

Digital Filter Design MathScript Functions

Use the Digital Filter Design MathScript functions to design digital filters using a [text-based language](#). The following is a list of Digital Filter Design MathScript classes of functions and commands that LabVIEW MathScript supports.

The LabVIEW Full and Professional Development Systems install [additional MathScript functions](#).

The LabVIEW Control Design and Simulation Module installs [additional MathScript functions](#).

Class	Description
multirate	Multirate filter design functions
singlerate	Single-rate filter design functions

multirate (Digital Filter Design Toolkit, MathScript Class)

Use members of the multirate class to design multirate filters, such as [Nyquist and halfband filters](#).

Function	Description
firhalfband	Halfband FIR filter
firnyquist	Lowpass FIR Nyquist filter

firhalfband (Digital Filter Design Toolkit, MathScript Function)

Owning Class: [multirate](#)

Syntax

`b = firhalfband(n, f)`

`b = firhalfband(n, f, 'high')`

`b = firhalfband('minorder', f, ripple)`

`b = firhalfband('minorder', f, ripple, 'kaiser')`

`b = firhalfband('minorder', f, ripple, 'kaiser', 'high')`

Description

Designs a [halfband, finite impulse response \(FIR\) filter](#).

[Examples](#)

Inputs

Name	Description
n	Specifies the order of the filter. n is an even, positive number. If you do not specify n , you must specify a valid value for ' minorder '.
f	Specifies the passband edge frequency. f is a double-precision, floating-point number that must fall in the range (0, 0.5).
' high '	Specifies that b returns a highpass, halfband FIR filter. If you do not specify ' high ', b returns a lowpass, halfband FIR filter.
' minorder '	Specifies that b returns a filter with the minimum order that meets the design requirements. If you do not specify ' minorder ', you must specify a valid value for n .
ripple	Specifies the maximum ripple in the passband and stopband. ripple is a double-precision, floating-point number that must fall in the range (0, 1).
' kaiser '	Specifies whether to use the Kaiser Window method to design the filter. If you do not specify ' kaiser ', this function uses the Remez method to design the filter.

Outputs

Name	Description
b	Returns the coefficients of the designed FIR filter. b is a real vector with a length of $n+1$ or ' minorder

Examples

```
b = firhalfband(20, 0.4);figure;  
freqz(b);
```

```
b = firhalfband('minorder', 0.4, 0.001);  
figure;  
freqz(b);
```

Related Topics

[fircband](#)

[firnyquist](#)

firnyquist (Digital Filter Design Toolkit, MathScript Function)

Owning Class: [multirate](#)

Syntax

`b = firnyquist(n, l, rolloff)`

`b = firnyquist(n, l, rolloff, 'nonnegative')`

`b = firnyquist('minorder', l, rolloff, ripple)`

Description

Designs a lowpass, [finite impulse response \(FIR\)](#), [Nyquist filter](#).

[Examples](#)

Inputs

Name	Description
n	Specifies the order of the filter. This input also specifies to use the Remez method to design the filter. n is an even, positive number. If you do not specify n , you must specify a valid value for ' minorder '.
l	Specifies the sampling frequency conversion factor of the multirate filter. l is an integer greater than one.
rolloff	Specifies the roll off factor. This factor determines the relative transition bandwidth, which equals $(\text{transition band}) / (2 * \text{passband} + \text{transition band})$. rolloff must fall in the range (0, 1). A smaller value of rolloff returns a narrower transition bandwidth if the value of l does not change.
' nonnegative '	Specifies that b returns an FIR filter with a nonnegative zero-phase response. If you do not specify ' nonnegative ', b might return an FIR filter with negative values in the zero-phase response.
' minorder '	Specifies that b returns a filter with the minimum order value that meets the design requirements. This input also specifies to use the Kaiser Window method to design the FIR filter. If you do not specify ' minorder ', you must specify a valid value for n .
ripple	Specifies the maximum ripple in the passband and stopband. ripple is a double-precision, floating-point number that must fall in the range (0, 1).

Outputs

Name	Description
b	Returns the coefficients of the designed FIR filter. b is a real vector with a length of $n+1$ or ' minorder

Examples

```
b = firnyquist(20, 4, 0.1);figure;  
freqz(b);
```

```
b = firnyquist(40, 5, 0.1, 'nonnegative');  
fft_mag = abs(fft(b, 16384));  
figure;  
plot(0:1/8192:1, fft_mag(1:8193));
```

```
b = firnyquist('minorder', 5, 0.1, 0.001);  
figure;  
freqz(b);
```

Related Topics

[fircband](#)

[firhalfband](#)

singlerate (Digital Filter Design Toolkit, MathScript Class)

Use members of the singlerate class to design [finite impulse response \(FIR\)](#) or [infinite impulse response \(IIR\)](#) filters.

Function	Description
fircband	FIR filter with constraints in frequency bands
firgr	FIR filter using minimax
firlpnorm	FIR filter using least p -th norm
firminphase	Minimum phase spectral factor of FIR filter
iircomb	IIR comb filter
iirgrpdelay	Allpass IIR filter approximating group delay
iirlpnorm	IIR filter using least p -th norm
iirlpnormc	IIR filter with constraints using least p -th norm
iirnotch	Second-order IIR notch filter
iirpeak	Second-order IIR peak filter

firband (Digital Filter Design Toolkit, MathScript Function)

Owning Class: [singlerate](#)

Syntax

`b = fircband(n, f, mag, w, c)`

`b = fircband(n, f, mag, w, c, t)`

`b = fircband(n, f, mag, ftype)`

`b = fircband(n, f, mag, ftype, t)`

Description

Designs a [finite impulse response \(FIR\) filter](#) with constraints in the frequency bands by using the minimax principle. The FIR filter minimizes the maximum error between the target frequency response and the designed filter frequency response. You can use this function to design the following FIR filters: [types I-IV linear phase](#), [minimum and maximum phase](#), [ripple constraint](#), [single-point band](#), [exact gain control](#), and [arbitrary shape](#).

[Examples](#)

Inputs

Name	Description								
n	Specifies the order of the filter. n is a nonnegative integer.								
f	Specifies the frequency points. f is a vector whose values increase monotonically between 0 and 1.								
mag	Specifies the magnitude response of the filter at f . mag is a vector with the same length as f .								
w	Specifies the weight of each frequency band. w is a vector whose length must equal <code>length(f)/2</code> .								
c	Specifies whether each element of w is a weight or a ripple constraint of a frequency band. c is a string of the same length as w that accepts a combination of the following values: <table border="1" data-bbox="316 871 1425 1081"> <tbody> <tr> <td>'w'</td> <td>Denotes that the corresponding element in w is a weight of the frequency band.</td> </tr> <tr> <td>'c'</td> <td>Denotes that the corresponding element in w is a ripple constraint of the frequency band.</td> </tr> </tbody> </table>	'w'	Denotes that the corresponding element in w is a weight of the frequency band.	'c'	Denotes that the corresponding element in w is a ripple constraint of the frequency band.				
'w'	Denotes that the corresponding element in w is a weight of the frequency band.								
'c'	Denotes that the corresponding element in w is a ripple constraint of the frequency band.								
t	Specifies the type of filter you want to design. t is a string that accepts the following values: <table border="1" data-bbox="316 1218 1123 1453"> <tbody> <tr> <td>'sym' (default)</td> <td>Designs a symmetric filter.</td> </tr> <tr> <td>'antisym'</td> <td>Designs an antisymmetric filter.</td> </tr> <tr> <td>'minphase'</td> <td>Designs a minimum phase filter.</td> </tr> <tr> <td>'maxphase'</td> <td>Designs a maximum phase filter.</td> </tr> </tbody> </table>	'sym' (default)	Designs a symmetric filter.	'antisym'	Designs an antisymmetric filter.	'minphase'	Designs a minimum phase filter.	'maxphase'	Designs a maximum phase filter.
'sym' (default)	Designs a symmetric filter.								
'antisym'	Designs an antisymmetric filter.								
'minphase'	Designs a minimum phase filter.								
'maxphase'	Designs a maximum phase filter.								
ftype	Specifies the type of each frequency point in f . ftype is a string of the same length as f that accepts a combination of the following values: <table border="1" data-bbox="316 1638 1153 1816"> <tbody> <tr> <td>'n'</td> <td>Denotes a regular frequency point.</td> </tr> <tr> <td>'s'</td> <td>Denotes a single-point band.</td> </tr> <tr> <td>'e'</td> <td>Denotes a frequency point with an exact gain.</td> </tr> </tbody> </table>	'n'	Denotes a regular frequency point.	's'	Denotes a single-point band.	'e'	Denotes a frequency point with an exact gain.		
'n'	Denotes a regular frequency point.								
's'	Denotes a single-point band.								
'e'	Denotes a frequency point with an exact gain.								

Outputs

Name	Description
b	Returns the coefficients of the designed FIR filter. b is a real vector with a length of $n+1$.

Examples

```
b = fircband(22, [0, 0.4, 0.5, 0.7, 0.8, 1], [1, 1, 0, 0, 1, 1], [0.05, 1, 0.1], 'cwc');  
fft_mag = abs(fft(b, 16384));  
figure;  
plot(0:1/8192:1, fft_mag(1:8193));
```

```
b = fircband(42, [0, 0.2, 0.25, 0.3, 0.5, 0.55, 0.6, 1], [1, 1, 0, 1, 1, 0, 1, 1],  
'nnsnnsnn');  
fft_mag = abs(fft(b, 16384));  
figure;  
plot(0:1/8192:1, fft_mag(1:8193));
```

```
b = fircband(82, [0, 0.0055, 0.03, 0.1, 0.15, 1], [0, 0, 0, 0, 1, 1], 'nnennn');  
fft_mag = abs(fft(b, 16384));  
figure;  
plot(0:1/8192:1, fft_mag(1:8193));
```

```
b = fircband(12, [0, 0.4, 0.5, 1], [1, 1, 0, 0], [1, 1], 'ww', 'minphase');  
fft_mag = abs(fft(b, 16384));  
figure;  
plot(0:1/8192:1, fft_mag(1:8193));  
figure;  
zplane(b, 1);
```

Related Topics

[firgr](#)

firgr (Digital Filter Design Toolkit, MathScript Function)

Owning Class: [singlerate](#)

Syntax

`b = firgr(m, f, mag, ripple)`

`b = firgr(m, f, mag, ripple, tm)`

`b = firgr(n, f, mag)`

`b = firgr(n, f, mag, w)`

`b = firgr(n, f, mag, w, ftype)`

`b = firgr(n, f, mag, w, ftype, t)`

Description

Designs a [finite impulse response \(FIR\) filter](#) using the minimax principle. The FIR filter minimizes the maximum error between the target frequency response and the designed filter frequency response. You can use this function to design the following FIR filters: [types I-IV linear phase](#), [minimum and maximum phase](#), [ripple constraint](#), [single-point band](#), [exact gain control](#), [arbitrary shape](#), [Hilbert transformers](#), and [differentiators](#). You can specify the filter order to use. This function also can calculate the minimum filter order, including the minimum odd and minimum even filter orders.

[Examples](#)

Inputs

Name	Description								
m	<p>Specifies the order of the filter. m is a string that accepts the following values:</p> <table border="1"> <tr> <td>'mineven'</td> <td>Specifies the minimum even order.</td> </tr> <tr> <td>'minodd'</td> <td>Specifies the minimum odd order.</td> </tr> <tr> <td>'minorder'</td> <td>Specifies the minimum order.</td> </tr> </table>	'mineven'	Specifies the minimum even order.	'minodd'	Specifies the minimum odd order.	'minorder'	Specifies the minimum order.		
'mineven'	Specifies the minimum even order.								
'minodd'	Specifies the minimum odd order.								
'minorder'	Specifies the minimum order.								
f	Specifies the frequency points. f is a vector whose values increase monotonically between 0 and 1.								
mag	Specifies the magnitude response of the filter at f . mag is a vector of the same length as f .								
ripple	Specifies the ripple of each frequency band. ripple is a vector whose length must equal $\text{length}(\mathbf{f})/2$. Every two consecutive elements in f make up one frequency band.								
tm	<p>Specifies the type of filter you want to design. If you specify a value for tm, you must set m to 'minorder'. If you do not specify a value for tm, you must set m to either 'mineven' or 'minodd'. tm is a string that accepts the following values:</p> <table border="1"> <tr> <td>'type I'</td> <td>Designs an even-order, symmetric filter.</td> </tr> <tr> <td>'type II'</td> <td>Designs an odd-order, symmetric filter.</td> </tr> <tr> <td>'type III'</td> <td>Designs an even-order, antisymmetric filter.</td> </tr> <tr> <td>'type IV'</td> <td>Designs an odd-order, antisymmetric filter.</td> </tr> </table>	'type I'	Designs an even-order, symmetric filter.	'type II'	Designs an odd-order, symmetric filter.	'type III'	Designs an even-order, antisymmetric filter.	'type IV'	Designs an odd-order, antisymmetric filter.
'type I'	Designs an even-order, symmetric filter.								
'type II'	Designs an odd-order, symmetric filter.								
'type III'	Designs an even-order, antisymmetric filter.								
'type IV'	Designs an odd-order, antisymmetric filter.								
n	Specifies the order of the filter. n is a nonnegative integer.								
w	Specifies the weight of each frequency point. w is a vector of the same length as f .								
ftype	<p>Specifies the type of each frequency point in f. ftype is a string of the same length as f that accepts a combination of the following values:</p> <table border="1"> <tr> <td>'n'</td> <td>Denotes a regular frequency point.</td> </tr> <tr> <td></td> <td></td> </tr> </table>	'n'	Denotes a regular frequency point.						
'n'	Denotes a regular frequency point.								

	's'	Denotes a single-point band.
	'e'	Denotes a frequency point with an exact gain.
t	Specifies the type of filter you want to design. t is a string that accepts the following values:	
	'hilbert'	Designs a Hilbert transformer.
	'differentiator'	Designs a differentiator.
	'minphase'	Designs a minimum phase filter.
	'maxphase'	Designs a maximum phase filter.

Outputs

Name	Description
b	Returns the coefficients of the designed FIR filter. b is a real vector with a length of n+1 or m+1 .

Examples

```
b = firgr('mineven', [0, 0.4, 0.5, 1], [1, 1, 0, 0], [0.1, 0.02]);  
fft_mag = abs(fft(b, 16384));  
figure;  
plot(0:1/8192:1, fft_mag(1:8193));
```

```
b = firgr(24, [0, 0.26, 0.3, 0.6, 0.64, 1], [0, 0, 1, 1, 0, 0], [1, 1, 1, 1, 2, 2],  
'nnnnnn', 'minphase');  
fft_mag = abs(fft(b, 16384));  
figure;  
plot(0:1/8192:1, fft_mag(1:8193));
```

```
b = firgr(20, [0.2, 0.8], [0.2, 0.8], [1, 1], 'nn', 'differentiator');  
fft_mag = abs(fft(b, 16384));  
figure;  
plot(0:1/8192:1, fft_mag(1:8193));
```

```
b = firgr('minorder', [0.04, 0.9], [1, 1], [0.1], 'type III');  
fft_mag = abs(fft(b, 16384));  
figure;  
plot(0:1/8192:1, fft_mag(1:8193));
```

Related Topics

[fircband](#)

firlnorm (Digital Filter Design Toolkit, MathScript Function)

Owning Class: [singlerate](#)

Syntax

`b = firlnorm(n, f, e, mag)`

`b = firlnorm(n, f, e, mag, w)`

`b = firlnorm(n, f, e, mag, w, p)`

`[b, err] = firlnorm(n, f, e, mag)`

`[b, err] = firlnorm(n, f, e, mag, w)`

`[b, err] = firlnorm(n, f, e, mag, w, p)`

Description

Designs a [finite impulse response \(FIR\) filter](#) that uses the [least p-th norm algorithm](#) to approximate the frequency response you specify.

[Examples](#)

Inputs

Name	Description
n	Specifies the order of the filter. n is a nonnegative integer.
f	Specifies the frequency points. f is a vector whose values increase monotonically between 0 and 1.
e	Specifies the band edge frequencies. e is a vector whose values must also exist in f .
mag	Specifies the magnitude response of the filter at f . mag is a vector of the same length as f .
w	Specifies the weight of each frequency point. w is a vector of the same length as f . The default is a vector in which each element has a value of 1.
p	Specifies the value of p to use in the least p -th norm algorithm. p is a positive integer that must fall in the range [1, 128]. The default is 128.

Outputs

Name	Description
b	Returns the coefficients of the designed FIR filter. b is a real vector with a length of $n+1$.
err	Returns the least p -th norm approximation error. err is a real number.

Examples

```
b = firlnorm(40, [0, 0.2, 0.5, 0.6, 1], [0, 0.5, 0.6, 1], [1, 2, 1, 0, 0]);fft_mag =  
abs(fft(b, 16384));  
figure;  
plot(0:1/8192:1, fft_mag(1:8193));
```

```
b = firlnorm(22, [0, 0.4, 0.6, 1], [0, 0.4, 0.6, 1], [1, 1, 0, 0], [1, 1, 1, 1], 2);  
figure;  
freqz(b);
```

```
b = firlnorm(22, [0, 0.4, 0.6, 1], [0, 0.4, 0.6, 1], [1, 1, 0, 0], [1, 1, 1, 1], 4);  
figure;  
freqz(b);
```

Related Topics

[iirlpnorm](#)

[iirlpnormc](#)

firminphase (Digital Filter Design Toolkit, MathScript Function)

Owning Class: [singlerate](#)

Syntax

```
b2 = firminphase(b1)
```

Description

Calculates the minimum phase spectral factor of a linear phase, [finite impulse response \(FIR\)](#) filter. The resulting spectral factor is also an FIR filter whose zeroes correspond to the zeroes of the original linear phase FIR filter inside or on the unit circle. If a zero is on the unit circle, the zero must be an even-multiplicity zero. In other words, the zero must occur an even number of times. The magnitude response of the spectral factor is the square root of that of the original FIR filter.

[Examples](#)

Inputs

Name	Description
b1	Specifies the coefficients of a linear phase filter with a nonnegative zero-phase response. b1 is a real vector.

Outputs

Name	Description
b2	Returns the minimum phase FIR spectral factor of b1 . b2 is a real vector.

Examples

```
bmin = fircband(12, [0, 0.4, 0.5, 1], [1, 1, 0, 0], [1, 1], 'ww', 'minphase');bmax =  
fliplr(bmin);  
b1 = conv(bmin, bmax);  
b2 = firminphase(b1);  
figure;  
zplane(b1, 1);  
figure;  
zplane(b2, 1);
```

iircomb (Digital Filter Design Toolkit, MathScript Function)

Owning Class: [singlerate](#)

Syntax

$[b, a] = \text{iircomb}(n, bw)$

$[b, a] = \text{iircomb}(n, bw, ab)$

$[b, a] = \text{iircomb}(n, bw, t)$

$[b, a] = \text{iircomb}(n, bw, ab, t)$

Description

Designs an [infinite impulse response \(IIR\) comb filter](#).

[Examples](#)

Inputs

Name	Description				
n	Specifies the order of the filter. n is a positive integer.				
bw	Specifies the bandwidth of the filter notch or peak at -ab dB. bw is a double-precision, floating-point number that must fall in the range $(0, 2/n)$.				
ab	Specifies the attenuation, in decibels, that corresponds to the bandwidth bw . ab is a double-precision, floating-point number greater than zero. The default is 3.0103, which corresponds to a 3 dB bandwidth, a commonly used bandwidth for a filter.				
t	Specifies the type of IIR comb filter you want to design. t is a string that accepts the following values: <table border="1" data-bbox="316 861 1063 976"><tbody><tr><td>'notch' (default)</td><td>Designs a comb notch filter.</td></tr><tr><td>'peak'</td><td>Designs a comb peak filter.</td></tr></tbody></table>	'notch' (default)	Designs a comb notch filter.	'peak'	Designs a comb peak filter.
'notch' (default)	Designs a comb notch filter.				
'peak'	Designs a comb peak filter.				

Outputs

Name	Description
b	Returns the numerator of the designed IIR filter. b is a real vector with a length of n+1 .
a	Returns the denominator of the designed IIR filter. a is a real vector with a length of n+1 .

Examples

```
[b, a] = iircomb(10, 0.001, 3.0103);  
fft_mag = abs(fft(b, 16384)./fft(a, 16384));  
figure;  
plot(0:1/8192:1, fft_mag(1:8193));
```

```
[b, a] = iircomb(10, 0.001, 3.0103, 'peak');  
fft_mag = abs(fft(b, 16384)./fft(a, 16384));  
figure;  
plot(0:1/8192:1, fft_mag(1:8193));
```

Related Topics

[iirnotch](#)

[iirpeak](#)

iirgrpdelay (Digital Filter Design Toolkit, MathScript Function)

Owning Class: [singlerate](#)

Syntax

`[b, a] = iirgrpdelay(n, f, e, grd)`

`[b, a] = iirgrpdelay(n, f, e, grd, w)`

`[b, a] = iirgrpdelay(n, f, e, grd, w, r)`

`[b, a, offset] = iirgrpdelay(n, f, e, grd)`

`[b, a, offset] = iirgrpdelay(n, f, e, grd, w)`

`[b, a, offset] = iirgrpdelay(n, f, e, grd, w, r)`

Description

Designs an allpass, [infinite impulse response \(IIR\) filter](#) that approximates the group delay you specify.

[Examples](#)

Inputs

Name	Description
n	Specifies the order of the filter. n is an even, positive integer.
f	Specifies the frequency points. f is a vector whose values increase monotonically between 0 and 1.
e	Specifies the band edge frequencies. e is a vector whose values must also exist in f .
grd	Specifies the group delay at f . grd is a vector of the same length as f .
w	Specifies the weight of each frequency point. w is a vector of the same length as f . The default is a vector in which each element has a value of 1.
r	Specifies the maximum value of the radius of any filter pole. r is a double-precision, floating-point number that must fall in the range (0, 1). The default is 0.9999.

Outputs

Name	Description
b	Returns the numerator of the designed IIR filter. b is a real vector with a length of n+1 .
a	Returns the denominator of the designed IIR filter. a is a real vector with a length of n+1 .
offset	Returns the group delay offset, which is the difference between the specified group delay and the approximated group delay. offset is a double-precision, floating-point number greater than zero.

Examples

```
[b, a] = iirgrpdelay(10, [0, 0.4], [0, 0.4], [9, 0]);figure;  
grpdelay(b, a);
```

```
[b0, a0] = ellip(7, 0.1, 60, 0.5);  
f = [0:0.4/256:0.4];  
grd0 = grpdelay(b0, a0, f*pi);  
[b, a] = iirgrpdelay(6, f, [0, 0.4], max(grd0)-grd0);  
comp_b = conv(b, b0);  
comp_a = conv(a, a0);  
figure;  
grpdelay(b0, a0);  
figure;  
grpdelay(comp_b, comp_a);
```

Related Topics

[iirlpnorm](#)

[iirlpnormc](#)

iirlpnorm (Digital Filter Design Toolkit, MathScript Function)

Owning Class: [singlerate](#)

Syntax

`[b, a] = iirlpnorm(n, d, f, e, mag)`

`[b, a] = iirlpnorm(n, d, f, e, mag, w)`

`[b, a] = iirlpnorm(n, d, f, e, mag, w, p)`

Description

Designs an [infinite impulse response \(IIR\) filter](#) that uses the [least p-th norm algorithm](#) to approximate the frequency response you specify.

[Examples](#)

Inputs

Name	Description
n	Specifies the order of the numerator. n is a nonnegative integer.
d	Specifies the order of the denominator. d is a nonnegative integer.
f	Specifies the frequency points. f is a vector whose values increase monotonically between 0 and 1.
e	Specifies the band edge frequencies. e is a vector whose values must also exist in f .
mag	Specifies the magnitude response of the filter at f . mag is a vector of the same length as f .
w	Specifies the weight of each frequency point. w is a vector of the same length as f . The default is a vector in which each element has a value of 1.
p	Specifies the value of p to use in the least p -th norm algorithm. p is a positive integer that must fall in the range [1, 128]. The default is 128.

Outputs

Name	Description
b	Returns the numerator of the designed IIR filter. b is a real vector with a length of n+1 .
a	Returns the denominator of the designed IIR filter. a is a real vector with a length of d+1 .

Examples

```
[b, a] = iirlpnorm(8, 8, [0, 0.2, 0.5, 0.6, 1], [0, 0.5, 0.6, 1], [1, 2, 1, 0, 0]);fft_mag  
= abs(fft(b, 16384)./fft(a, 16384));  
figure;  
plot(0:1/8192:1, fft_mag(1:8193));
```

```
[b, a] = iirlpnorm(6, 6, [0, 0.5, 0.6, 1], [0, 0.5, 0.6, 1], [1, 1, 0, 0], [1, 1, 1, 1],  
12);  
figure;  
freqz(b, a);  
figure;  
zplane(b, a);
```

Related Topics

[firlpnorm](#)

[iirlpnormc](#)

iirlpnormc (Digital Filter Design Toolkit, MathScript Function)

Owning Class: [singlerate](#)

Syntax

`[b, a] = iirlpnormc(n, d, f, e, mag)`

`[b, a] = iirlpnormc(n, d, f, e, mag, w)`

`[b, a] = iirlpnormc(n, d, f, e, mag, w, r)`

`[b, a] = iirlpnormc(n, d, f, e, mag, w, r, p)`

`[b, a, err, sos, gain] = iirlpnormc(n, d, f, e, mag)`

`[b, a, err, sos, gain] = iirlpnormc(n, d, f, e, mag, w)`

`[b, a, err, sos, gain] = iirlpnormc(n, d, f, e, mag, w, r)`

`[b, a, err, sos, gain] = iirlpnormc(n, d, f, e, mag, w, r, p)`

Description

Designs an [infinite impulse response \(IIR\) filter](#) that uses the [least p-th norm algorithm](#) to approximate the frequency response you specify. You can specify a pole radius constraint for the IIR filter.

[Examples](#)

Inputs

Name	Description
n	Specifies the order of the numerator. n is a nonnegative integer.
d	Specifies the order of the denominator. d is a nonnegative integer.
f	Specifies the frequency points. f is a vector whose values increase monotonically between 0 and 1.
e	Specifies the band edge frequencies. e is a vector whose values must also exist in f .
mag	Specifies the magnitude response of the filter at f . mag is a vector of the same length as f .
w	Specifies the weight of each frequency point. w is a vector of the same length as f . The default is a vector in which each element has a value of 1.
r	Specifies the maximum value of the radius of any filter pole. r is a double-precision, floating-point number that must fall in the range (0, 1). The default is 0.9999.
p	Specifies the value of p to use in the least p -th norm algorithm. p is a positive integer that must fall in the range [1, 128]. The default is 128.

Outputs

Name	Description
b	Returns the numerator of the designed IIR filter. b is a real vector with a length of n+1 .
a	Returns the denominator of the designed IIR filter. a is a real vector with a length of d+1 .
err	Returns the least p -th norm approximation error. err is a real number.
sos	Returns the second-order sections representation of the designed IIR filter. sos is an L -by-6 matrix, where L is the number of rows of the matrix. Each row of sos contains the coefficients of one filter section in the form $[b_0 \ b_1 \ b_2 \ 1 \ a_1 \ a_2]$.
gain	Returns the gain of the designed IIR filter. gain is a real number.

Examples

```
[b, a] = iirlpnormc(8, 8, [0, 0.2, 0.5, 0.6, 1], [0, 0.5, 0.6, 1], [1, 2, 1, 0, 0], [1, 1, 1, 1, 1], 0.9);fft_mag = abs(fft(b, 16384)./fft(a, 16384));  
figure;  
plot(0:1/8192:1, fft_mag(1:8193));
```

```
[b, a] = iirlpnormc(6, 6, [0, 0.5, 0.6, 1], [0, 0.5, 0.6, 1], [1, 1, 0, 0], [1, 1, 1, 1], 0.9, 12);  
figure;  
freqz(b, a);  
figure;  
zplane(b, a);
```

Related Topics

[firlpnorm](#)

[iirlpnorm](#)

iirnotch (Digital Filter Design Toolkit, MathScript Function)

Owning Class: [singlerate](#)

Syntax

$[b, a] = \text{iirnotch}(w, bw)$

$[b, a] = \text{iirnotch}(w, bw, ab)$

Description

Designs a second-order, [infinite impulse response \(IIR\) notch filter](#).

[Examples](#)

Inputs

Name	Description
w	Specifies the center frequency of the filter notch. w is a double-precision, floating-point number that must fall in the range (0, 1).
bw	Specifies the bandwidth of the filter notch at -ab dB. bw is a double-precision, floating-point number that must fall in the range (0, 1).
ab	Specifies the attenuation, in decibels, that corresponds to the bandwidth bw . ab is a double-precision, floating-point number greater than zero. The default is 3.0103, which corresponds to a 3 dB bandwidth, a commonly used bandwidth for a filter.

Outputs

Name	Description
b	Returns the numerator of the designed IIR filter. b is a three-element real vector.
a	Returns the denominator of the designed IIR filter. a is a three-element real vector.

Examples

```
[b, a] = iirnotch(0.1, 0.001);fft_mag = abs(fft(b, 16384)./fft(a, 16384));  
figure;  
plot(0:1/8192:1, fft_mag(1:8193));
```

Related Topics

[iircomb](#)

[iirpeak](#)

iirpeak (Digital Filter Design Toolkit, MathScript Function)

Owning Class: [singlerate](#)

Syntax

$[b, a] = \text{iirpeak}(w, bw)$

$[b, a] = \text{iirpeak}(w, bw, ab)$

Description

Designs a second-order, [infinite impulse response \(IIR\) peak filter](#).

[Examples](#)

Inputs

Name	Description
w	Specifies the center frequency of the filter peak. w is a double-precision, floating-point number that must fall in the range (0, 1).
bw	Specifies the bandwidth of the filter peak at -ab dB. bw is a double-precision, floating-point number that must fall in the range (0, 1).
ab	Specifies the attenuation, in decibels, that corresponds to the bandwidth bw . ab is a double-precision, floating-point number greater than zero. The default is 3.0103, which corresponds to a 3 dB bandwidth, a commonly used bandwidth for a filter.

Outputs

Name	Description
b	Returns the numerator of the designed IIR filter. b is a three-element real vector.
a	Returns the denominator of the designed IIR filter. a is a three-element real vector.

Examples

```
[b, a] = iirpeak(0.1, 0.001);fft_mag = abs(fft(b, 16384)./fft(a, 16384));  
figure;  
plot(0:1/8192:1, fft_mag(1:8193));
```

Related Topics

[iircomb](#)

[iirnotch](#)