Structures and Handles - Structure of Handles

Author: Rajesh Bhaskaran, Cornell University

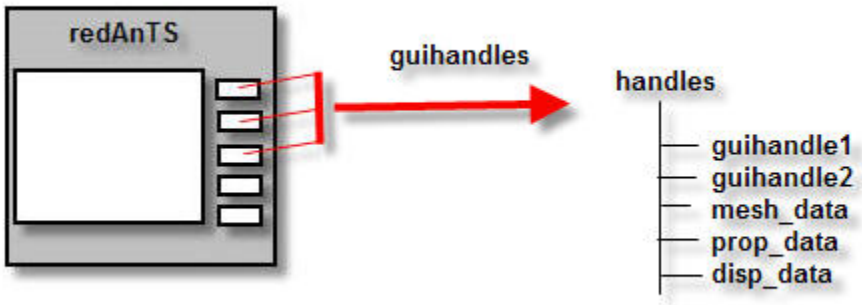
Structure of Handles

Preparation

Before going through this step, make sure you go through the first five steps of the redAnTS Tutorial #1and calculate the nodal displacements. If you have saved the mesh and input data from the tutorial previously, launch *redAnTS*, read in the mesh and input data and solve to obtain the nodal displacements. Leave the *redAnTS* GUI running.

Introduction

MATLAB assigns a handle to every graphics object it creates. One can use this handle to change the object's properties. *redAnTS* creates a structure, called the "handles structure", that contains all the handles of the objects in its graphical user interface (GUI). It then augments this "handles structure" by adding fields corresponding to the mesh, material properties, nodal displacements, element strains, etc. In other words, the handles structure is the container that is used to store this FEA data. The handles structure is illustrated in the figure below.



In the above figure, the fields *guihandle1* and *guihandle2* are handles for "objects" in the *redAnTS* GUI. The field *mesh_data* contains the mesh information (the x and y coordinates of the nodes etc.) and is, in turn, a structure. The field *prop_data* contains the material properties (E, nu) and is also a structure. Yada yada yada.

Handles structure in *redAnTS*

Within your redAnTS folder, there is a folder called *PostProcessMenuFiles* and within that folder, a file called *CalcStrainStress.m*. Bring up *CalcStrainStress .m* in the MATLAB editor. This function is used to calculate the element strain and stress values from the nodal displacements. This is one of the functions that you need to modify to add post-processing capabilities to your version of *redAnTS*. Let's take a closer look at the following two statements in this function.

```
function sigepsstruct=CalcStrainStress(fighandle)
handles=guidata(fighandle);
```

The variable *fighandle* is the handle to the main *redAnTS* window and is passed into this function. The second statement passes this handle on to the *guida ta* command to create a variable called *handles*. This variable *handles* is our exalted handles structure associated with the *redAnTS* GUI (bow to it a few times in respect). Its fields contain all data required to calculate the element strain and stress as discussed below.

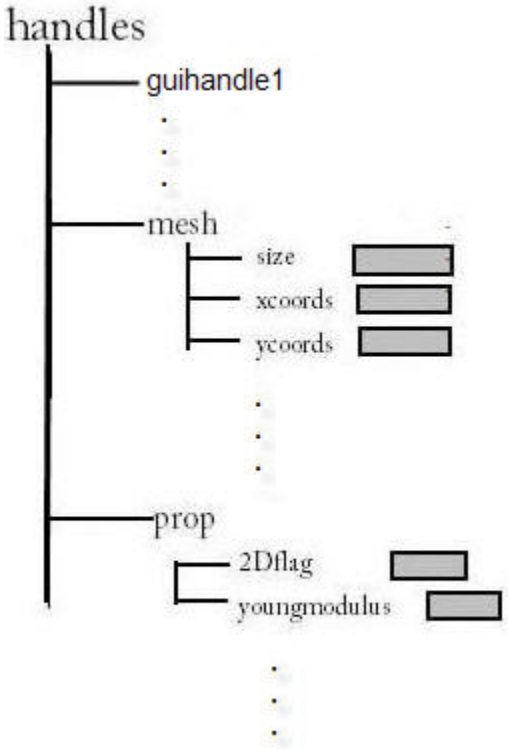
The data fields of interest in the handles structure are listed in the *redAnTS* main help. To see this, type `help redAnTS` at the MATLAB command line. Below is a snapshot from the help.

```
All data is stored in a structure called handles.
The structure can be accessed by using the command:
handles=guidata(FIGHANDLE);
where FIGHANDLE is the handle to the main GUI figure window. That handle
is stored in handles.mainguifig and can be passed as an input argument to
functions.

The data fields of interest in the handles structure are as follows:

    handles.mesh        : contains all mesh information including the fields
        handles.mesh.size    : [number of elements ,
                                number of global nodes ,
                                number of nodes per element ,
                                global number of degrees of freedom ,
                                number of degrees of freedom per element]
        handles.mesh.xcoords: [1 x (number of nodes)] : X coordinate of each node
        handles.mesh.ycoords: [1 x (number of nodes)] : Y coordinate of each node
        handles.mesh.connect: [(number of elements) x 3] : connectivity array
```

Thus, the *handles* structure has a field called *mesh* that is also a structure. The *mesh* field in turn contains the *xcoords* and other fields. The statement

```
x= handles.mesh.xcoords
```

will give you a row vector containing the x-coordinates of all the nodes. You can visualize the handles structure as per the figure below.

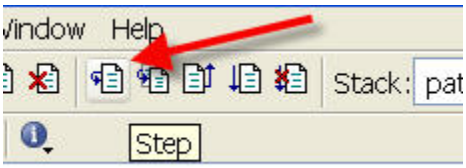


Investigate the *handles* structure using the debugger

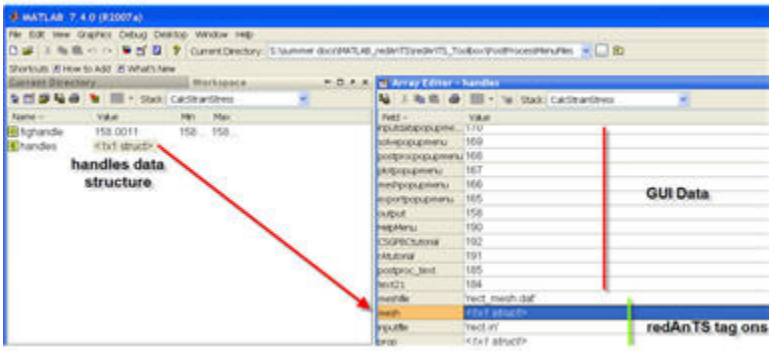
You can use the debugger to investigate further what the fields of the *handles* structure look like. In the MATLAB editor, add a breakpoint in *CalcStrainStress.m* at the statement `handles=guidata(fighandle).` You can add a breakpoint by clicking to the left of the statement. A red circle should appear as shown below.

```
20  handles=guidata(fighandle);
```

Now we need to drive *redAnTS* into *CalcStrainStress.m* function so that it will stop execution at the breakpoint that we just set. *CalcStrainStress.m* is invoked when we calculate the stress components in the *redAnTS* GUI. To calculate the stress components, go to the *redAnTS* GUI and click on ***Strain***, ***Stress*** under ***Post-Process***. Program execution should stop at the above statement. Check the Workspace. Then, step through this statement using the step icon.



In the Workspace, you should see that *handles* structure has been created. Double-click on this variable to see what the fields of the structure look like. Reconcile what you see with the fields listed in the main *redAnTS* help page.

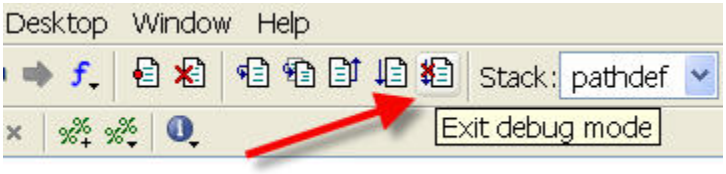


Type the following at the command line to see what the nodal x-coordinates are:

```
x= handles.mesh.xcoords
```

Note that the strain and stress fields, *eps* and *sig*, respectively, don't yet exist. When you update the structure *sigepsstruct* in *CalcStrainStress.m*, these fields will be added to the handles structure by the function that calls *CalcStrainStress.m*.

After you are done investigating the *handles* structure in the debugger, exit the debug mode by clicking on the ***Exit debug mode*** icon in the editor.



You should now have enough background to provide post-processing capabilities in redAnTS by modifying the following functions:

CalcStrainStress.m
CalcPrincipals.m
CalcEffectiveStress.m

Happy computing!

Go to Comments

Go to all MATLAB Learning Modules