You can also use Integrate1D to perform a higher dimensional integrals. For example, consider the function: $F(x,y) = 2x + 3y + xy$.

In this case, the integral

$$h = \int dy \int f(x,y)dx$$

can be performed by establishing two user functions:

```
Function Do2dIntegration(xmin,xmax,ymin,ymax)
	Variable xmin,xmax,ymin,ymax

	Variable/G globalXmin=xmin
	Variable/G globalXmax=xmax
	Variable/G globalY

	return Integrate1d(userFunction2,ymin,ymax,1)  // Romberg integration
End

Function UserFunction1(inX)
	Variable inX

	NVAR globalY=globalY

	return (3*inX+2*globalY+inX*globalY)
End

Function UserFunction2(inY)
	Variable inY

	NVAR globalY=globalY
	globalY=inY
	NVAR globalXmin=globalXmin
	NVAR globalXmax=globalXmax

	// Romberg integration
	return Integrate1D(userFunction1,globalXmin,globalXmax,1)
End
```

This method can be extended to higher dimensions.

If the integration fails to converge or if the integrand diverges, Integrate1D returns NaN. When a function fails to converge it is a good idea to try another integration method or to use a user-defined number of intervals (as specified by the count parameter). Note that the trapezoidal method is prone to convergence problems when the absolute value of the integral is very small.

See Also

Integrate, **Integrate2D**, **SumSeries**

Integrate2D

`Integrate2D` `[`***`flags`***`] [`***`keyword = value`***` [, `***`keyword = value`***` …]]`

The Integrate2D operation calculates a two-dimensional numeric integral of a real-valued user-defined function or a wave. The result of the operation is stored in the variable V_value and the variable V_Flag is set to zero if there are no errors.

This operation was added in Igor Pro 7.00.

Flags

/OPTS=*op*	Sets the integration options. By default, both the x and the y integrations are performed using the adaptive trapezoidal method.

op is a bitwise parameter that you set to select the x and y integration methods. Set one bit for x and one bit for y:

Bit 0:	Trapezoidal in Y (1)
Bit 1:	Romberg in Y (2)
Bit 2:	Gaussian Quadrature in Y (4)
Bit 3:	Trapezoidal in X (8)
Bit 4:	Romberg in X (16)
Bit 5:	Gaussian Quadrature in X (32)

See **Setting Bit Parameters** on page IV-12 for details about bit settings.

Using these constants you can specify, for example, Romberg integration in the Y direction and Gaussian Quadrature in the X direction using /OPTS=(2 | 32).

/Q Suppress printing to the history area.

/Z=zFlag Set zFlag to 1 to suppress error reporting.

Keywords

epsilon=*ep* Specifies the convergence parameter. By default *ep*=1e-5. Smaller values lead to more accurate integration result but the tradeoff is longer computation time.

integrand=*uF* Specifies the user function to be integrated. See **The Integrand Function** below for details.

innerLowerLimit=*y1* Specifies the lower limit of the inner integral if this limit is fixed, i.e., if it is not a function of x. See the innerLowerFunc keyword if you need to specify a function for this limit.

innerUpperLimit=*y2* Specifies the upper limit of the inner integral if this limit is fixed, i.e., if it is not a function of x. See the innerUpperFunc keyword if you need to specify a function for this limit.

innerLowerFunc=*y1Func* Specifies a user-defined function for the lower limit of the inner integral. See **The Limit Functions** below.

innerUpperFunc=*y2Func* Specifies a user-defined function for the upper limit of the inner integral. See **The Limit Functions** below.

outerLowerLimit=*x1* Specifies the lower limit of the outer integral.

outerUpperLimit=*x2* Specifies the upper limit of the outer integral.

paramWave=*pWave* Specifies a wave to be passed to the integrand and limit user-defined functions as the pWave parameter. The wave may contain any number of values that you might need to evaluate the integrand or the integration limits.

If you omit paramWave then the pWave parameter to the functions will be NULL.

srcWave=*mWave* If you need to perform 2D integration of some data, you can specify the data directly instead of providing a user-defined function that returns interpolated data. mWave must be a 2D wave. Higher dimensional waves are accepted but only the first layer of the wave is used in the integration.

The Integrand Function

Integrate2D computes the general two-dimensional integral of a user-defined integrand function which you specify using the integrand keyword. The integrand function has this form:

```
Function integrandFunc(pWave,inX,inY)
   Wave/Z pWave
   Variable inX,inY
```

```
    ... do something
    return result
End
```

The function can have any name - integrandFunc is just an example. The function must take the parameters shown and must return a real numeric result. Returning a NaN terminates the integration.

pWave is a parameter wave that you specify using the paramWave keyword. The operation passes this wave on every call to the integrand function. If you omit paramWave when invoking Integrate2D then pWave will be NULL.

The Limit Functions

The limit functions provide lower and/or upper limits of integration for the inner integral if they are functions of x rather than fixed values. You specify a limit function using the innerLowerFunc and innerUpperFunc keywords. The form of the limit function is:

```
Function limitFunction(pWave,inX)
    Wave/Z pWave
    Variable inX

    ... do something
    return result
End
```

Details

The operation computes the general two-dimensional integral of the form

$$I = \int_{x1}^{x2} dx \int_{y1(x)}^{y2(x)} f(x,y)dy.$$

Here y1 and y2 are in general functions of x but could also be simple constants, and f(x,y) is real valued function. The integral is evaluated by considering the "outer" integral

$$I = \int_{x1}^{x2} G(x)dx,$$

where G(x) is the "inner" integral

$$G(x) = \int_{y1(x)}^{y2(x)} f(x,y)dy.$$

The operation allows you to specify different algorithms for integrating the inner and outer integrals. The simplest integration algorithm is the Trapezoidal method. You can typically improve on the accuracy of the calculation using Romberg integration and the performance of Gaussian quadrature depends significantly on the nature of the integrand.

Example 1: Integrating a 2D function over fixed limits

Suppose we wanted to check the normalization of the built-in two-dimensional Gauss function. The user-defined function would be:

```
Function myIntegrand1(pWave,inX,inY)
    Wave/Z pWave
    Variable inX,inY
    return Gauss(inX,50,10,inY,50,10)
End
```

To perform the integration, execute:

```
Integrate2D outerLowerLimit=0, outerUpperLimit=100, innerLowerLimit=0,
    innerUpperLimit=100, integrand=myIntegrand1
Print/D V_Value
```