

above using the specified accuracy. If you need functions that are not supported by this operation, you may have to precompute them and store the results in a local variable.

The operation stores the result in *destStr*, which may or may not exist prior to execution. When you execute the operation from the command line, *destStr* becomes a global string in the current data folder if it does not already exist. If it exists, then the result of the operation overwrites its value (as with any normal string assignment). In a user function, *destStr* can be a local string, an SVAR, or a string passed by reference. If *destStr* is not one of these then the operation creates a local string by that name.

Arbitrary precision math calculations are much slower (by a factor of about 300) than equivalent floating point calculations. Execution time is a function of the number of digits, so you should use the /N flag to limit the evaluation to the minimum number of required digits.

Output Variables

In Igor Pro 8.00 or later, the APMath operation returns information in the following output variables:

V_Flag	Set to 0 if the operation succeeds or to a non-zero error code.
V_Value	Set to a double-precision representation of the output string.

Examples

Evaluate pi to 50 digits:

```
APMath/V aa = pi
```

Evaluate ratios of large factorials:

```
APMath/V aa = factorial(500)/factorial(499)
```

Evaluate ratios of large exponentials:

```
APMath/V aa = exp(-1000)/exp(-1001)
```

Division of mixed size values:

```
APMath/V aa = 1-sgn(1-(1-0.00000000000000000001234)/(1-0.000000000000000000012345)))
```

you'll get a different result trying to evaluate this using double precision.

Difference between built-in pi and the arbitrary precision pi:

```
Variable/G biPi = pi
APMath/V aa = biPi-pi
```

Precision control:

```
Function test()
    APMath aa = pi                // Assign 50 digit pi to the string aa.
    APMath/V bb = aa              // Create local string bb equal to aa.
    APMath/V bb = aa-pi           // Subtract arb. prec. pi from aa.
                                // note the default exDigits=6.
    APMath/V/N=50/ex=0 bb = aa-pi // setting exDigits=0.
End
```

Numerical recreation:

```
APMath/V/N=16 aa = 111111111^2
```

The solution for the sum of three cubes problem for the number 42:

```
APMath/V aa = pow((-80538738812075974),3) + pow(80435758145817515,3) +
    pow(12602123297335631,3)
```

Append

Append

The Append operation is interpreted as **AppendToGraph**, **AppendToTable**, or **AppendToLayout**, depending on the target window. This does not work when executing a user-defined function. Therefore we now recommend that you use **AppendToGraph**, **AppendToTable**, or **AppendLayoutObject** rather than Append.