**Structures**

A structure is a data type which contains several values, possibly of different types, referenced by name. The simplest way to create a structure is by simple assignment. For example, consider the function

displaymath783

The following m-file f.m computes the value, gradient, and Hessian of *f* at a point *x*, and returns them in a structure:

function fx = f(x)

fx.Value = (x(1)-1)^2+x(1)\*x(2);

fx.Gradient = [2\*(x(1)-1)+x(2);x(1)];

fx.Hessian = [2 1;1 0];

We can now use the function as follows:

>> x = [2;1]

x =

2

1

>> fx = f(x)

fx =

Value: 3

Gradient: [2x1 double]

Hessian: [2x2 double]

>> whos

Name Size Bytes Class

fx 1x1 428 struct array

x 2x1 16 double array

Grand total is 12 elements using 444 bytes

The potential of structures for organizing information in a program should be obvious.

Note that, in the previous example, Matlab reports fx as being a tex2html_wrap_inline635``struct array''. We can have multi-dimensional arrays of structs, but in this case, each struct must have the same field names:

>> gx.Value = 12;

>> gx.Gradient = [2;1];

>> A(1,1) = fx;

>> A(2,1) = gx;

??? Subscripted assignment between dissimilar structures.

>> fieldnames(fx)

ans =

'Value'

'Gradient'

'Hessian'

>> fieldnames(gx)

ans =

'Value'

'Gradient'

(Note the use of the command fieldnames, which lists the field names of a structure.)

Beyond simple assignment, there is a command struct for creating structures. For information on this and other commands for manipulating structures, see help struct.

*Mark S. Gockenbach   
Wed Sep 8 10:44:13 EDT 1999*