

WhichListItem

WhichListItem(*itemStr*, *listStr* [, *listSepStr* [, *startIndex* [, *matchCase*]]])

The WhichListItem function returns the index of the first item of *listStr* that matches *itemStr*. *listStr* should contain items separated by *listSepStr* which typically is ";". If the item is not found in the list, -1 is returned.

Use WhichListItem to locate an item in a string containing a list of items separated by a string (usually a single ASCII character), such as those returned by functions like **TraceNameList** or **AnnotationList**, or a line from a delimited text file.

listSepStr, *startIndex*, and *matchCase* are optional; their defaults are ";", 0, and 1 respectively.

Details

WhichListItem differs from **FindListItem** in that WhichListItem returns a list index, while FindListItem returns a character offset into a string.

listStr is searched for *itemStr* bound by *listSepStr* on the left and right.

listStr is treated as if it ends with a *listSepStr* even if it doesn't.

Searches for *listSepStr* are always case-sensitive. The comparison of *itemStr* to the contents of *listStr* is usually case-sensitive. Setting the optional *matchCase* parameter to 0 makes the comparison case insensitive.

If *itemStr* is not found, if *listStr* is "", or if *startIndex* is not within the range of 0 to **ItemsInList**(*listStr*)-1, then -1 is returned.

In Igor6, only the first byte of *listSepStr* was used. In Igor7 and later, all bytes are used.

Items can be empty. In "abc;def;;ghi", the third item, whose zero-based index is 2, is empty. In ";def;;ghi;" the first and third items, whose zero-based indices are 0 and 2, are empty.

If *startIndex* is specified, then *listSepStr* must also be specified. If *matchCase* is specified, *startIndex* and *listSepStr* must be specified.

Examples

```
Print WhichListItem("wave0", "wave0;wave1;")           // prints 0
Print WhichListItem("c", "a;b;")                       // prints -1
Print WhichListItem("", "a;b;")                       // prints 1
Print WhichListItem("c", "a,b,c,x,c", ",", "")        // prints 2
Print WhichListItem("c", "a,b,c,x,c", ",", "", 3)     // prints 4
Print WhichListItem("C", "x;c;C;")                     // prints 2
Print WhichListItem("C", "x;c;C;", ";", 0, 0)         // prints 1
```

See Also

The **AddListItem**, **FindListItem**, **FunctionList**, **ItemsInList**, **RemoveListItem**, **RemoveFromList**, **StringFromList**, **StringList**, **TraceNameList**, **VariableList**, and **WaveList** functions.

WignerTransform

WignerTransform [/Z] [/WIDE=*wSize*] [/GAUS=*gaussianWidth*] [/DEST=*destWave*] *srcWave*

The WignerTransform operation computes the Wigner transformation of a 1D signal in *srcWave*, which is the name of a real or complex wave. The result of the WignerTransform is stored in *destWave* or in the wave M_Wigner in the current data folder.

Flags

- | | |
|------------------------|---|
| /DEST= <i>destWave</i> | Creates by default a real wave reference for the destination wave in a user function. See Automatic Creation of WAVE References on page IV-72 for details. |
| /GAUS= <i>gWidth</i> | Computes the Gaussian Wigner Transform, which is a convolution of the Wigner Transform with a two-dimensional Gaussian (in the two parameters of the transform). The computation of the transform simplifies significantly when the product of the widths of the two Gaussians is unity (minimum uncertainty ellipse). <i>gWidth</i> uses the same units as the <i>srcWave</i> scaling. |

WignerTransform

<code>/OUT=type</code>	Sets the output data type of the standard (not Gaussian) Wigner transform. The following data types are supported: 1: Complex 2: Real (default) 3: Magnitude 4: Squared magnitude <code>/OUT</code> is not allowed with the Gaussian Wigner transform (<code>/GAUS</code>) in which the output is always real. The <code>/OUT</code> flag was added in Igor Pro 8.00.
<code>/WIDE=wSize</code>	Computes Wigner Transform and sets the transform width to <i>wSize</i> . This is the default transformation with <i>wSize</i> set to the size of <i>srcWave</i> .
<code>/Z</code>	No error reporting.

Details

The Wigner transform maps a time signal $U(t)$ into a 2D time-frequency representation:

$$W(t, \nu) = \int_{-\infty}^{\infty} U\left(t + \frac{x}{2}\right) U^*\left(t - \frac{x}{2}\right) e^{-2\pi i x \nu} dx.$$

The computation of the Wigner transform evaluates the offset product

$$U\left(t + \frac{x}{2}\right) U^*\left(t - \frac{x}{2}\right)$$

over a finite window and then Fourier transforms the result. The offset product can be evaluated over a finite window width, which can vary from a few elements of the input wave to the full length of the wave. You can control the width of this window using the `/WIDE` flag. If you do not specify the output destination, WignerTransform saves the results in the wave `M_Wigner` in the current data folder.

Although the Wigner transform is real, the output will be complex when *srcWave* is complex. By inspecting the complex wave you can gain some insight into the numerical stability of the algorithm. The X-scaling of the output wave is identical to the scaling of *srcWave*. The Y-scaling of the input wave is taken from the Fourier Transform of the offset product, which in turn is determined by the X-scaling of *srcWave*.

Specifically, if `dx=DimDelta(srcWave, 0)` and *srcWave* has N points then `dy=DimDelta(M_Wigner, 1)=1/(dx*N)`. WignerTransform does not set the units of the output wave.

The Ambiguity Function is related to the Wigner Transform by a Fourier Transform, and is defined by

$$A(\tau, \nu) = \int_{-\infty}^{\infty} U\left(t + \frac{\tau}{2}\right) U^*\left(t - \frac{\tau}{2}\right) e^{-2\pi i t \nu} dt.$$

Convolving the Wigner Transform with a 2D Gaussian leads to what is sometimes called the Gaussian Wigner Transform or GWT. Formally the GWT is given by the equation:

$$GWT(t, \nu; \delta_t, \delta_\nu) = \frac{1}{\delta_t \delta_\nu} \iint dt' d\nu' W(t', \nu') \exp \left\{ -2\pi \left[\left(\frac{t-t'}{\delta_t} \right)^2 + \left(\frac{\nu-\nu'}{\delta_\nu} \right)^2 \right] \right\}.$$

Computationally this equation simplifies if the respective widths of the two Gaussians satisfy the minimum uncertainty condition $\delta_t^* \delta_\nu = 1$. The `/GAUS` flag calculates the Gaussian Wigner Transform using your specified width, δ_t , and it selects a δ_ν such that it satisfies the minimum uncertainty condition.