control the CurveFit, FuncFit and FuncFitMD operations. The use of these variables is documented along with the operations that they affect.

When an Igor operation creates V_ and S_ variables, they are global if the operation was executed from the command line and local if the operation was executed in a procedure. See **Accessing Variables Used by Igor Operations** on page IV-123 for details.

Numeric Variables

You create numeric user variables using the `Variable` operation from the command line or in a procedure. The syntax for the Variable operation is:

```
Variable [flags] varName [=numExpr] [,varName [=numExpr]]...
```

There are three optional flags:

/C	Specifies complex variable.
/D	Obsolete. Used in previous versions to specify double-precision. Now all variables are double-precision.
/G	Specifies variable is to be global and overwrites any existing variable.

The variable is initialized when it is created if you supply the initial value with a numeric expression using `=numExpr`. If you create a numeric variable and specify no initializer, it is initialized to zero.

You can create more than one variable at a time by separating the names and optional initializers for multiple variables with a comma.

When used in a procedure, the new variable is local to that procedure unless the /G flag is used. When used on the command line, the new variable is always global.

Here is an example of a variable creation with initialization:

```
Variable v1=1.1, v2=2.2, v3=3.3*sin(v2)/exp(v1)
```

Since the /C flag was not specified, the data type of v1, v2 and v3 is double-precision real.

Since the /G flag was not specified, these variables would be global if you invoked the `Variable` operation directly from the command line or local if you invoked it in a procedure.

`Variable/G varname` can be invoked whether or not a variable of the specified name already exists. If it does exist as a variable, its contents are not altered by the operation unless the operation includes an initial value for the variable.

To assign a value to a complex variable, use the `cmplx()` function:

```
Variable/C cv1 = cmplx(1,2)
```

You can kill (delete) a global user variable using the Data Browser or the `KillVariables` operation. The syntax is:

```
KillVariables [flags] [variableName [,variableName]...]
```

There are two optional *flags*:

/A	Kills all global variables in the current data folder. If you use /A, omit variableName.
/Z	Doesn't generate an error if a global variable to be killed does not exist.

For example, to kill global variable `cv1` without worrying about whether it was previously defined, use the command:

```
KillVariables/Z cv1
```