Filter Calculations

1.1 All Pass

i. Filter Coefficients

b0 = 1;
b1 = 0;
a1 = 0;
b2 = 0;
a2 = 0;

1.2 Phase Shift

Note: Matlab equations call this All Pass. We are calling it Phase Shift in Filter GUI

i. Filter Parameters

BW = Bandwidth in Hz
Fc = Center Frequency in Hz
Fs = Sample rate in Hz

ii. Error Checking

0 <= BW <= Fs/2
0 <= Fc <= Fs/2
0 <= Fs <= 192K

iii. Equation

a = (1 - tan(pi * BW / Fs)) / (1 + tan(pi * BW / Fs));
d = - cos(2 * pi * Fc / Fs);

iv. Filter Coefficients

b0 = a;
b1 = d * (1 + a);
a1 = b1;
b2 = 1;
a2 = a;

B = [b0 b1 b2];
A = [1 a1 a2];

All coefficients are in a 5.23 format that is packed in a 32 bit word

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Unused Sign Integer Fraction	U	U	U	U	S	I	I	I	I	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
Max Negative	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Max Positive	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

1.3 EQ

i. Filter Parameters

G = Filter Gain in dB
Fc = Filter Center Frequency in Hz
BW = Filter Bandwidth in Hz
Fs = Sample rate in Hz

ii. Error Checking

0 <= BW <= Fs/2
0 <= Fc <= Fs/2
0 <= Fs <= 192K
-140 <= G <= 48

iii. Equation

```
 A = 10^(G / 20)

if A < 1
   a = (tan(pi * BW / Fs) - A) / (tan(pi * BW / Fs) + A);
else
   a = (tan(pi * BW / Fs) - 1) / (tan(pi * BW / Fs) + 1);
end

H = A  -  1;
d =  - cos(2 * pi * Fc / Fs);
```

iv. Filter Coefficients

```
b0 = 1  +  (1 + a) * H / 2;
b1 = d * (1 - a);
a1 = b1;
b2 = ( - a - (1 + a) * H / 2);
a2 =  - a;

B = [b0 b1 b2];
A = [1 a1 a2];
```

All coefficients are in a 5.23 format that is packed in a 32 bit word

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Unused Sign Integer Fraction	U	U	U	U	S	I	I	I	I	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
Max Negative	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Max Positive	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

1.4 Notch

i. Filter Parameters

BW = Notch Bandwidth in Hertz
Fc = Notch Center Frequency in Hz
Fs = Sample rate in Hz

ii. Error Checking

0 <= BW <= Fs/2

0 <= Fc <= Fs/2

0 <= Fs <= 192K

iii. Equation

a = (1 - tan(pi * BW / Fs)) / (1 + tan(pi * BW / Fs));
d = - cos(2 * pi * Fc / Fs);

b0 = a;
b1 = d * (1 + a);
a1 = b1;
b2 = 1;
a2 = a;

iv. Filter Coefficients

B = [b0 b1 b2];
A = [1 a1 a2];
B = 0.5 * (B + A);

All coefficients are in a 5.23 format that is packed in a 32 bit word

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Unused Sign Integer Fraction	U	U	U	U	S	I	I	I	I	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
Max Negative	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Max Positive	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

1.5 Treble Shelf

i. Filter Parameters

Fs = Sample rate in Hz
G = Treble Shelf Gain in dB
Fc = Treble Shelf Corner Frequency in Hz

ii. Error Checking

0 <= Fc <= Fs/2
0 <= Fs <= 192K
-24 <= G <= 24

iii. Equation

```
g= 10^(G / 20));

s = sqrt(2) / 2;
rho = pi / 2;
phi = (Fc / Fs) * pi;

A = g;

G = 20 * log10(A);

if G >  - 6 & G < 6
  F = sqrt(A);
elseif A > 1
  F = A  /  sqrt(2);
else
  F = A  *  sqrt(2);
end

gd = ((F^2  -  1) / (A^2  -  F^2))^0.25;
gn = sqrt(A)  *  gd;

a = tan(pi * (Fc / Fs  -  1 / 4));

b0 = (gn^2 * a^2 + 2 * s * gn - 2 * gn^2 * a + 1 - 2 * s * gn * a^2 + a^2
+ gn^2 + 2 * a) / (1 + gd^2 + 2 * s * gd - 2 * s * gd * a^2 + gd^2 * a^2 -
2 * gd^2 * a + a^2 + 2 * a);
b1 = (2 - 2 * gn^2 * a^2 + 4 * gn^2 * a + 4 * a - 2 * gn^2 + 2 * a^2) / (1
+ gd^2 + 2 * s * gd - 2 * s * gd * a^2 + gd^2 * a^2 - 2 * gd^2 * a + a^2
+ 2 * a);
b2 = (1 + 2 * s * gn * a^2 - 2 * s * gn + 2 * a + a^2 + gn^2 - 2 * gn^2 * a
+ gn^2 * a^2) / (1 + gd^2 + 2 * s * gd - 2 * s * gd * a^2 + gd^2 * a^2 - 2
* gd^2 * a + a^2 + 2 * a);

a0 = 1;
a1 = (2 - 2 * gd^2 * a^2 + 4 * gd^2 * a + 2 * a^2 - 2 * gd^2 + 4 * a) / (1
+ gd^2 + 2 * s * gd - 2 * s * gd * a^2 + gd^2 * a^2 - 2 * gd^2 * a + a^2
+ 2 * a);
```

a2 = (1 - 2 * gd^2 * a + 2 * a + gd^2 - 2 * s * gd + a^2 + gd^2 * a^2 + 2 * s * gd * a^2) / (1 + gd^2 + 2 * s * gd - 2 * s * gd * a^2 + gd^2 * a^2 - 2 * gd^2 * a + a^2 + 2 * a);

iv. Filter Coefficients

B = [b0 b1 b2];
A = [a0 a1 a2];

All coefficients are in a 5.23 format that is packed in a 32 bit word

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Unused Sign Integer Fraction	U	U	U	U	S	I	I	I	I	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
Max Negative	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Max Positive	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

1.6 Bass Shelf

i. Filter Parameters

Fs = Sample rate in Hz
G = Bass Shelf Gain in dB
Fc = Bass Shelf Corner Frequency in Hz

ii. Error Checking

0 <= Fc <= Fs/2
0 <= Fs <= 192K
-24 <= G <= 24

iii. Equation

```
g= 10^(G / 20));

s = sqrt(2) / 2;
rho = pi / 2;
phi = (Fc / Fs) * pi;

A = g;

G = 20 * log10(A);

if G >  - 6 & G < 6
   F = sqrt(A);
elseif A > 1
   F = A  /  sqrt(2);
else
   F = A  *  sqrt(2);
end

gd = ((F^2  -  1) / (A^2  -  F^2))^0.25;
gn = sqrt(A)  *  gd;

a = tan(pi * (Fc / Fs  -  1 / 4));

b0 =  - ( - 1 - gn^2 * a^2 - a^2 - 2 * gn^2 * a - gn^2 - 2 * s * gn + 2 * s *
gn * a^2 + 2 * a) / (2 * s * gd + 1 - 2 * s * gd * a^2 + gd^2 * a^2 + 2 *
gd^2 * a + a^2 + gd^2 - 2 * a);
b1 =  - (2 - 4 * a - 4 * gn^2 * a - 2 * gn^2 * a^2 - 2 * gn^2 + 2 * a^2) / (2
* s * gd + 1 - 2 * s * gd * a^2 + gd^2 * a^2 + 2 * gd^2 * a + a^2 + gd^2
- 2 * a);
b2 = (1 + 2 * s * gn * a^2 - 2 * a + gn^2 - 2 * s * gn + 2 * gn^2 * a + a^2
+ gn^2 * a^2) / (2 * s * gd + 1 - 2 * s * gd * a^2 + gd^2 * a^2 + 2 * gd^2
* a + a^2 + gd^2 - 2 * a);

a0 = 1;
a1 = ( - 2 + 2 * gd^2 * a^2 + 4 * gd^2 * a - 2 * a^2 + 2 * gd^2 + 4 * a) /
(2 * s * gd + 1 - 2 * s * gd * a^2 + gd^2 * a^2 + 2 * gd^2 * a + a^2 +
gd^2 - 2 * a);
```

a2 = (gd^2 * a^2 - 2 * a + 1 + 2 * gd^2 * a - 2 * s * gd + a^2 + 2 * s * gd * a^2 + gd^2) / (2 * s * gd + 1 - 2 * s * gd * a^2 + gd^2 * a^2 + 2 * gd^2 * a + a^2 + gd^2 - 2 * a);

iv. Filter Coefficients

B = [b0 b1 b2];
A = [a0 a1 a2];

All coefficients are in a 5.23 format that is packed in a 32 bit word

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Unused Sign Integer Fraction	U	U	U	U	S	I	I	I	I	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
Max Negative	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Max Positive	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

1.7 2^{nd} order Linkwitz Riley

i. Filter Parameters

Fs = Sample rate in Hz

Fc = Cut Frequency in Hz

HL= LR Filter type (high, low)

ii. Error Checking

0 <= Fc <= Fs/2

0 <= Fs <= 192K

HL= (high, low)

iii. Equation

```
wc = 2 * pi *  Fc;

if HL(1:3) == 'low'
        Ba = [0   0   wc^2];
        Aa = [1   2 * wc   wc^2];
else
        Ba = [1   0    0];
        Aa = [1 2 * wc   wc^2];
end

k = 2 * pi * fc / tan( pi * fc/Fs);

B = [Ba(1) * k^2 + Ba(3) + Ba(2) * k, -2 * Ba(1) * k^2 + 2 * Ba(3),   -Ba(2)
*  k + Ba(1) *  k^2 + Ba(3)];

A = [Aa(1) * k^2 + Aa(3) + Aa(2) * k, -2 * Aa(1) * k^2 + 2 * Aa(3),   -Aa(2)
* k + Aa(1) *  k^2 + Aa(3)];
```

iv. Filter Coefficients

```
B = B / A(1);
A = A / A(1);
```

All coefficients are in a 5.23 format that is packed in a 32 bit word

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Unused Sign Integer Fraction	U	U	U	U	S	I	I	I	I	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
Max Negative	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Max Positive	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

1.8 2nd Order Variable Q

i. Filter Parameters

Fs = Sample rate in Hz
Fc = Cut Frequency in Hz
HL= LR Filter type (high, low)
Q = Filter Q (s^2 + (wc / Q) * s + wc^2)

ii. Error Checking

0 <= Fc <= Fs/2
0 <= Fs <= 192K
HL= (high, low)
0 <= Q <= 100

iii. Equation

```
wc = 2 * pi * Fc;

if HL(1:3) == 'low'
        Ba = [0  0   wc^2];
        Aa = [1  wc/Q   wc^2];
else
        Ba = [1   0   0];
        Aa = [1   wc / Q   wc^2];
end

k = 2 * pi *Fc / tan(pi * Fc / Fs);

B = [Ba(1) * k^2 + Ba(3) + Ba(2) * k, -2 * Ba(1) * k^2 + 2 * Ba(3),   -Ba(2)
* k + Ba(1) * k^2 + Ba(3)];

A = [Aa(1) * k^2 + Aa(3) + Aa(2) * k, -2 * Aa(1) * k^2 + 2 * Aa(3),   -Aa(2)
* k + Aa(1) * k^2 + Aa(3)];
```

iv. Filter Coefficients

```
B = B / A(1);
A = A / A(1);
```

All coefficients are in a 5.23 format that is packed in a 32 bit word

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Unused Sign Integer Fraction	U	U	U	U	S	I	I	I	I	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
Max Negative	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Max Positive	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

1.9 2nd order Butterworth Filter (Use Variable Q w/ Q=.707)

Make 2nd Order Butterworth Filter using the Variable Q

Butterworth is a variable Q with Q = .707

1.10 2nd order Bessel Filter (Use Variable Q w/ Q=.58)

Make 2nd Order Bessel Filter using the Variable Q
Bessel is a variable Q with Q = .58

1.11 1st First Order Butterworth

i. Filter Parameters

Fs = Sample rate in Hz

Fc = Cut Frequency in Hz

HL= LR Filter type (high, low)

ii. Error Checking

0 <= Fc <= Fs/2

0 <= Fs <= 192K

HL= (high, low)

iii. Equation

k = (2 * pi * Fc) / tan((pi * Fc) / Fs);

Wc = (2 * pi * Fc)

If HL(1:3) = 'low'

b0 = Wc /(k + Wc);
b1 = Wc / (k + Wc);

else

b0 = k / (k + Wc);
b1 = - k / (k + Wc);

end

a1 = (Wc – k) / (k + Wc);

iv. Filter Coefficients

B = [b0 b1 0]
A = [1 a1 0]

All coefficients are in a 5.23 format that is packed in a 32 bit word

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Unused Sign Integer Fraction	U	U	U	U	S	I	I	I	I	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
Max Negative	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Max Positive	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

1.12 2nd Order Chebychev

(This needs equation for ripple in stop band)

i. Filter Parameters

```
Fs = Sample rate in Hz
rip = Ripple specification in dB
typ = Filter type (high, low, stop )

 if typ(1:3) == 'sto'
          Fc = Stop band  Input Lower and upper frequencies [f1, f2]
  else
           Fc = Cut frequency in Hz
           nrm = If Scale peak to 0dB == 1,  If Scale PB to 0dB ==  - 1
           if nrm ~= 1
                  rip = rip *  - 1;
            end

  end
```

ii. Error Checking

```
0 <= Fc <= Fs/2
0 <= Fs <= 192K
0 <= rip <= 10
Nrm = 1,  1
HL = high, low, stop
```

iii. Equation

```
if typ(1:3) == 'sto'

    Call  cheby1(ord, rip, 2 * Fc / Fs, HL);

else

     Call soCHBI(Fc, Fs, rip, HL);

 End

 Function  soCHBI(Fc, Fs, rip, HL)

 if sign(rip) > 0
   sf = 1;
 else
   sf = 0;
 end

 R = abs(rip);
```

```
if R == 0
  B = [1  0  0];
  A = [1  0  0];
else

  wc = 2 * pi * Fc;
  epsilon = sqrt(10^(R / 10)  -  1);
  alpha = asinh(1 / epsilon) / 2;
  beta1 = 3 * pi / 4;
  beta2 = 5 * pi / 4;

  s1 = (sinh(alpha) * cos(beta1)  +  cosh(alpha) * sin(beta1) * i);
  s2 = (sinh(alpha) * cos(beta2)  +  cosh(alpha) * sin(beta2) * i);
  a = real(s1 + s2);
  b = real(s1 * s2);
  c = b;
  if sf
    c = c / sqrt(1 + epsilon^2);
  end

  if HL(1:3) == 'low'
    Ba = [0 0 c * wc^2];
    Aa = [1 wc * a b * wc^2];
  else
    Ba = [c / b 0 0];
    Aa = [1 wc * a / b (wc^2) / b];
  end

  k = 2 * pi * Fc / tan(pi * Fc / Fs);
  B = [Ba(1) * k^2 + Ba(3) + Ba(2) * k,  - 2 * Ba(1) * k^2 + 2 * Ba(3),    -
Ba(2) * k + Ba(1) * k^2 + Ba(3)];
  A = [Aa(1) * k^2 + Aa(3) + Aa(2) * k,  - 2 * Aa(1) * k^2 + 2 * Aa(3),    -
Aa(2) * k + Aa(1) * k^2 + Aa(3)];

  B = B / A(1);
  A = A / A(1);
```