Help on Problem 11.2:

Full credit will be given on this problem if you demonstrate the use of the **`set`** method. You may omit the dot and structure methods, which are less preferable methods (as of this date). Details follow. . .

When MATLAB first came out in the 1980's it was designed to operate under the Microsoft DOS operating system, not Microsoft Windows. DOS does not support objects. Objects were possible if the program provided the support—but that was not very practical. Therefore plots, figures, etc. were not objects in the DOS versions of MATLAB. Properties such as colors, line-types (solid, dashed, dotted, etc.) were passed to the plotting procedure as parameters, just as you would pass parameters to any function. This method is still supported and if it meets your need, Professor De Boer recommends you use it because it executes quickly, is highly compatible with many programs beside MATAB (GNU Octave is just one example) and is easy-to-understand.

The set method

All modern versions of MATLAB treat figures, plots, etc. as objects. This provides a level of control and interactivity that cannot be achieved otherwise. When these figure and plot objects were first introduced in MATLAB, the object properties had to be manipulated via the **`get`** and **`set`** methods. (And there are a few other sometimes-helpful methods. Page 370 shows a typical list of available methods for the Figure object.) Pages 368 through the top of 370 discuss this type of control of a MATLAB object. In particular, Problem 11.2 is asking you to use the **`set`** method along with the plot's handle, the property, and the desired value of the property to change the color of the plot-line. This is the only aspect of this problem that you need to do. The dot and structure methods are optional.

Suppose for a moment that the **`get`** and **`set`** (and related) methods must always be used in order to manipulate an object's properties. Then the internal structure of the binary coding of the properties does not matter. MATLAB can encode the properties one way and GNU Octave can encode the properties another way. Everything will be fine so long as MATLAB's version of **`get`** and **`set`** (and related) methods knows MATLAB's encoding of the properties and GNU Octave's version of the **`get`** and **`set`** (and related) methods know GNU Octave's encoding. Suppose MATLAB uses a structure to encode the properties (it does) and suppose GNU Octave uses some sort of cell array. Both work just fine and code that runs on MATLAB will run on GNU Octave too. In the world of object-oriented programming it is the norm to always manipulate an object's properties via the object's methods.

The dot method

Beginning with release 2014b MATLAB allowed direct manipulation of object properties using the dot method. This works fine, provided you know the internal arrangement of the properties of the object. This method is described on the bottom half of page 370. But GNU Octave does not yet support this method, and they probably have some work to do to make it fully compatible with newer versions of MATLAB because of the different internal coding of the object properties in GNU Octave. Using the dot method this way is an overload of the dot operator because the operator is acting on a property of an object, whereas it was originally defined to operate on a structure of data in memory. Using the dot method on an object's properties is a convenience for the programmer, but it is outside the norm for the concept of object-oriented programming.

The structure method

If one really wants to use the dot method with an earlier version of MATLAB or with GNU Octave, one can use **`get`** to copy all the object's properties into a structure variable. Then one can manipulate the structure variable with the dot method. However, that alone does nothing to the object (the plot) since the structure variable is outside the object. One needs to follow up by using the **`set`** method to write the changed structure variable back into the object' properties. But that is a problem. If any of the properties of the object are read-only (and practically every object will have some read-only properties) then writing the structure variable back in its entirety will fail. One needs to write back only the writable property(s) of interest.