Convolution of a Gaussian peak with an exponential decay

```
Function convfunc(pw, yw, xw) : FitFunc
	WAVE pw, yw, xw

	// pw[0] = gaussian baseline offset
	// pw[1] = gaussian amplitude
	// pw[2] = gaussian position
	// pw[3] = gaussian width
	// pw[4] = exponential decay constant of instrument response

	// make a wave to contain an exponential with decay
	// constant pw[4]. The wave needs enough points to allow
	// any reasonable decay constant to get really close to zero.
	// The scaling is made symmetric about zero to avoid an X
	// offset from Convolve/A
	Variable dT = deltax(yw)
	Make/D/O/N=201 expwave
	// long enough to allow decay to zero
	setscale/P x -dT*100,dT,expwave

	// fill expwave with exponential decay
	expwave = (x>=0)*pw[4]*dT*exp(-pw[4]*x)

	// Normalize exponential so that convolution doesn't change
	// the amplitude of the result
	Variable sumexp
	sumexp = sum(expwave, -inf,inf)
	expwave /= sumexp

	// Put a Gaussian peak into the output wave
	yw = pw[0]+pw[1]*exp(-((x-pw[2])/pw[3])^2)
	// and convolve with the exponential; NOTE /A
	convolve/A expwave, yw
End
```

Some things to be aware of with regard to this function:

1. A wave is created inside the function to store the exponential decay. Making a wave can be a time-consuming operation; it is less convenient for the user of the function, but can save computation time if you make a suitable wave ahead of time and then simply reference it inside the function.

2. The wave containing the exponential decay ispassed to the convolve operation. The use of the /A flag prevents the convolve operation from

changing the length of the output wave yw. You may wish to read the section on Convolution.

3. The output wave yw is used as a parameter to the convolve operation. Since the convolve operation assumes that the data are evenly-space, this use of yw means that the function does not satisfy points 2) or 3) above. If you use input data to the fit that has missing points or unevenly-spaced X values, this function will fail.

You may also find the section Waveform Arithmetic and Assignment helpful. Here is a (much) more complicated version of the fitting function that solves these problems, and also is more robust in terms of accuracy. The comments in the code explain how the function works.

Note that this function makes at least two different waves using the Make operation, and that we have used Make/D to make the waves double-precision. This can be crucial.

```
Function convfunc(pw, yw, xw) : FitFunc
	WAVE pw, yw, xw

	// pw[0] = gaussian baseline offset
	// pw[1] = gaussian amplitude
	// pw[2] = gaussian position
	// pw[3] = gaussian width
	// pw[4] = exponential decay constant of instrument
	//			response (this parameter is actually
	//		the inverse of the time constant, in
	//			order to be just like Igor's built-in exp fit
	//			function, which was written in the days when a
	//			floating-point divide took much longer than a
	//			multiply).

	// make a wave to contain an exponential with decay
	// constant pw[4]. The wave needs enough points to allow
	// any reasonable decay constant to get really close to zero.
	// The scaling is made symmetric about zero to avoid an X
	// offset from Convolve/A

	// resolutionFactor sets the degree to which the exponential
	// will be over-sampled with regard to the problems parameters.
	// Increasing this number increases the number of time
	// constants included in the calculation. It also decreases
	// the point spacing relative to the problem's time constants.
	// Increasing will also increase the time required to compute

	Variable resolutionFactor = 10
```

```
// dt contains information on important time constants.
// We wish to set the point spacing for model calculations
// much smaller than exponential time constant or gaussian width.

Variable dT = min(1/(resolutionFactor*pw[4]), pw[3]/resolutionFactor)

// Calculate suitable number points for the exponential. Length
// of exponential wave is 10 time constants; doubled so
// exponential can start in the middle; +1 to make it odd so
// exponential starts at t=0, and t=0 is exactly the middle
// point. That is better for the convolution.

Variable nExpWavePnts = round(resolutionFactor/(pw[4]*dT))*2 + 1

Make/D/O/N=(nExpWavePnts) expwave


// In this version of the function, we make a y output wave
// ourselves, so that we can control the resolution and
// accuracy of the calculation. It also will allow us to use
// a wave assignment later to solve the problem of variable
// X spacing or missing points.

Variable nYPnts = max(resolutionFactor*numpnts(yw), nExpWavePnts)
Make/D/O/N=(nYPnts) yWave


// This wave scaling is set such that the exponential will
// start at the middle of the wave

setscale/P x -dT*(nExpWavePnts/2),dT,expwave

// Set the wave scaling of the intermediate output wave to have
// the resolution calculated above, and to start at the first
// X value.
setscale/P x xw[0],dT, yWave

// fill expwave with exponential decay
expwave = (x>=0)*pw[4]*dT*exp(-pw[4]*x)

// Normalize exponential so that convolution doesn't change
// the amplitude of the result
Variable sumexp
sumexp = sum(expwave, -inf,inf)
expwave /= sumexp
```

```
	// Put a Gaussian peak into the intermediate output wave. We use
	// our own wave (yWave) because the convolution requires a wave
	// with even spacing in X, whereas we may get X values input that
	// are not evenly spaced.
	yWave = pw[0]+pw[1]*exp(-((x-pw[2])/pw[3])^2)

	// and convolve with the exponential; NOTE /A
	convolve/A expwave, yWave

	// move appropriate values corresponding to input X data into
	// the output Y wave. We use a wave assignment involving the
	// input X wave. This will extract the appropriate values from
	// the intermediate wave, interpolating values where the
	// intermediate wave doesn't have a value precisely at an X
	// value that is required by the input X wave. This wave
	// assignment also solves the problem with auto-destination:
	// The function can be called with an X wave that sets any X
	// spacing, so it doesn't matter what X values are required.
	yw = yWave(xw[p])
End
```

Note that none of the computations involve the wave yw. That was done so that the computations could be done at finer resolution than the resolution of the data to be fit. By making a separate wave, it is not necessary to modify yw.
The actual return values are picked out of yWave using the special X-scale-based indexing available for one-dimensional waves in a wave assignment.