

## List and Description of MATLAB Script Files

### 1. `add_2(n1,n2,b)`

`add_2(n1,n2,b)`, `n1` and `n2` are data samples to be added with `b` bits of precision.

Script file forms sum using 2-compl arithmetic with `b` bits of precision. Script is called by `hogen50`.

### 2. `cascade2z(del)`

Script file that performs arbitrary interpolation of input signal with 32 stage polyphase filter. Routine demonstrates arbitrary interpolator. First stage is 1-to-4 shaping filter. Second stage is 1-to-32 polyphase (4-taps per stage). Argument `del` is step size between 32 output stages. If `del < 1` further interpolation (try 0.29). If `del > 1`, down samples by `del` after 1-to-32 up sample resulting in `32/del` (try 4.2)

### 3. `eight_to_one`

Demonstration and comparison of various filters that reduce bandwidth eight-to-one for use as 8-to-1 down sampler. Bandwidth is +/- 20 kHz, transition bandwidth is 5 kHz with sample rate of 320 kHz. Filter and down sample 8-to-1. Compares following options, 7-th order IIR elliptic filter, 105 tap FIR filter, 9-coefficient two-path all pass filter, 8-path polyphase linear phase recursive all pass, 8-path polyphase non-uniform phase recursive all pass, 3-elliptic half band filters, 3-half band FIR filters, 3-exact half band FIR filters, and 3 non uniform phase recursive all-pass half band filters.

### 4. `farrow9g`

Script file demonstrating properties of Farrow filter. Can see prototype filter, polynomial approximations to segments, spectral response of approximation process, spectra of Farrow weights, and finally a 1-to-13 up sampling with 4-th order Taylor series.

5. filter\_8

Demonstration of efficient 1-to-8 up sampler. Input sample rate is 8 kHz, input bandwidth 3.5 kHz. 12 bit dynamic range. Recursive all-pass half band filter chain. First stage has 5 coefficients, second stage has 2-coefficients, and third stage has 2-coefficients. Total workload is approximately two multiplies per output sample point. Script file generates a comb of sinusoids spanning input frequency span.

6. filter\_ten\_a

Animated spectra and time response of 10-stage polyphase filter bank. Can move single tone through filter bank via a slider control or by a scheduled FM sweep. Can view time series in each channel or can view spectrum in each channel. Program starts by depressing one of 4 GUI push buttons. This file calls filter\_ten\_a\_call

7. filter\_ten\_a\_call

Script file called by filter\_ten\_a to animate time and spectral representation of input and output of 10-channel channelizers.

8. filter\_ten\_x(flag),

flag = 1 for 1/f stop band, flag = 0 for equiripple side lobes.

Script file for 10 path polyphase filter, presents time and frequency response of channels as well as time and frequency response of multiple tones residing in three of the channels. Also compares time, frequency, phase, and group delay of ten channels.

9. fredfil3(initval)

Script file, function fredfil3(a0) is called by polyz3 to operate interpolator example based on all-pass recursive filter.

#### 10. hilb\_dem

Script file demonstrating the frequency response, time response and pole-zero diagram of a half band filter as a base band and a quarter sample rate pass band filter converted to Hilbert transform filter. Illustrates time domain and spectral representation of sinusoid at input and output of Hilbert transform filter along with analytic signal representation of same signal.

#### 11. hogen\_16a

Script file demonstrating 1-to-16 up sampling with a 3-stage Hogenauer filter operating on a shaping filter with 4-samples per symbol. Time and frequency descriptions at various points in processing chain

#### 12. hogen\_16b

Script file demonstrating 1-to-16 up sampling with a 4-stage Hogenauer filter operating on a shaping filter with 4-samples per symbol. Time and frequency descriptions at various points in processing chain

#### 13. hogen\_16c

Script file demonstrating 1-to-16 up sampling with a 3-stage Hogenauer filter operating on a shaping filter with 5-samples per symbol. Time and frequency descriptions at various points in processing chain

#### 14. hogen\_50(mm,n\_bits)

hogen\_50(mm,n\_bits), mm is down sampling ratio less than 50,  
n-bits is bit width of accumulators

4-stage Hogenauer filter for mm to 1 down sampling. Integer implementation simulating 2-complement arithmetic with n\_bit accumulators. Script file calls add\_2 script file. Hogen50 (mm, n\_bits) try hogen50(20, 23) then hogen50(20, 22)

#### 15. interp\_x1

Script file, GUI driven demonstration of arbitrary resampling of input signal by 5-path, 4th order Farrow filter. Figure presents time and frequency descriptions of input and output waveforms. GUI sliders change input frequency and up-sample ratio, while pushbuttons enable zoomed frequency response as well as selects sinewave or triangle wave. Can see spectral artifacts rise as frequency of input signal exceeds design limit. Script file calls interp\_x1\_call.

#### 16. interp\_x1\_call

Script file that performs arbitrary resampling with 5-path, 4th order Farrow Filter. Designed for 60 dB artifact levels

#### 17. interp\_x2

Script file, GUI driven demonstration of arbitrary resampling of input signal by 128-path polyphase filter. Figure presents time and frequency descriptions of input and output waveforms. GUI sliders change input frequency and up-sample ratio, while pushbuttons enable zoomed frequency response as well as selects sinewave or triangle wave. Can see spectral artifacts rise as frequency of input signal exceeds design limit. Script file calls interp\_x1\_call.

#### 18. interp\_x2\_call

Script file that performs arbitrary resampling with 128 path Filter. Designed for 60 dB artifact levels

#### 19. iso\_filter

Fractional octave bandwidth ISO standard filter bank implemented by recursive all-pass band pass filters, along with recursive all-pass low-pass half band filters.

#### 20. iso\_filter\_2

Fractional octave filter bank, 12 filters per octave implemented as passband recursive all-pass filter bank and recursive all-pass half band filters that down sample successive octaves into filter bank at successively reduced sample rates.

## 21. lineardesign\_2

Graphical user interface (GUI) driven script file presents coefficients of M-path recursive all-pass filters with approximately linear phase. Two-path filters are an obvious subset of this filter design. Five figures are formed by lineardesign\_2. These include the GUI figure which presents the frequency response, a figure that presents the impulse response, a figure that presents the phase response, a figure that presents the group delay response, and the pole-zero diagram of the designed filter. The GUI allows selection of number of paths, number of coefficients per path, and the stop band frequency edge. There are parameters used by the design routine that can be altered by the user, these include, number of iterations in the design process, a parameter initialized to 20, the frequency resolution of the process, a parameter initialized at 300. After a response to the design cycle initiated by depressing the *compute* button, the user can depress the *reiterate* button to continue with additional iterations. The initial design exhibits 1/f spectra side lobe response while the reiterated design converges to equal-ripple stop band response. One additional option permits the user to move the stop band zeros to alter the stop band behavior. This routine calls 10 other script files which are bundled with the lineardesign\_2 script file in a sub file called lineardesign.

The M-path filter designed by this routine contains first order and second order polynomials in  $Z^M$ . The denominators of these polynomials are of the form:

$$Z^M + a_1,$$

and

$$Z^{2M} + a_1 Z^M + a_2$$

The zero path contains delay only with the denominator of the form

$$Z^{(R*M*(M-1))}$$

Where M is the number of paths and R is the number of second order segments in each path. Each of the remaining M-1 paths also has a cascade delay of the form:

$$\text{Path}(r) \text{ delay} = Z^{(-r)}, \quad r = 1, 2, 3, \dots, M-1$$

When the filter is used as a resampling filter, invoking the noble identity, the polynomials revert to first and second polynomials in Z, the zero path reverts to delay  $Z^{(-R*(M-1))}$  and the indexed path delays are absorbed in the equivalent input or output commutator.

The denominator roots and the denominator coefficients of each path are presented to the MATLAB command window when the *compute* button is depressed on the GUI figure.

## 22. lin\_p2

Script shows comparison of two high performance half band filters, FIR and linear phase recursive all-pass filter. Recursive half band designed with `lineardesign_2`. Various figures compare roots of two filters, frequency response, time response, phase response, group delay response, in band amplitude and group delay ripple.

## 23. myfrf

Script file that modifies `remez` algorithm so that low pass filters designed by the `remez` algorithm exhibit  $1/f$  rate of side lobe decay. Used as:

```
h1 = remez(49,[0 100 200 1000]/1000,{'myfrf',[1 1 0 0]],[1 10])
```

as opposed to standard call

```
h2 = remez(49,[0 100 200 1000]/1000,[1 1 0 0],[1 10])
```

## 24. poly\_phase

Script file presents an illustration of how Nyquist zones in a 10-to-1 down sampled signal acquire unique phase profiles as a function of input time delay. This is an animated view of adjacent Nyquist zones:

## 25. polyphase\_30a

Script file illustrating 30 stage polyphase partition. Signal is followed through resampling steps. Signal is seen at input rate, at down sampled and aliased output rate in two polyphase arms, and at down sampled and phase corrected un-aliased output rate. A sparse signal set lets us examine residual spectral levels in empty spectral bands.

## 26. polyphase\_50a

Script file illustrating 50 channel channelizers. Tones in each channel are separated and time response and spectra of each channel are displayed

## 27. polyphs2

Function demonstrates 1-to-5 up sampling polyphase fir filter. Figures show amplitude and phase responses of prototype filter and polyphase partition. Figures track a sinusoid at various points in 1-to-5 up sample process.

## 28. polyz3

Script file polyz3 forms cascade of two recursive all-pass filters and adjusts the parameter alpha in each to demonstrate use the filters as a variable time delay network. Script calls fredfil3(alpha). Animated display showing phase and group delay of filter along with spectrum and time response of signal being filtered by all pass network.

## 29. nyq\_4

Script file that presents coefficients of a square-root Nyquist filter designed with the remez algorithm. By not restricting the taper to match the cosine taper the filter exhibits significantly lower out of band side-lobe levels and reduced level of in-band ripple. Used as:

```
h1 = nyq_4(f_symbol, f_smpl, alpha)
```

and in particular,

```
h1 = nyq_4(1, 4, 0.25)
```

designs a square-root Nyquist filter with 4-samples per symbol, with roll off equal to 25% of symbol rate. When running the program, a prompt requests the user to enter a desired filter length in symbols and suggests a good length. If the user enters the suggested prompt value, the resulting filter will exhibit -80 dB side lobe levels with a 1/f fall off. If the user responds with a larger or small value than the suggested prompt level, the filter design will exhibit greater or lesser level side lobe levels. Script file calls standard remez algorithm and myfrf script file.

### 30. receiver\_40a

Script file illustrating 40 channel polyphase filter bank to access 30 channels with symbol rates of 20 MHz, separated by 28 MHz centers (1.4 times symbol rate). Input sample rate is  $40 \times 28 = 1120$  MHz. The channelizers performs a 40 point transform on the output of a 40-stage polyphase filter with the polyphase filter operating at input rate but with an output rate, oft 2 samples/symbol or 40 MHz. The resampling rate is  $1120/40 = 28$ -to-1, thus output from the 40-channels are computed once for every 28 input samples. Channelizer is not a matched filter, but rather is prototype filter 10% wider than two sided bandwidth of input signal to accommodate frequency uncertainty of separate channel centers.

### 31. remez\_myfrf\_example

Script file to illustrate the difference in filter design when using standard remez algorithm call and the modified call using the myfrf in remez call. See myfrf entry. Two figures generated by file are the frequency response of a pair of filters designed by the two options.

### 32. sig\_del\_ex\_1

Script file illustrating two stage resampling filter at the output of high quality sigma-delta modulator. Resampling filter is a 5-to-1 three stage CIC filter followed by a 3-to-1 polyphase FIR filter with CIC  $\sin(x)/x$  correction.

### 33. six\_stage\_4

Script file illustrating three ways to achieve 64-to-1 down sampling of the output of a sigma-delta modulator. First choice is 6 stages of half band filtering, second choice is 2 stages of 8-to-1 polyphase filters, and third choice is a single 64-to-1 polyphase filter. The 64-to-1 is presented in two options, equiripple design and shaped design to match noise shaping of sigma-delta modulator. Successive figures are displayed showing spectra of each stage and then spectra of filtered time series. There is a lengthy pause while the sigma-delta modulated time series is generated for use with the filters. It is worth the wait to see the signal reconstructed by successive reductions in noise bandwidth.



#### 34. time\_10a

Script file to illustrate Farrow like continuous time delay engine formed by recursive all-pass filter coefficients. The polynomial description of the time delay coefficient sets are derived from a ten-path polyphase recursive all-pass filter bank designed by lineardesign\_2.

#### 35. tony\_c

Script file that presents coefficients of two path recursive all-pass filter. Used as `tony_c(n_order, w_stop)`, where `n_order` defines the number of poles which must be an odd positive integer and where `w_stop` is the start of the stop band frequency normalized by the sample rate, `fsmpl`. A sample call would be `tony_c(7,0.30)`.

#### 36. tonycxx

Script file called by `tony_des_2`. This script file computes the coefficients of the two-path recursive all-pass filters. The file also applies the low-pass to low-pass transformation that converts the half-band filter to an arbitrary bandwidth filter. It also applies the low-pass to band-pass transformation that converts an arbitrary bandwidth low pass filter to a band pass filter. See `tony_des_2`.

#### 37. tony\_des\_2

Graphical user interface (GUI) driven script file presents coefficients of two-path recursive all-pass filters. The two path filter options include half-band filters, an option to tune the half-band filter to an arbitrary bandwidth, an option to tune the filter to an arbitrary center frequency, and an option to zero pack the impulse response for iterated filter designs. Two figures are formed by `tony_des_2`, the frequency response and the pole-zero diagram of the designed filter. The script file `tony_des_2` a second script file `tonycxx`.

### 38. two\_path\_demo

Script file to illustrate the phase relationship between the two paths of a two-path, recursive all pass filter. The file generates 5 figures showing the phase response of the two paths and the magnitude response obtained by adding the outputs from the two paths. The particular filters are a linear phase, recursive half band low pass filter, a non uniform phase, recursive half band low pass filter, a non uniform phase, recursive 10% bandwidth low pass filter, a non uniform phase, recursive 2.5% bandwidth low pass filter, and a non uniform phase, recursive 15% bandwidth band pass filter.