the Get-functions located in `SBI_path/code/get_functions/`, which all have the syntax `variable = GetVariable(model);`

```
1 nodes = GetNodes(model);  
2 labels = GetLabels(model);
```

5 Function Reference