

Gauss1D

Gauss1D(*w*, *x*)

The Gauss1D function returns the value of a Gaussian peak defined by the coefficients in the wave *w*. The equation is the same as the Gauss curve fit:

$$w[0] + w[1] \exp \left[- \left(\frac{x - w[2]}{w[3]} \right)^2 \right].$$

Examples

Do a fit to a Gaussian peak in a portion of a wave, then extend the model trace to the rest of the X range:

```
Make/O/N=100 junkg // fake data wave
Setscale/I x -1,1,junkg
Display junkg
junkg = 1+2.5*exp(-(x-.5)/.3)^2)+gnoise(.1)
Duplicate/O junkg, junkgfit
junkgfit = NaN
AppendToGraph junkgfit
CurveFit gauss junkg[50,99] /D=junkgfit
// now extend the model trace
junkgfit = Gauss1D(w_coef, x)
```

See Also

The **CurveFit** operation.

Gauss2D

Gauss2D(*w*, *x*, *y*)

The Gauss2D function returns the value of a two-dimensional Gaussian peak defined by the coefficients in the wave *w*. The equation is the same as the Gauss2D curve fit:

$$w[0] + w[1] \exp \left\{ \frac{-1}{2(1 - w[6]^2)} \left[\left(\frac{x - w[2]}{w[3]} \right)^2 + \left(\frac{y - w[4]}{w[5]} \right)^2 - \left(\frac{2w[6](x - w[2])(y - w[4])}{w[3]w[5]} \right) \right] \right\}.$$

Examples

Do a fit to a Gaussian peak in a portion of a wave, then extend the model trace to the rest of the X range (watch out for the very long wave assignment to junkg2D):

```
Make/O/N=(100,100) junkg2D // fake data wave
Setscale/I x -1,1,junkg2D
Setscale/I y -1,1,junkg2D
Display; AppendImage junkg2D
//Caution! Next command wrapped to fit page:
junkg2D = -1 + 2.5*exp((-1/(2*(1-.4^2)))*((x-.1)/.2)^2+((y+.2)/.35)^2+2*.4*
((x-.1)/.2)*((y+.2)/.35))
junkg2D += gnoise(.01)
Duplicate/O junkg2D, junkg2Dfit
junkg2Dfit = NaN
AppendMatrixContour junkg2Dfit
CurveFit gauss2D junkg2D[20,80][10,70] /D=junkg2Dfit[20,80][10,70]
// now extend the model trace
junkg2Dfit = Gauss2D(w_coef, x, y)
```

See Also

The **CurveFit** operation.

GBLoadWave

GBLoadWave [*flags*] [*fileNameStr*]

The GBLoadWave operation loads data from a binary file into waves.

For more complex applications such as loading structured data into Igor structures see the **FBinRead** operation.

Parameters

If *fileNameStr* is omitted or is "", or if the /I flag is used, GBLoadWave presents an Open File dialog from which you can choose the file to load.

If you use a full or partial path for *fileNameStr*, see **Path Separators** on page III-451 for details on forming the path.

Flags

/A	Automatically assigns arbitrary wave names using "wave" as the base name. Skips names already in use.
/A= <i>baseName</i>	Same as /A but it automatically assigns wave names of the form <i>baseName0</i> , <i>baseName1</i> .
/D= <i>d</i>	New programming should use the /T flag instead of the /D, /L and /F flags. <i>d</i> =0: Creates single-precision waves. <i>d</i> =1: Creates double-precision waves. /D by itself is equivalent to /D=1.
/F= <i>f</i>	New programming should use the /T flag instead of the /D, /L and /F flags. <i>f</i> specifies the data format of the file: <i>f</i> =1: Signed integer (8, 16, 32 bits allowed) <i>f</i> =2: Creates double-precision waves <i>f</i> =3: Floating point (default, 32, 64 bits allowed)
/FILT= <i>fileFilterStr</i>	Provides control over the file filter menu in the Open File dialog. This flag was added in Igor Pro 7.00. The construction of the <i>fileFilterStr</i> parameter is the same as for the /F= <i>fileFilterStr</i> flag of the Open operation. See Open File Dialog File Filters on page IV-149 for details.
/I [{ <i>macFilterStr</i> , <i>winFilterStr</i> }]	Specifies interactive mode which displays the Open File dialog. In Igor7 and later, the <i>macFilterStr</i> and <i>winFilterStr</i> parameters are ignored. Use the /FILT flag instead.
/J= <i>j</i>	Specifies how input floating point data is interpreted. <i>j</i> =0: IEEE floating point (default) <i>j</i> =1: VAX floating point
/L= <i>length</i>	New programming should use the /T flag instead of the /D, /L and /F flags. <i>length</i> specifies the data length of the data in the file in bits (default = 32). Allowable data lengths are 8, 16, 32, 64.
/N	Same as /A except that, instead of choosing names that are not in use, it overwrites existing waves.
/N= <i>baseName</i>	Same as /N except that it automatically assigns wave names of the form <i>baseName0</i> , <i>baseName1</i> .
/O= <i>o</i>	Controls overwriting of waves in case of a name conflict. <i>o</i> =0: Use unique wave names. <i>o</i> =1: Overwrite existing waves. /O by itself is equivalent to /O=1.
/P= <i>pathName</i>	Specifies the folder to look in for <i>fileNameStr</i> . <i>pathName</i> is the name of an existing symbolic path.

<code>/Q=q</code>	Controls messages written to the history area of the command window. <code>q=0:</code> Write messages. <code>q=1:</code> Suppress messages. <code>/Q</code> by itself is equivalent to <code>/Q=1</code> .
<code>/S=s</code>	<code>s</code> is the number of bytes at the start of the file to skip. It defaults to 0.
<code>/T={fType,wType}</code>	Specifies the data type of the file (<i>fType</i>) and the data type of the wave or waves to be created (<i>wType</i>). The allowed codes for both <i>fType</i> and <i>wType</i> are: 2: Single-precision floating point 4: Double-precision floating point 8: 8-bit signed integer 16: 16-bit signed integer 32: 32-bit signed integer 128: 64-bit signed integer (Igor7 or later) 72: 8-bit unsigned integer (8+64) 80: 16-bit unsigned integer (16+64) 96: 32-bit unsigned integer (32+64) 192: 64-bit unsigned integer (128+64) (Igor7 or later)
<code>/U=u</code>	Specifies the number of points of data per array in the file. The default is 0 which means "auto". In this case GBLoadWave calculate the number of data pointers per array based on the number of bytes in the file, the number of bytes to be skipped at the start of the file (<code>/S</code> flag), and the number of arrays in the file (<code>/W</code> flag).
<code>/V=v</code>	Specifies interleaving of data in the file. <code>v=0:</code> Data in file is not interleaved (default) <code>v=1:</code> Data in file is interleaved <code>/V</code> by itself is equivalent to <code>/V=1</code> .
<code>/W=w</code>	Specifies the number of arrays in the file. The default is 1. If you omit <code>/W</code> but specify the number of points per data array in the file via <code>/U</code> then GBLoadWave calculates the number of waves to be loaded based on the number of bytes in the file, the number of bytes to be skipped at the start of the file (<code>/S</code> flag), and the specified number of points per data array in the file (<code>/U</code> flag). Therefore, if you specify <code>/U</code> and want to load just one wave you must also specify <code>/W=1</code> .
<code>/Y={offset, mult}</code>	Data loaded into waves is scaled using offset and mult: $\text{output data} = (\text{input data} + \text{offset}) * \text{multiplier}$ This is useful to convert integer data into scaled, real numbers.

Details

The `/N` flag instructs Igor to automatically name new waves "wave" (or *baseName* if `/N=baseName` is used) plus a number. The number starts from zero and increments by one for each wave loaded from the file. If the resulting name conflicts with an existing wave, the existing wave is overwritten.

The `/A` flag is like `/N` except that Igor skips names already in use.

The `/T` flag allows you to specify a data type for both the input (data in the file) and the output (data in the waves). You should use the `/T` flag instead of the `/D`, `/L` and `/F` flags. These flags are obsolete but are still supported.

GBLoadWave Open File Dialog

If you include the `/I` flag, or if the `/P=pathName` and `fileNameStr` parameters do not fully specify the file to be loaded, GBLoadWave displays the Open File dialog.