Structures and Handles - Creating a structure for one element

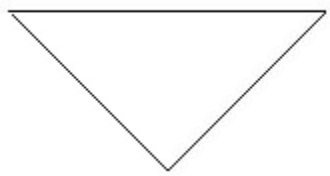
Author: Rajesh Bhaskaran, Cornell University

Overview
1. Creating a structure for one element
2. Creating a structure for two elements
3. Structure of handles
Comments

Creating a Structure for One Element

Background on Structures

Structures are a convenient way to organize data. To demonstrate how structures work, consider the triangular element below with three nodes, one at each corner.

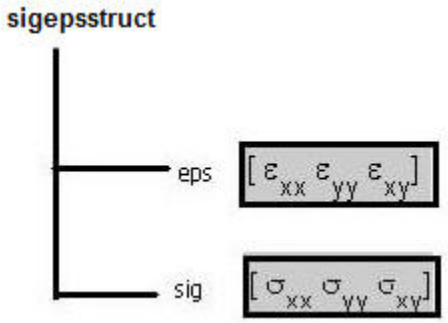


For this element, the strain and stress are constant over the element. Thus, the element has three components each of strain and stress associated with it as shown below.

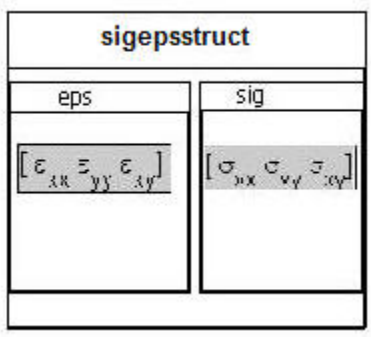
$$\varepsilon = [\varepsilon_{xx}\ \varepsilon_{yy}\ \varepsilon_{xy}]$$

$$\sigma = [\sigma_{xx}\ \sigma_{yy}\ \sigma_{xy}]$$

There are many ways one can store this information. One simple but elegant way to do this would be to create a data structure that has two data containers (or fields) as shown below. The first field contains the three strain components and is labeled *eps*. The second field contains the three stress components and is labeled *sig*. We label the data structure *sigepsstruct*. All these labels are consistent with *redAnTS*.



In the above representation, *sigepsstruct* is refered to as a structure with two fields: *eps* and *sig*. A useful way to visualize this is that *sigepsstruct* is a box that contains two smaller boxes, *eps* and *sig*, which hold the strain and stress values, respectively. So if I want, say, the stress values, I ask MATLAB to open the *sig* box within *sigepsstruct* and look inside.



Create a structure

Let's create the above structure in MATLAB. Start MATLAB. Create a working directory at an appropriate location using the **Current Directory** window. If you cannot see this window, select **Main Menu > Desktop > Current Directory**. You can right-click in this window to access the option for creating a new folder (i.e. directory). Browse to this folder to make this your working directory.

To figure out how to create a structure, let's turn to the help pages. After poking around in the help browser, I found the following useful introduction to structures: **Help > Contents >MATLAB > Programming > Data Types > Structures**. Bring up this page in the help browser. You can alternately view it in your web browser by clicking here. Read through this page until and including the topic *Building Structure Arrays Using the struct Function*. Yes, I mean actually read the manual; real women and men do it.

Let's use the information gleaned from this page to create our structure using the *struct* function. The help says that the format for using the *struct* function is

```
strArray = struct('field1',val1,'field2',val2, ...)
```

Bring up the editor by clicking on the New M-file icon (or any other preferred way). Copy the above *struct* statement into the M-file and modify it to create the *sigepstruct* structure.

```
sigepsstruct = struct('eps', eps_val, ...
                      'sig', sig_val);
```

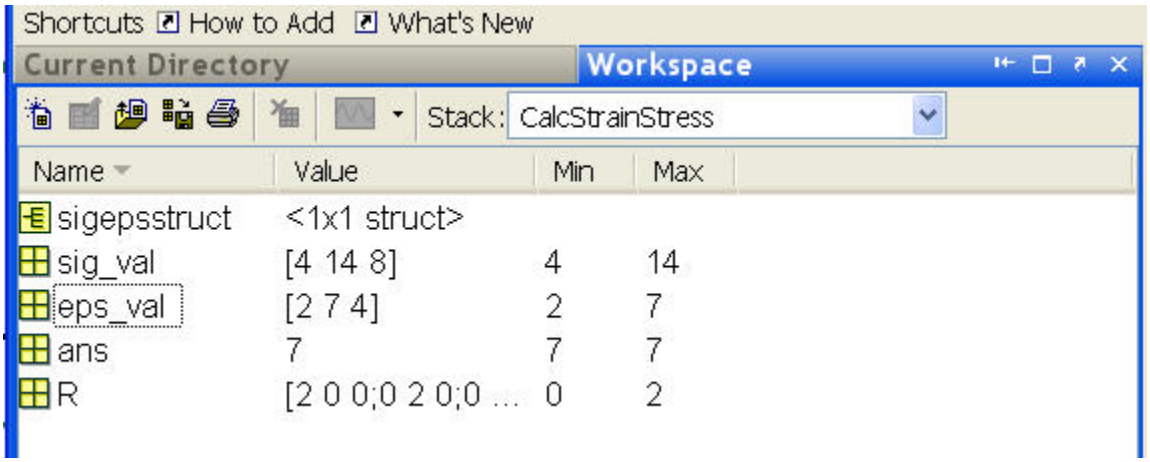
We use "..." to start a new line. In order for this to work, we, of course, need to allocate strain and stress values to *eps_val* and *sig_val*, respectively, *above*t his statement. We'll use dummy values that would horrify any self-respecting solid mechanics professor. We do so in order to make the programming aspects easier to understand.

```
%Testing the use of structure
clear all; %Clear all variables from workspace
eps_val = [2 7 4];%Nonsense values
R = 2*eye(3); %Dummy R
sig_val = (R*eps_val')';%Derived nonsense
```

In your actual *redAnTS* implementation, the strains will be derived from the displacements that *redAnTS* provides from inversion of the stiffness matrix. And you'd have a meaningful *R* matrix. Note that we specify *eps_val* as a row vector; we have to use the transpose operator (') to ensure we end up with a row vector for *sig_val* also.

Save and run your M-file.

The **Workspace** window shows all the currently defined variables. If you don't see this window, select **Main Menu > Desktop > Workspace** so that a tick mark appears next to the window name. Double-click on any variable name in the *Workspace* to take a closer peek at it. Double-clicking on *sigepsstruct* shows the fields this structure contains.



Note: The fields of structures can hold any type of data, text or even other structures, not just numbers.

We have now created our structure. Let's now see how we can extract parts of the structure. Since you are a real woman/man, go back to the above help page and read the section on *Accessing Data in Structure Arrays*. It tells you that to access a field of a particular structure, include a period (.) after the structure name followed by the field name. Use this to extract the entire stress vector and, say, the second item in the *eps* field (which is $_{yy}$).

```
%Extract data from structure
sigepsstruct.sig %Extract the stress vector
sigepsstruct.eps(2)%Extract the 2nd element of eps
```

Check in the command window that the correct values are extracted.

The final code for creating the structure and extracting data from it should look like this:

I hear you saying "this is a piece of cake, gimme more!". So let's move on to Step 2.

Go to Step 2: Creating a structure for two elements

Go to all MATLAB Learning Modules