Although it is mathematically suspect, rectangular integration using /METH=0 would be correct if the X scaling of the output wave is offset by ΔX.

`Differentiate/METH=1/EP=1` is the inverse of `Integrate/METH=2`, but `Integrate/METH=2` is the inverse of `Differentiate/METH=1/EP=1` only if the original first data point is added to the output wave.

Integrate applied to an XY pair of waves does not check the ordering of the X values and doesn't care about it. However, it is usually the case that your X values should be monotonic. If your X values are not monotonic, you should be aware that the X values will be taken from your X wave in the order they are found, which will result in random X intervals for the X differences. It is usually best to sort the X and Y waves first (see **Sort**).

See Also

Differentiate, **Integrate2D**, **Integrate1D**, **area** , **areaXY**

Integrate1D

Integrate1D(*UserFunctionName*, *min_x*, *max_x* [, *options* [, *count* [, *pWave*]]])

The Integrate1D function performs numerical integration of a user function between the specified limits (*min_x* and *max_x*).

Parameters

UserFunctionName must have this format:

```
Function UserFunctionName(inX)
   Variable inX
   ... do something
   return result
End
```

However, if you supply the optional *pWave* parameter then it must have this format:

```
Function UserFunctionName(pWave, inX)
   Wave pWave
   Variable inX
   ... do something
   return result
End
```

options is one of the following:

0:	Trapezoidal integration (default).
1:	Romberg integration.
2:	Gaussian Quadrature integration.

By default, *options* is 0 and the function performs trapezoidal integration. In this case Igor evaluates the integral iteratively. In each iteration the number of points where Igor evaluates the function increases by a factor of 2. The iterations terminate at convergence to tolerance or when the number of evaluations is 2^{23}.

The *count* parameter specifies the number of subintervals in which the integral is evaluated. If you specify 0 or a negative number for count, the function performs an adaptive Gaussian Quadrature integration in which Igor bisects the interval and performs a recursive refining of the integration only in parts of the interval where the integral does not converge to tolerance.

pWave is an optional parameter that, if present, is passed to your function as the first parameter. It is intended for your private use, to pass one or more values to your function, and is not modified by Igor. The *pWave* parameter was added in Igor Pro 7.00.

Details

You can integrate complex-valued functions using a function with the format:

```
Function/C complexUserFunction(inX)
   Variable inX
   Variable/C result
   //… do something
   return result
End
```

The syntax used to invoke the function is:

```
Variable/C cIntegralResult=Integrate1D(complexUserFunction,min_x,max_x…)
```