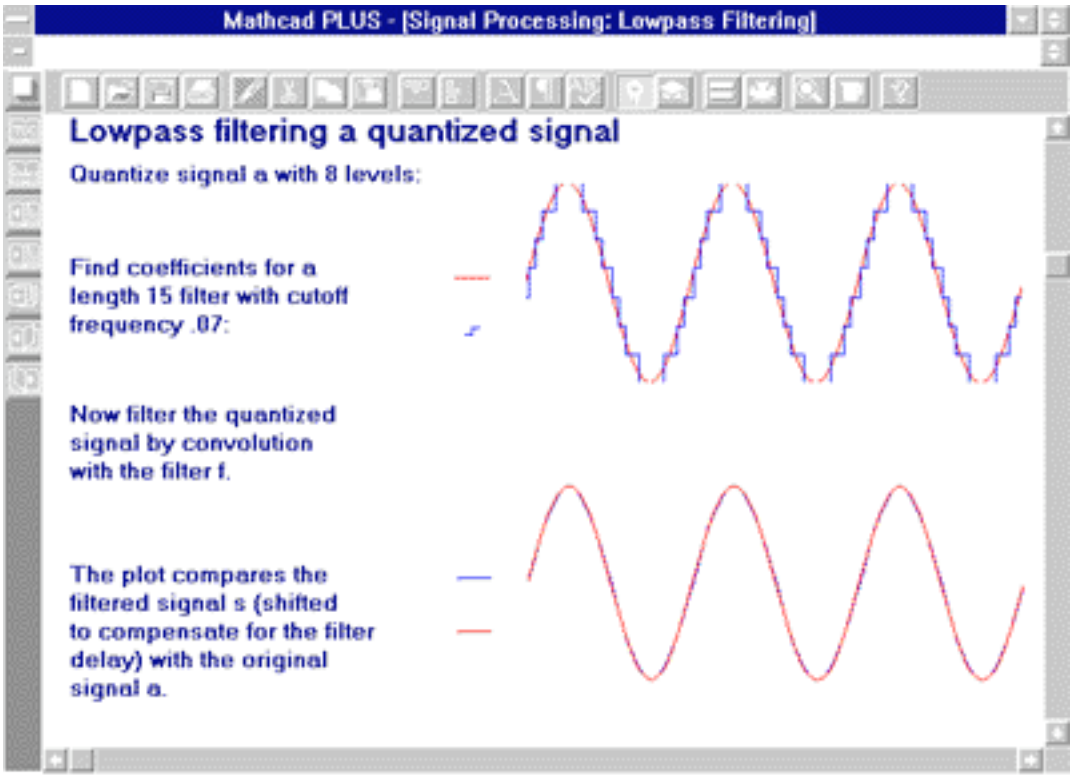
Mathcad Signal Processing Function Pack

Platform: Windows
Requires Mathcad PLUS 5.0 or higher, 1 MB hard disk space
Available for immediate download (size 651805 bytes) or ground shipment

This Function Pack comes with more than 60 signal processing functions for Mathcad that add demonstration and implementation capabilities. Perform many variations of signal analysis including, acoustic or optical and analogy or digital. For example, generate coefficients for a FIR lowpass filter, convolve the impulse response with a test signal, and plot the input and output signals for comparison. Because these new functions become part of Mathcad's built-in functionality, you can immediately see results and explore the effects of changing parameters in your routines.

Table of Contents | Product Sample | Back to Product List



Alter the filter length, the cutoff frequency or the window used to compare various input and output signals and let Mathcad calculate the results instantly.

The Function Pack also comes with a built-in Electronic Book that documents the functions and illustrates their use. Topics include: Function Generators, Transform Analysis, Spectral Analysis, Time Series Analysis, Digital Filtering, Filter Design, Windowing, Correlation, Convolution, and much more.

Mathcad Signal Processing Function Pack

TABLE OF CONTENTS (page 1 of 1)

Mathcad Signal Processing Function Pack

SAMPLE PAGE (page 1 of 9)

Filter Gain and Response

The **gain(C,f)** and response**(x,C,n)** functions are used to compute the complex gain and the response of a digital FIR or IIR filter.

The two arguments to **gain** are a coefficient array **C** and a frequency **f**. If **C** has one column, it is treated as the coefficients for an FIR filter. If there is an even number of columns, the function assumes that each pair of columns gives the coefficients for a section of an IIR filter in cascade form. The first column in each pair should contain the coefficients of the numerator of the section's transfer function and the second column the denominator coefficients, with the transfer function normalized so that the constant term in the denominator is 1.

The frequency argument gives the frequency at which the complex gain is to be found as a fraction of the sampling frequency. This frequency must be between **0** and **1** and will generally be between **0** and **.5**. Here are a few examples, using coefficients generated by the **iirlow** and **lowpass** functions.

The array **A** gives the coefficients of a second order lowpass Butterworth filter with cutoff frequency .2:

$$A := \text{iirlow}(\text{butter}(2), .2)$$

The filter coefficients are

$$A = \begin{pmatrix} 0.207 & 1 \\ 0.413 & -0.37 \\ 0.207 & 0.196 \end{pmatrix}$$

. . . so the transfer function is

$$\frac{.207 + .413 \cdot z^{-1} + .207 \cdot z^{-2}}{1 - .37 \cdot z^{-1} + .196 \cdot z^{-2}}$$

Table of Contents

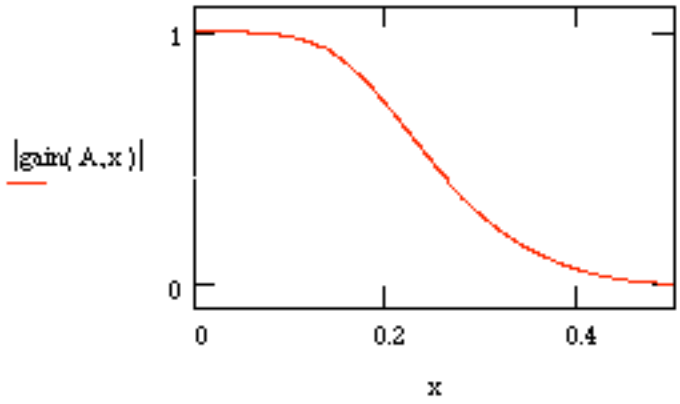
Back to Product List

Mathcad Signal Processing Function Pack

SAMPLE PAGE (page 2 of 9)

We plot the magnitude of the gain over frequencies ranging from 0 to half the sampling frequency:

$x := 0, .001 .. .5$



If you prefer to scale the frequency so that the sampling frequency is represented as 2 pi, you can define your range variable accordingly and then divide the frequency argument to the gain function by 2 pi :

$\omega := 0, .005 .. \pi$

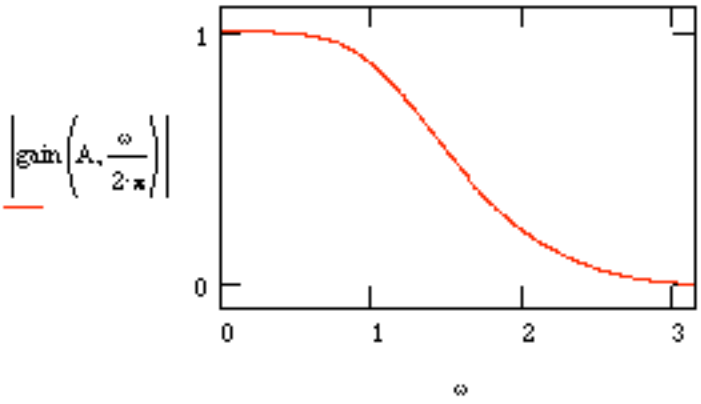


Table of Contents

Back to Product List

Mathcad Signal Processing Function Pack

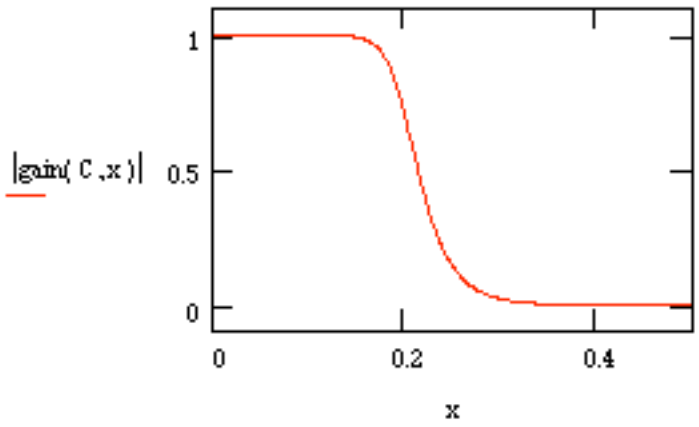
SAMPLE PAGE (page 3 of 9)

The next example is a sixth-order lowpass filter with three sections corresponding to three quadratic factors of the numerator and denominator of the transfer function:

$$C := \text{iirlow}(\text{butter}(6), .2)$$

$$C = \begin{pmatrix} 0.277 & 1 & 0.207 & 1 & 0.18 & 1 \\ 0.554 & -0.496 & 0.413 & -0.37 & 0.36 & -0.322 \\ 0.277 & 0.605 & 0.207 & 0.196 & 0.18 & 0.042 \end{pmatrix}$$

For this sixth-order filter the response falls off much more rapidly than for the second-order filter defined by the array **A**. Here's a plot of the gain:



Finally we compute the gain for an FIR filter designed using the function **bandpass**.

Choose a Blackman window:

$$\text{setwindow}(6) = 6$$

Compute coefficients for a length 51 bandpass filter with passband between **.2** and **.4**:

$$F := \text{bandpass}(.2, .4, 51)$$

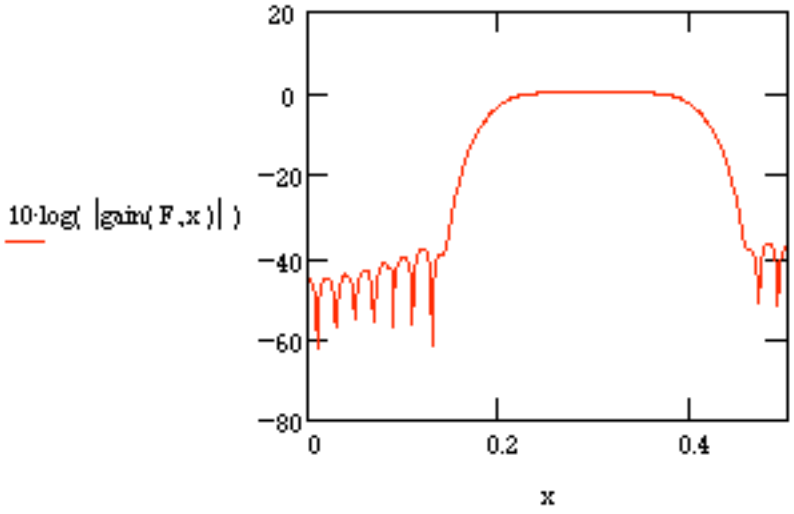
Table of Contents

Back to Product List

Mathcad Signal Processing Function Pack

SAMPLE PAGE (page 4 of 9)

Plot the gain in dB:



You can compute gains in **Mathcad** directly from the definition of the transfer function, using the summation operator. For example, the gain of the filter **F** at a frequency **x** is given by

$$n := 0 .. 50$$

$$g(x) := \sum_{n} \exp(-2 \cdot \pi \cdot j \cdot n \cdot x) \cdot F_n$$

This is simply the transfer function evaluated at **z = 2 pi jx**. Here are a few values compared with the corresponding values returned by the **gain** function:

g(.25) = −0.999j	gain(F, .25) = −0.999j
g(.2) = 0.5	gain(F, .2) = 0.5

Computations done with **gain** are much faster those done with the summation operator, so **gain** is particularly convenient when you want to compute the gain at a large number of frequencies, for example when you're plotting the frequency response. However, note that for very long filters even the computation with **gain** will take a while, so you'll want to choose a fairly coarse grid (say .01) for plotting.

Table of Contents

Back to Product List

Mathcad Signal Processing Function Pack

SAMPLE PAGE (page 5 of 9)

The phase or argument of the complex gain represents the phase shift of the filter. As an example, we'll generate coefficients for a lowpass FIR filter of length 37 using a Hanning window.

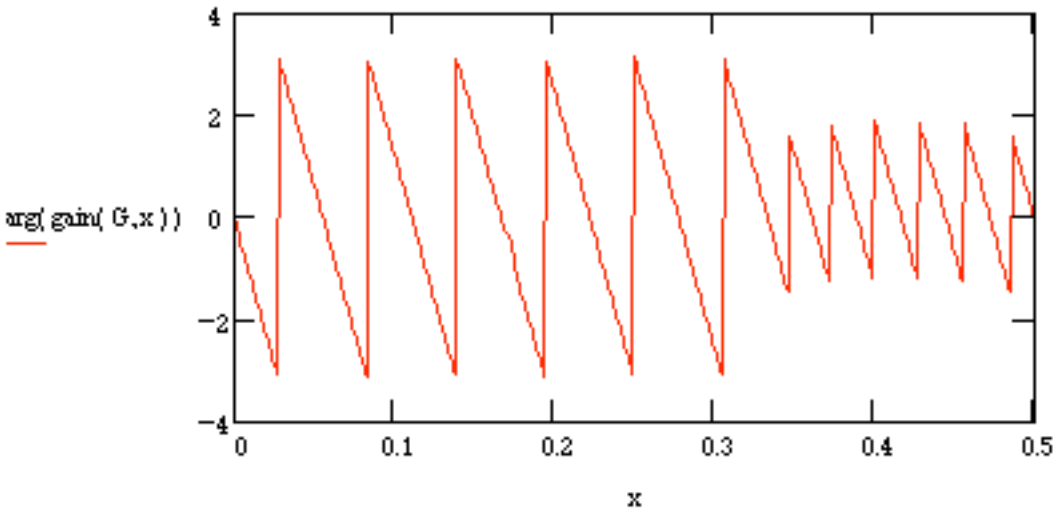
Set the window to Hanning:

setwindow(4) = 4

Compute coefficients:

G := lowpass(.3, 37)

Plot the phase change using the **arg** function:



The phase is linear in the passband, as we'd expect, but we're seeing the effect of the filter delay. We can compute the complex gain relative to the delayed signal by dividing by the transfer function of the delay **D**, which is

$$z^{-D}$$

where

$$z = \exp(j \cdot \omega) = \exp(j \cdot 2 \cdot \pi \cdot x)$$

Table of Contents

Back to Product List

Mathcad Signal Processing Function Pack

SAMPLE PAGE (page 6 of 9)

For this length 37 filter the delay **D** is 18, so the relative phase shift is

$$\frac{\text{gain}(G, x)}{\exp(-j \cdot 2 \cdot \pi \cdot x \cdot 18)}$$

When the gain is real and negative, roundoff errors in the gain computation can leave a tiny positive or negative imaginary part, so the phase may jump arbitrarily between **pi** and **- pi**. To make sure these values get handled uniformly, redefine the **arg** as follows:

$$\text{arg}(z) := \arg\left(z + j \cdot 10^{-15}\right)$$

Now we'll plot the magnitude of the gain in decibels and the phase shift relative to the delayed signal.

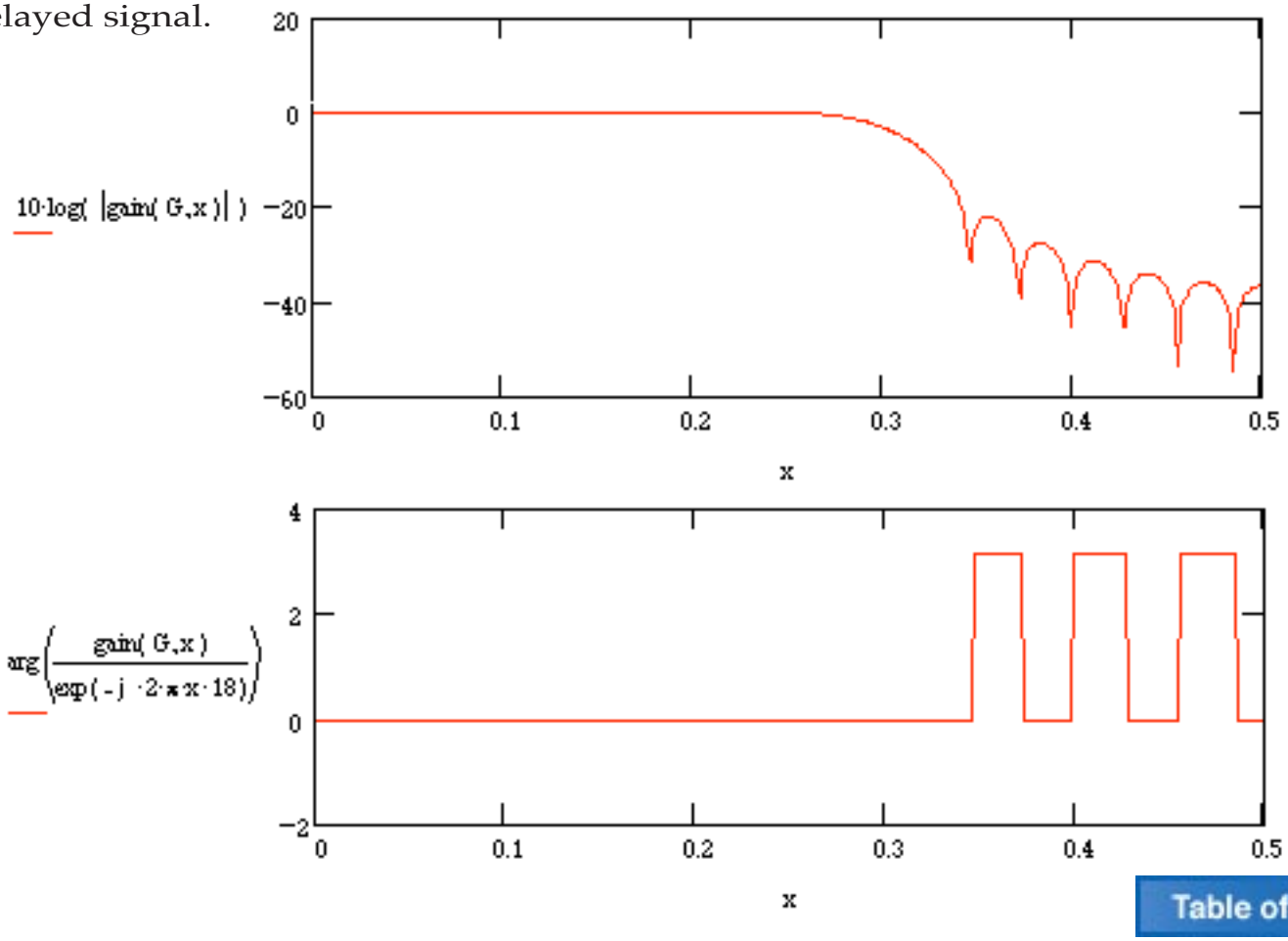


Table of Contents

Back to Product List

Mathcad Signal Processing Function Pack

SAMPLE PAGE (page 7 of 9)

The function **response(x,C,n)** computes the first **n** elements of the output of the FIR or IIR filter with coefficient array **C** for the real input vector **x**. The conditions on **C** are the same as for the gain function; **n**, the length of the output vector, should be no greater than the length of the input.

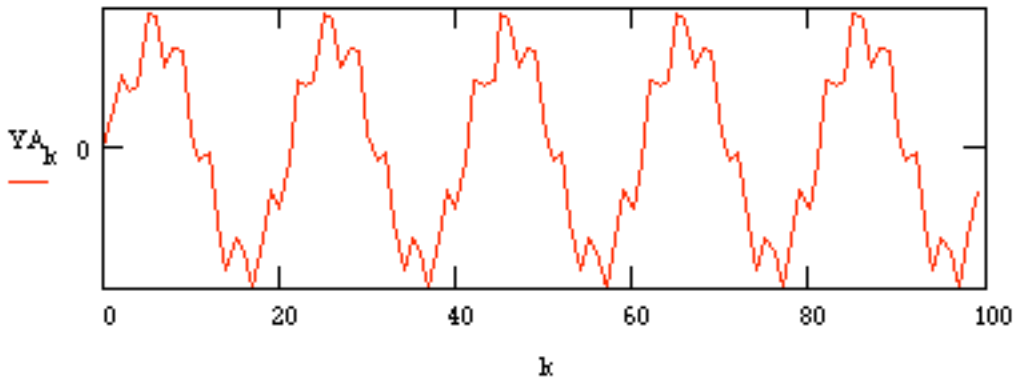
We'll define an input signal with a low frequency component at .05 of the sampling frequency and another component at .3.

$$i := 0 .. 999 \qquad X_i := \sin\left(2\cdot\pi\cdot\frac{i}{1000}\cdot 50\right) + \sin\left(2\cdot\pi\cdot\frac{i}{1000}\cdot 300\right)$$

The first 100 steps of the response from the first filter are given by

$$YA := response(X, A, 100)$$

$$k := 0 .. 99$$



For the order 6 filter **C** the **response** function computes the response by feeding the output of the first section into the second section, and this output into the third:

$$YC := response(X, C, 100)$$

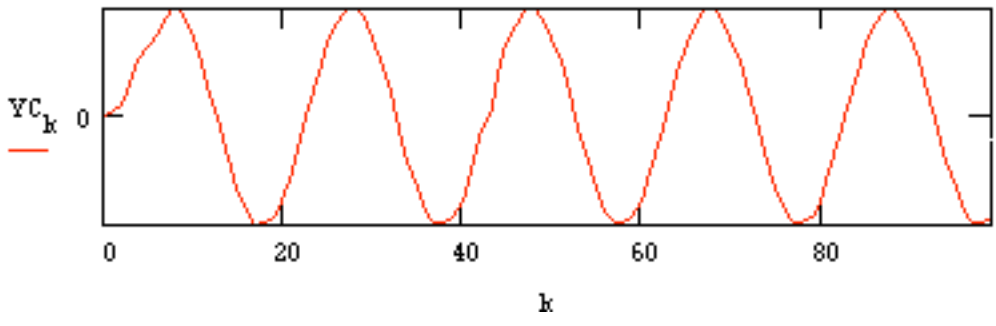


Table of Contents

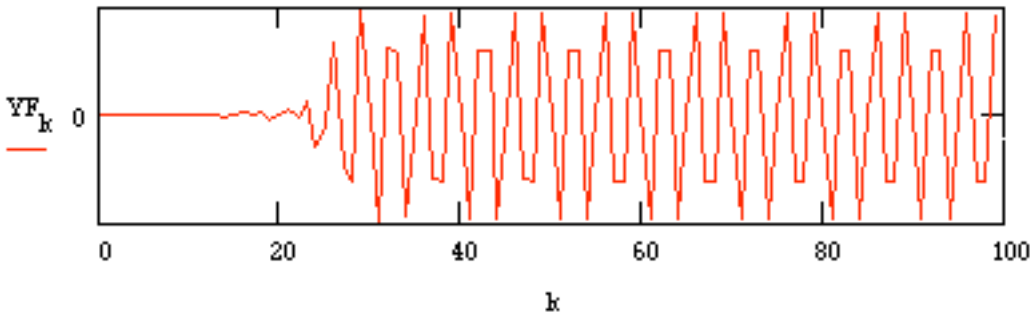
Back to Product List

Mathcad Signal Processing Function Pack

SAMPLE PAGE (page 8 of 9)

The bandpass filter **F** transmits the higher of the two signal frequencies:

$YF := response(X, F, 100)$



As a final example, we'll verify that the lowpass filter **G** defined above has 0 phase shift for several frequencies in the passband by plotting the filter output together with the input delayed by the filter delay of 18.

$i := 0 .. 1000$

$x1_i := \sin\left(2 \cdot \pi \cdot \frac{i}{10}\right)$

$x2_i := \sin\left(2 \cdot \pi \cdot \frac{i}{27}\right)$

$y1 := response(x1, G, 150)$

$y2 := response(x2, G, 150)$

$k := 0 .. 100$

$x1d_{k+18} := x1_k$

$x2d_{k+18} := x2_k$

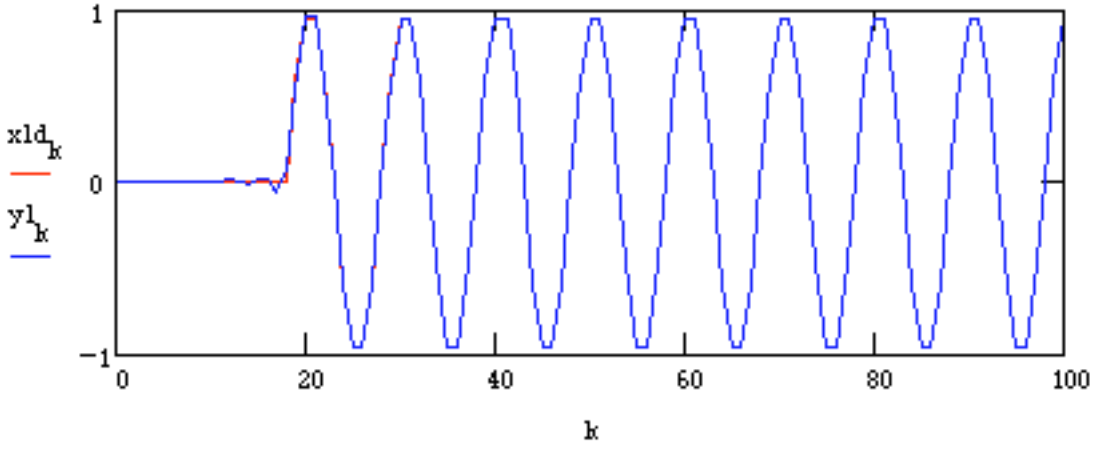
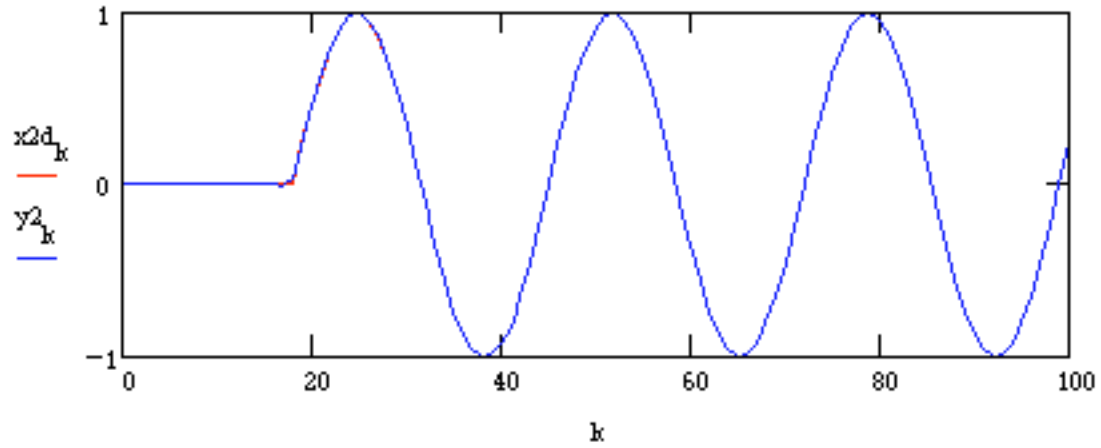


Table of Contents

Back to Product List

Mathcad Signal Processing Function Pack

SAMPLE PAGE (page 9 of 9)



Using response for convolution

Note that if the filter array has one column, **response** is simply carrying out a convolution. Thus this function provides an alternative to the FFT-based function **convol**; **response** is much faster if you only want to look at a small initial portion of the convolution of two long sequences. The example on the next screen compares the two calculations.

$j := 0 .. 200$ $\quad x_j := rnd(1)$ $\quad y_j := rnd(1)$

$Y1 := convol(x, y)$ $\quad Y2 := response(x, y, 21)$ $\quad k := 0 .. 7$

$Y1_k$	$Y2_k$
$6.825 \cdot 10^{-5}$	$6.825 \cdot 10^{-5}$
0.011	0.011
0.133	0.133
0.345	0.345
0.456	0.456
1.025	1.025
0.786	0.786
1.745	1.745

Table of Contents

Back to Product List