Matching of Poles and Zeros of FE SVF to TR SVF

created : 3rd March 2012
updated : 4th Feb 2016 (tidy up for public viewing, added comments, moved working to helper function section, removed high, band, notch sections)
updated: 4th Feb 2016 (added output mixing version)

The basic idea is to calculate the positions of the poles and then solve for a transformation of the input cutoff and damping of the forward Euler version to match that of the Trapezoidal version. This matching keeps the filter stable across the entire frequency range. To match the response additional gain and or low pass FIRs are required, the position of these corrections will be influenced by implementation details including non-linearies if they are added to the filter.

I classify this as a “semi-implicit” method, since it arrives at the implicit solution in the linear time invariant case, but differs from the full implicit solution if non-linearities and modulation are added, although the error is much better than abstract structures like direct form 1 biquads.

The same method can be applied to the Sallen Key type structure, or indeed any two pole filter, but symbolic solution of cascade type structures don’t work out nearly as neat.

Helper Functions

```
ClearAll["Global`*"]
TransferFunctionAndPoles[resp_] := Block[{},
    t = TransferFunctionModel[resp, z, SamplingPeriod → 1 / 2];
    s = TransferFunctionPoles[t][[1, 1]] // Simplify;
    Return[{t, s}];
   ];
PlotTransferFunctionAndPoles[t1_, s1_, t2_, s2_] := Block[{g1, g2},
    {Flatten[{Table[Show[Table[ParametricPlot[{Re[s1[[sln]]], Im[s1[[sln]]]} /.
              {g1 → 2 Sin[π w]}, {w, 0, 1 / 2}, AspectRatio → 1,
             PlotRange → {{-1, 1}, {-4, 4}}], {d1, 1, 2, 1}]], {sln, 1, 2, 1}],
       Show[Table[RootLocusPlot[t1, {g1, 0, 2}, PlotRange → All,
          Epilog → Circle[{0, 0}, 1]], {d1, 0, 2, 1}], AspectRatio → 0.6]}
     ],
     Flatten[{Table[Show[Table[ParametricPlot[{Re[s2[[sln]]], Im[s2[[sln]]]} /.
              {g2 → Tan[π w]}, {w, 0, 1 / 2}, AspectRatio → 1,
             PlotRange → {{-1, 1}, {-1, 1}}], {d2, 1, 2, 1}]], {sln, 1, 2, 1}],
       Show[Table[RootLocusPlot[t2, {g2, 0, 2}, PlotRange → All,
          AspectRatio → 0.6, Epilog → Circle[{0, 0}, 1]], {d2, 0, 2, 1}]]}]
    }
   ];
MatchPoles[resp1_, resp2_, s1_, s2_, poleorder_] :=
  Block[{gb, z, g2, wc, poles, sln3, sln4, gain},
   (* match poles*)poles =
    Solve[{s1[[poleorder[[1]]]] == s2[[1]], s1[[poleorder[[2]]]] == s2[[2]]},
      {g1, d1}][[2]] // Simplify;
   Return[poles];
  ];
MatchGain[resp1_, resp2_, poles_] :=
  Block[{gb, z, g2, wc, sln3, sln4, gain, d2},
   (* match gain*)
   sln3 = (((gb resp1) /. {z → i g2}) /. (poles))
     (((gb resp1) /. {z → -i g2}) /. (poles)) // FullSimplify;
   sln4 = ((resp2) /. {z → i g2}) ((resp2) /. {z → -i g2}) // FullSimplify;
   gain = FullSimplify[gb /. Solve[sln3 == sln4, gb][[2]], {g2 > 0, d2 > 0}];
   Return[gain];
  ];
PlotPoleMapping[poles_] := Block[{g1, g2, d1, d2},
    {Show[Table[Plot[(g1 /. poles) /. {g2 → Tan[π w]}, {w, 0, 1 / 2}],
       {d2, -1 / 2, 3, 1 / 4}]],
     Show[Table[Plot[(d1 /. poles) /. {g2 → Tan[π w]}, {w, 0, 1 / 2},
       PlotRange → All], {d2, -1 / 2, 3, 1 / 4}]]}
  ];
PlotResponses[resp1_, resp2_, poles_, gain_, damp_] := Block[{g2, d2, z, dB},
    dB[x_] := Log[2, Abs[x] + 1*^-6];
    Show[Table[LogLinearPlot[
       {dB[(gain resp1 /. poles) /. ({g2 → Tan[π wc], d2 → damp, z → Exp[2 π i w]})],
        dB[(resp2) /. {g2 → Tan[π wc], d2 → damp, z → Exp[2 π i w]}]},
       {w, 0.01, 0.5}, PlotRange → {-6, 6}], {wc, 0.1, 0.4, 0.1}]]
  ];
sub = {v0z → v0 z^-1, v1z → v1 z^-1, v2z → v2 z^-1,
    v1zz -> v1 z^-2, v2zz -> v2 z^-2, v1zzz → v1 z^-3, ic1eqz → ic1eq z^-1,
    ic2eqz → ic2eq z^-1, g1z → g1 z^-1, d1z → d1 z^-1, g2z → g2 z^-1, d2z → d2 z^-1};
```

Low SVF Matching of FE to TR

Low Pass, damping limit 0, averaging FIR on input and output

This is the standard Chamberlin form, where the current times bandpass output is computed and then used to compute the current low pass output

Difference equations

```
svffeeqn = {v1 == v1z + g1 (1 / 2 (v0 + v0z) - d1 v1z - v2z),
    v2 == v2z + g1 (v1),  lp == (1 / 2 ( v2 + v2z)) / v0} /. sub;
svffe = lp /. Solve[svffeeqn, {lp}, {v0, v1, v2}][[1]] // FullSimplify

svftreq = {vhp == v0 - d2 v1 - v2,  0 == g2 (vhp - 0) - (v1 - 0) + ic1eq,
    0 == g2 (v1) - (v2 - 0) + ic2eq,  ic1eq == (2 (v1 - 0) - ic1eq) z^-1,
    ic2eq == (2 (v2 - 0) - ic2eq) z^-1,  lp == (v2) / v0} /. sub;
svftr = lp /. Solve[svftreq, {lp}, {vhp, v0, v1, v2, ic1eq, ic2eq}][[1]] //
  FullSimplify
```

$$\frac{g1^2\ (1+z)^2}{4\ \left(1+d1\ g1\ (-1+z)+z\ \left(-2+g1^2+z\right)\right)}$$

$$\frac{g2^2\ (1+z)^2}{(-1+z)^2+g2^2\ (1+z)^2+d2\ g2\ \left(-1+z^2\right)}$$

Plot of Root Locus to match roots

```
{tfe, sfe} = TransferFunctionAndPoles[svffe]
{ttr, str} = TransferFunctionAndPoles[svftr]
PlotTransferFunctionAndPoles[tfe, sfe, ttr, str]
```

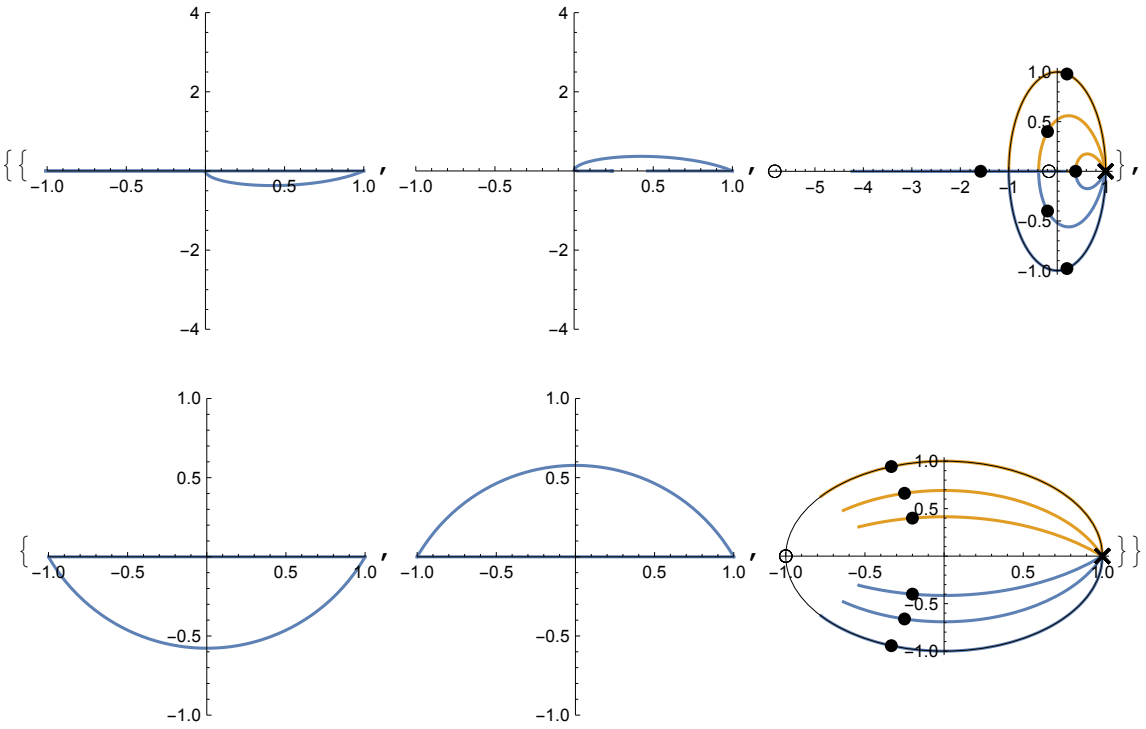
$$\left\{\left(\frac{g1^2\ (1+z)^2}{4\ \left(1+d1\ g1\ (-1+z)+z\ \left(-2+g1^2+z\right)\right)}\right)_{\frac{1}{2}}^{\mathcal{T}},\right.$$

$$\left\{\frac{1}{2}\left(2-d1\ g1-g1^2-g1\sqrt{-4+d1^2+2\ d1\ g1+g1^2}\right),\right.$$

$$\left.\left.\frac{1}{2}\left(2-d1\ g1-g1^2+g1\sqrt{-4+d1^2+2\ d1\ g1+g1^2}\right)\right\}\right\}$$

$$\left\{\left(\frac{g2^2\ (1+z)^2}{(-1+z)^2+g2^2\ (1+z)^2+d2\ g2\ \left(-1+z^2\right)}\right)_{\frac{1}{2}}^{\mathcal{T}},\right.$$

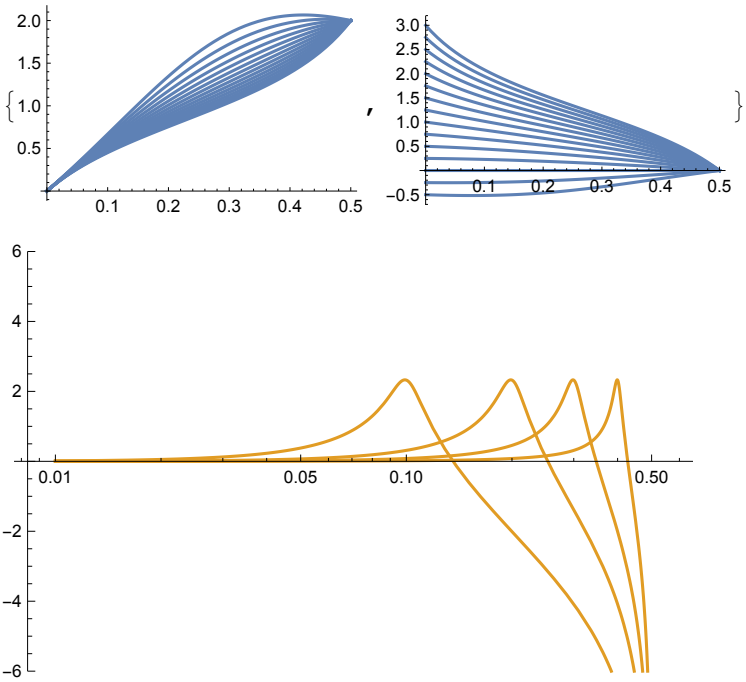
$$\left.\left\{-\frac{-1+g2^2+\sqrt{\left(-4+d2^2\right)\ g2^2}}{1+d2\ g2+g2^2},\ \frac{1-g2^2+\sqrt{\left(-4+d2^2\right)\ g2^2}}{1+d2\ g2+g2^2}\right\}\right\}$$



Solve to transform poles of FE to poles of TR

```
poleorderfe = {1, 2};
poles = MatchPoles[svffe, svftr, sfe, str, poleorderfe]
PlotPoleMapping[poles]
PlotResponses[svffe, svftr, poles, gain, 0.2]
```

$$\left\{g1 \to \frac{2\ g2}{\sqrt{1 + d2\ g2 + g2^2}},\ d1 \to \frac{d2}{\sqrt{1 + d2\ g2 + g2^2}}\right\}$$



Psuedo Code

```
init:
g2 = tan(pi*cutoff/samplerate)
d2 = 2 - 2*res
nrm = 1/sqrt (1 + d2*g2 + g2*g2)
g = 2*g2*nrm
d = d2*nrm

clear:
v0z = 0;
v1 = 0;
v2 = 0;

tick:
v0 = input
v1z = v1
v2z = v2
v1 = v1z + g*(0.5*(v0 + v0z) - d*v1z - v2z)
v2 = v2z + g*(v1)
output = 0.5*(v2 + v2z)
v0z = v0
```

Low Pass, damping limit 1, averaging FIR on input and inside structure on band pass

This form should give the best results when non-linearities are added to the circuit as the low pass FIR appears on both the input and inside the structure on the input to the low pass integrator

Difference equations

```
svffeeqn = {v1 == v1z + g1 (1 / 2 (v0 + v0z) - d1 v1z - v2z),
    v2 == v2z + g1 (1 / 2 (v1 + v1z)), lp == (1 (v2 + 0 v2z)) / v0} /. sub;
svffe = lp /. Solve[svffeeqn, {lp}, {v0, v1, v2}][[1]] // FullSimplify

svftreq = {vhp == v0 - d2 v1 - v2, 0 == g2 (vhp - 0) - (v1 - 0) + ic1eq,
     0 == g2 (v1) - (v2 - 0) + ic2eq, ic1eq == (2 (v1 - 0) - ic1eq) z^-1,
     ic2eq == (2 (v2 - 0) - ic2eq) z^-1, lp == (v2) / v0} /. sub;
svftr = lp /. Solve[svftreq, {lp}, {vhp, v0, v1, v2, ic1eq, ic2eq}][[1]] //
  FullSimplify
```

$$\frac{g1^2\ (1+z)^2}{2\left(2\ d1\ g1\ (-1+z)+2\ (-1+z)^2+g1^2\ (1+z)\right)}$$

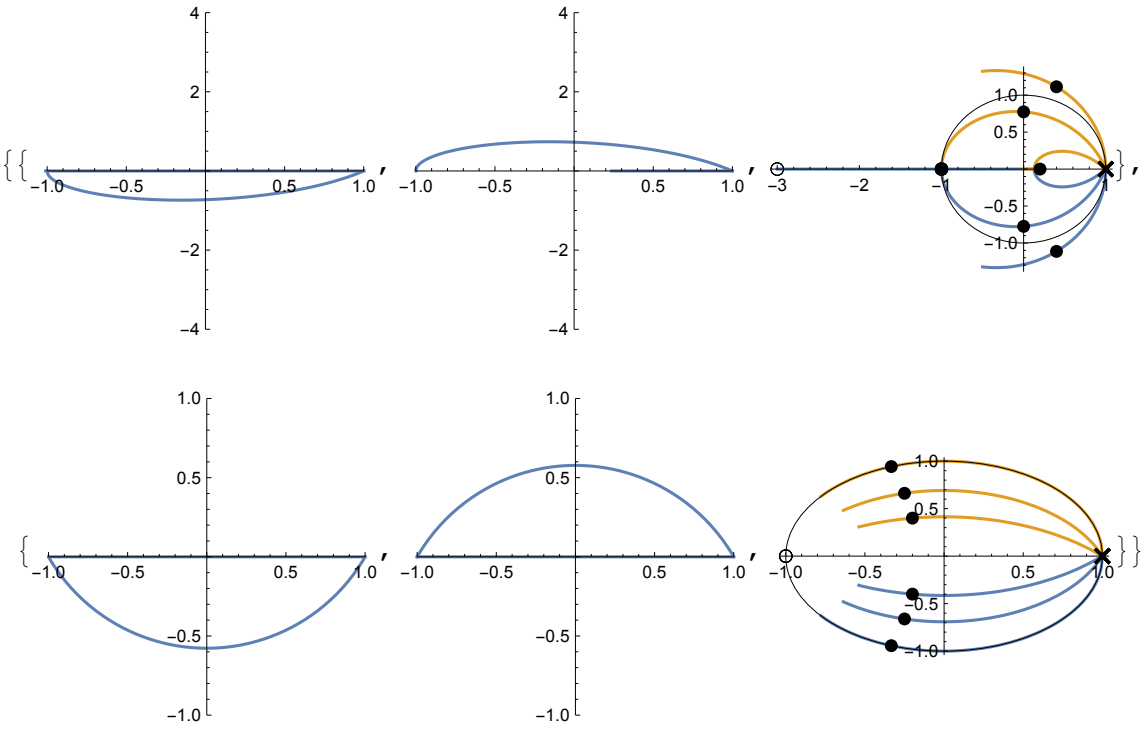
$$\frac{g2^2\ (1+z)^2}{(-1+z)^2+g2^2\ (1+z)^2+d2\ g2\ \left(-1+z^2\right)}$$

Plot of Root Locus to match roots

```
{tfe, sfe} = TransferFunctionAndPoles[svffe]
{ttr, str} = TransferFunctionAndPoles[svftr]
PlotTransferFunctionAndPoles[tfe, sfe, ttr, str]
```

$$\left\{\left(\frac{g1^2\ (1+z)^2}{2\left(2\ d1\ g1\ (-1+z)+2\ (-1+z)^2+g1^2\ (1+z)\right)}\right)_{\frac{1}{2}}^{\mathcal{T}},\right.$$
$$\left\{\frac{1}{4}\left(4-2\ d1\ g1-g1^2-g1\sqrt{-16+4\ d1^2+4\ d1\ g1+g1^2}\right),\right.$$
$$\left.\left.\frac{1}{4}\left(4-2\ d1\ g1-g1^2+g1\sqrt{-16+4\ d1^2+4\ d1\ g1+g1^2}\right)\right\}\right\}$$

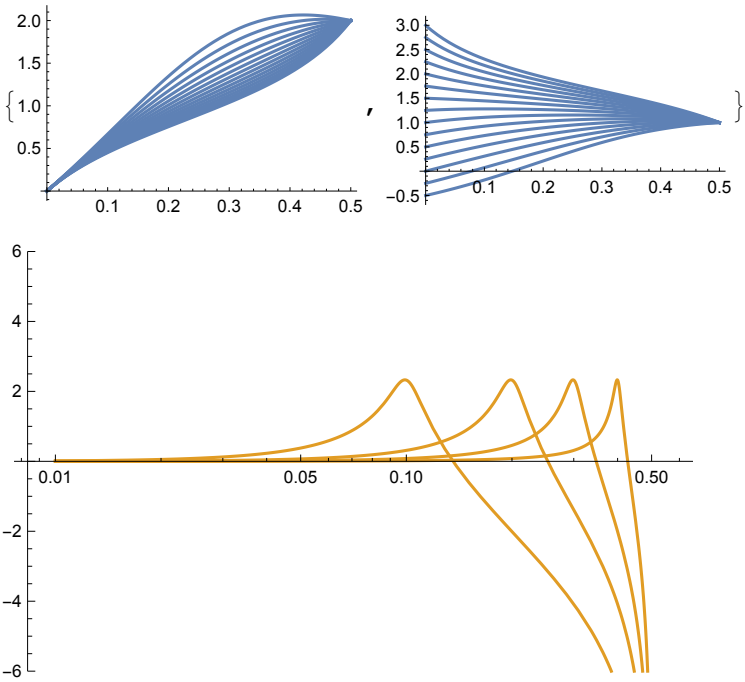
$$\left\{\left(\frac{g2^2\ (1+z)^2}{(-1+z)^2+g2^2\ (1+z)^2+d2\ g2\ \left(-1+z^2\right)}\right)_{\frac{1}{2}}^{\mathcal{T}},\right.$$
$$\left.\left\{-\frac{-1+g2^2+\sqrt{\left(-4+d2^2\right)\ g2^2}}{1+d2\ g2+g2^2},\ \frac{1-g2^2+\sqrt{\left(-4+d2^2\right)\ g2^2}}{1+d2\ g2+g2^2}\right\}\right\}$$



Solve to transform poles of FE to poles of TR

```
poleorderfe = {1, 2};
poles = MatchPoles[svffe, svftr, sfe, str, poleorderfe]
PlotPoleMapping[poles]
PlotResponses[svffe, svftr, poles, gain, 0.2]
```

$$\left\{g1 \to \frac{2\ g2}{\sqrt{1 + d2\ g2 + g2^2}},\ d1 \to \frac{d2 + g2}{\sqrt{1 + d2\ g2 + g2^2}}\right\}$$



Psuedo Code

```
init:
g2 = tan(pi*cutoff/samplerate)
d2 = 2 - 2*res
nrm = 1/sqrt (1 + d2*g2 + g2*g2)
g = 2*g2*nrm
d = (g2+d2)*nrm

clear:
v0z = 0;
v1 = 0;
v2 = 0;

tick:
v0 = input
v1z = v1
v2z = v2
v1 = v1z + g*(0.5*(v0 + v0z) - d*v1z - v2z)
v2 = v2z + g*(0.5*(v1 + v1z))
output = v2
v0z = v0
```

Low Pass, damping limit 2, averaging FIR on input and output

Note that this method is standard forward Euler without any tricks, all output states are computed from previous states only so they can be computed in parallel which is more efficient

Difference equations

```
svffeeqn = {v1 == v1z + g1 (1 / 2 (v0 + v0z) - d1 v1z - v2z),
    v2 == v2z + g1 (v1z), lp == (1 / 2 ( v2 + v2z)) / v0} /. sub
svffe = lp /. Solve[svffeeqn, {lp}, {v0, v1, v2}][[1]] // FullSimplify

svftreq = {vhp == v0 - d2 v1 - v2, 0 == g2 (vhp - 0) - (v1 - 0) + ic1eq,
     0 == g2 (v1) - (v2 - 0) + ic2eq, ic1eq == (2 (v1 - 0) - ic1eq) z^-1,
     ic2eq == (2 (v2 - 0) - ic2eq) z^-1, lp == (v2) / v0} /. sub;
svftr = lp /. Solve[svftreq, {lp}, {vhp, v0, v1, v2, ic1eq, ic2eq}][[1]] //
  FullSimplify
```

$$\left\{v1 == g1 \left(\frac{1}{2}\left(v0 + \frac{v0}{z}\right) - \frac{d1\ v1}{z} - \frac{v2}{z}\right) + \frac{v1}{z},\ v2 == \frac{g1\ v1}{z} + \frac{v2}{z},\ lp == \frac{v2 + \frac{v2}{z}}{2\ v0}\right\}$$

$$\frac{g1^2\ (1+z)^2}{4\left(g1^2 + d1\ g1\ (-1+z) + (-1+z)^2\right) z}$$

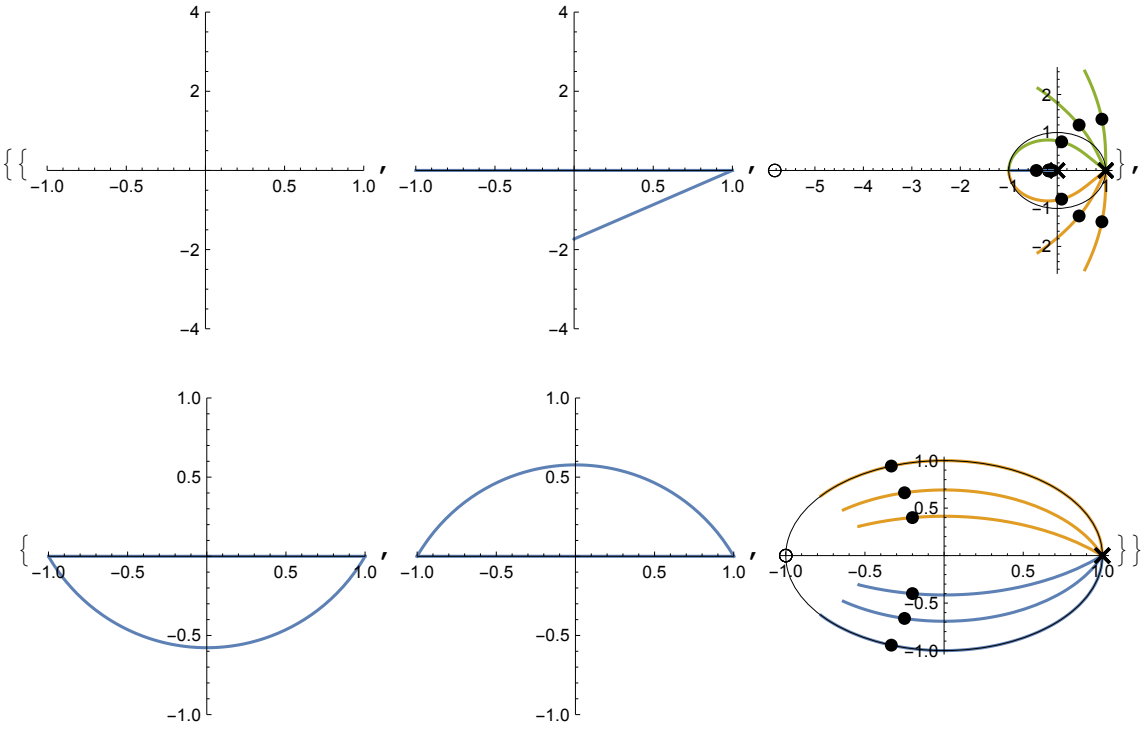
$$\frac{g2^2\ (1+z)^2}{(-1+z)^2 + g2^2\ (1+z)^2 + d2\ g2\left(-1+z^2\right)}$$

Plot of Root Locus to match roots

```
{tfe, sfe} = TransferFunctionAndPoles[svffe]
{ttr, str} = TransferFunctionAndPoles[svftr]
PlotTransferFunctionAndPoles[tfe, sfe, ttr, str]
```

$$\left\{\left(\frac{g1^2\ (1+z)^2}{4\ \left(g1^2+d1\ g1\ (-1+z)+(-1+z)^2\right)\ z}\right)^{\frac{1}{2}}\mathcal{T},\ \left\{0,\ \frac{1}{2}\left(2-d1\ g1-\sqrt{-4+d1^2}\ g1\right),\ \frac{1}{2}\left(2-d1\ g1+\sqrt{-4+d1^2}\ g1\right)\right\}\right\}$$

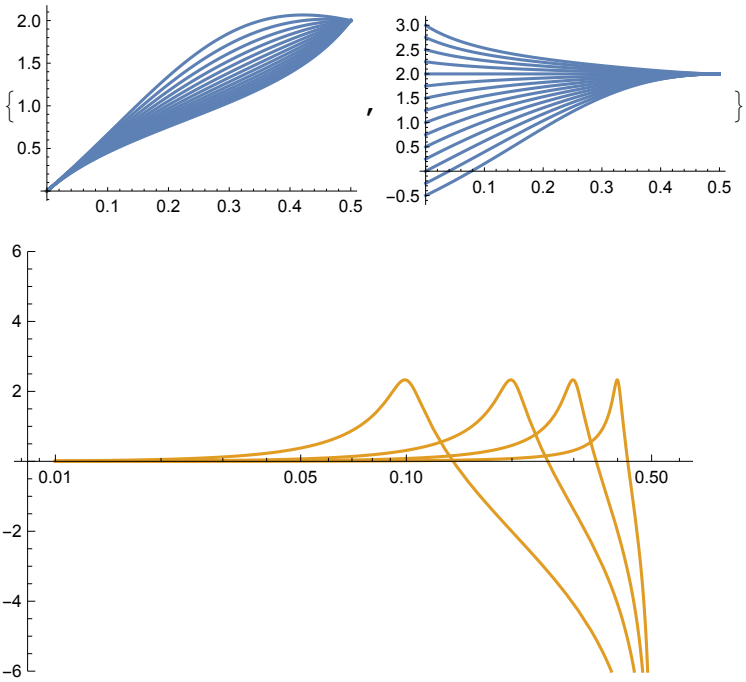
$$\left\{\left(\frac{g2^2\ (1+z)^2}{(-1+z)^2+g2^2\ (1+z)^2+d2\ g2\ \left(-1+z^2\right)}\right)^{\frac{1}{2}}\mathcal{T},\ \left\{-\frac{-1+g2^2+\sqrt{\left(-4+d2^2\right)\ g2^2}}{1+d2\ g2+g2^2},\ \frac{1-g2^2+\sqrt{\left(-4+d2^2\right)\ g2^2}}{1+d2\ g2+g2^2}\right\}\right\}$$



Solve to transform poles of FE to poles of TR

```
poleorderfe = {2, 3};
poles = MatchPoles[svffe, svftr, sfe, str, poleorderfe]
PlotPoleMapping[poles]
PlotResponses[svffe, svftr, poles, gain, 0.2]
```

$$\left\{g1 \to \frac{2\ g2}{\sqrt{1 + d2\ g2 + g2^2}},\ d1 \to \frac{d2 + 2\ g2}{\sqrt{1 + d2\ g2 + g2^2}}\right\}$$



Psuedo Code

```
init:
g2 = tan(pi*cutoff/samplerate)
d2 = 2 - 2*res
nrm = 1/sqrt (1 + d2*g2 + g2*g2)
g = 2*g2*nrm
d = g + d2*nrm

clear:
v0z = 0;
v1 = 0;
v2 = 0;

tick:
v0 = input
v1z = v1
v2z = v2
v1 = v1z + g*(0.5*(v0 + v0z) - d*v1z - v2z)
v2 = v2z + g*(v1z)
output = 0.5*(v2 + v2z)
v0z = v0
```

Mixing SVF Matching of FE to TR

Mixing SVF, damping limit 0, gain adjustment and averaging FIRs on outputs

This is the standard Chamberlin form, where the current times bandpass output is computed and then used to compute the current low pass output

Difference equations

```
svffeeqn = {vhp == v0 - d1 v1z - v2z, v1 == v1z + g1 (vhp), v2 == v2z + g1 (v1), hmix ==
      (m0 (vhp) + m1 (1 / 2 (v1 + v1z)) + m2 (1 / 4 v2 + 1 / 2 v2z + 1 / 4 v2zz)) / v0} /. sub;
svffemix[m0_, m1_, m2_] := Evaluate[
  hmix /. Solve[svffeeqn, {hmix}, {vhp, v0, v1, v2}][[1]] // FullSimplify]
svffelp = svffemix[0, 0, 1]
svffebp = svffemix[0, 1, 0]
svffehp = svffemix[1, 0, 0]

svftreq = {vhp == v0 - d2 v1 - v2, 0 == g2 (vhp - 0) - (v1 - 0) + ic1eq,
     0 == g2 (v1) - (v2 - 0) + ic2eq, ic1eq == (2 (v1 - 0) - ic1eq) z^-1,
     ic2eq == (2 (v2 - 0) - ic2eq) z^-1, hmix == (m0 vhp + m1 v1 + m2 v2) / v0} /. sub;
svftrmix[m0_, m1_, m2_] := Evaluate[
  hmix /. Solve[svftreq, {hmix}, {vhp, v0, v1, v2, ic1eq, ic2eq}][[1]] //
   FullSimplify]
svftrlp = svftrmix[0, 0, 1]
svftrbp = svftrmix[0, 1, 0]
svftrhp = svftrmix[1, 0, 0]
```

$$\frac{g1^2\,(1+z)^2}{4\left(1+d1\,g1\,(-1+z)+z\left(-2+g1^2+z\right)\right)}$$

$$\frac{g1\,(-1+z)\,(1+z)}{2\left(1+d1\,g1\,(-1+z)+z\left(-2+g1^2+z\right)\right)}$$

$$\frac{(-1+z)^2}{1+d1\,g1\,(-1+z)+z\left(-2+g1^2+z\right)}$$

$$\frac{g2^2\,(1+z)^2}{(-1+z)^2+g2^2\,(1+z)^2+d2\,g2\left(-1+z^2\right)}$$

$$\frac{g2\,(-1+z)\,(1+z)}{(-1+z)^2+g2^2\,(1+z)^2+d2\,g2\left(-1+z^2\right)}$$

$$\frac{(-1+z)^2}{(-1+z)^2+g2^2\,(1+z)^2+d2\,g2\left(-1+z^2\right)}$$

Plot of Root Locus to match roots

```
{tfelp, sfelp} = TransferFunctionAndPoles[svffelp]
{ttrlp, strlp} = TransferFunctionAndPoles[svftrlp]
PlotTransferFunctionAndPoles[tfelp, sfelp, ttrlp, strlp]
```

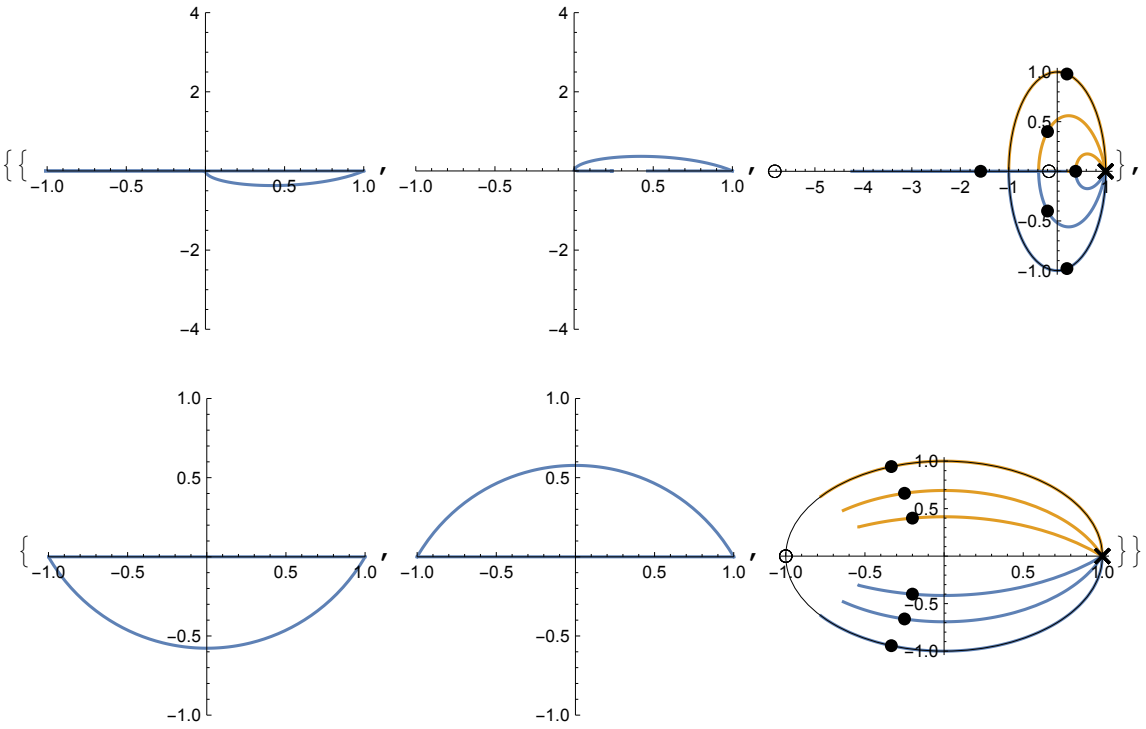
$$\left\{\left(\frac{g1^2\ (1+z)^2}{4\ \left(1+d1\ g1\ (-1+z)+z\ \left(-2+g1^2+z\right)\right)}\right)_{\frac{1}{2}}^{\mathcal{T}},\right.$$

$$\left\{\frac{1}{2}\left(2-d1\ g1-g1^2-g1\sqrt{-4+d1^2+2\ d1\ g1+g1^2}\right),\right.$$

$$\left.\left.\frac{1}{2}\left(2-d1\ g1-g1^2+g1\sqrt{-4+d1^2+2\ d1\ g1+g1^2}\right)\right\}\right\}$$

$$\left\{\left(\frac{g2^2\ (1+z)^2}{(-1+z)^2+g2^2\ (1+z)^2+d2\ g2\ \left(-1+z^2\right)}\right)_{\frac{1}{2}}^{\mathcal{T}},\right.$$

$$\left.\left\{-\frac{-1+g2^2+\sqrt{\left(-4+d2^2\right)\ g2^2}}{1+d2\ g2+g2^2},\ \frac{1-g2^2+\sqrt{\left(-4+d2^2\right)\ g2^2}}{1+d2\ g2+g2^2}\right\}\right\}$$



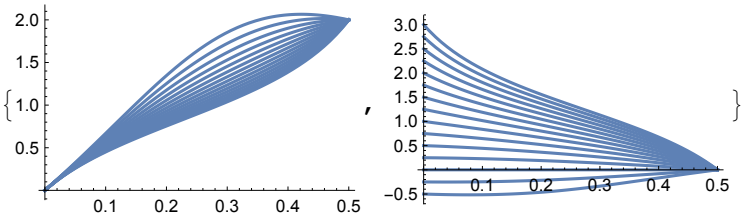
Solve to transform poles of FE to poles of TR

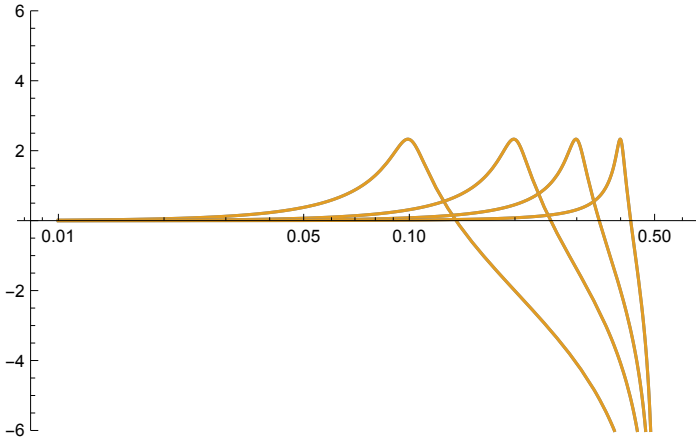
```
poleorderfelp = {1, 2};
poles = MatchPoles[svffelp, svftrlp, sfelp, strlp, poleorderfelp]
gainhp = MatchGain[svffehp, svftrhp, poles]
gainbp = MatchGain[svffebp, svftrbp, poles]
PlotPoleMapping[poles]
PlotResponses[svffemix[0, 0, 1], svftrmix[0, 0, 1], poles, 1, 0.2]
PlotResponses[svffemix[gainhp, 0, 0], svftrmix[1, 0, 0], poles, 1, 0.2]
PlotResponses[svffemix[0, gainbp, 0], svftrmix[0, 1, 0], poles, 1, 0.2]
PlotResponses[svffemix[gainhp, 0, 1], svftrmix[1, 0, 1], poles, 1, 0.2]
PlotResponses[svffemix[-gainhp, 0, 1], svftrmix[-1, 0, 1], poles, 1, 0.2]
```

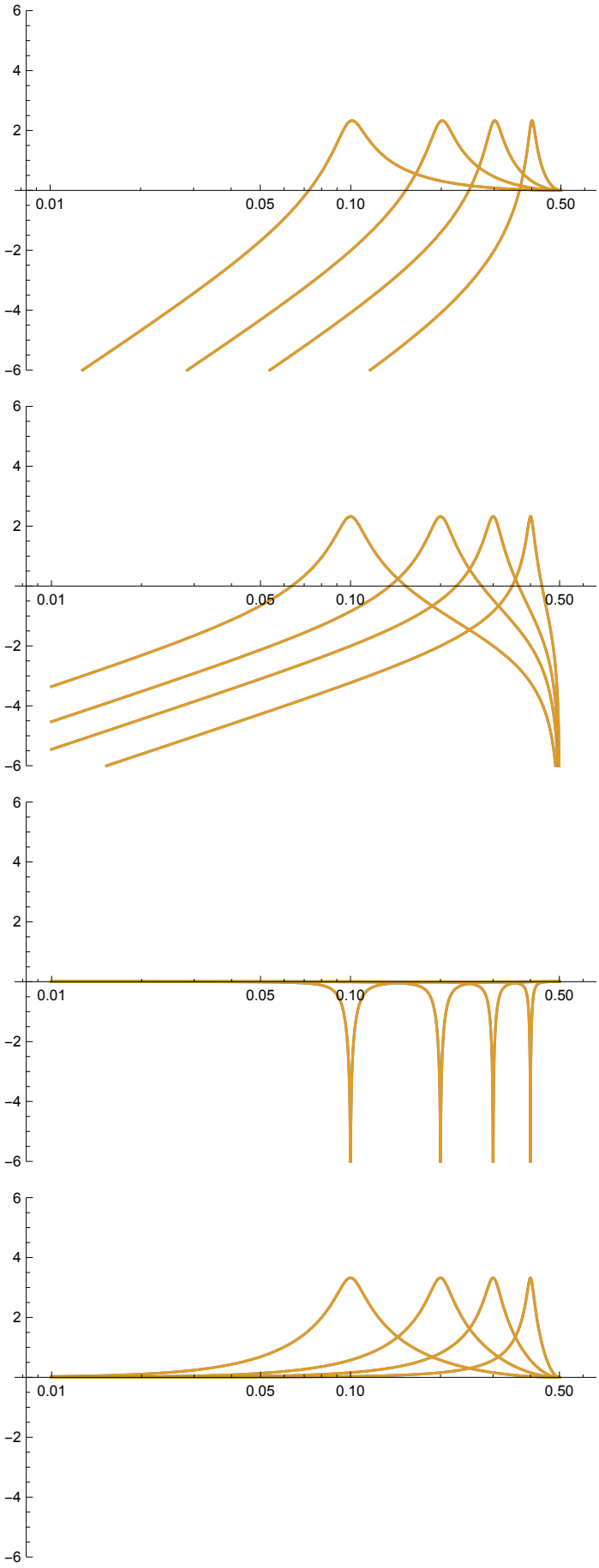
$$\left\{g1 \to \frac{2\ g2}{\sqrt{1 + d2\ g2 + g2^2}},\ d1 \to \frac{d2}{\sqrt{1 + d2\ g2 + g2^2}}\right\}$$

$$\frac{1}{1 + g2\ (d2 + g2)}$$

$$\frac{1}{\sqrt{1 + g2\ (d2 + g2)}}$$






6
4
2
−2
−4
−6
0.01
0.05
0.10
0.50

Psuedo Code

```
init:
g2 = tan(pi*cutoff/samplerate)
d2 = 2 - 2*res
nrm = 1/sqrt (1 + d2*g2 + g2*g2)
g = 2*g2*nrm
d = d2*nrm
m0 = high pass mix
m1 = band pass mix
m2 = low pass mix

clear:
v0z = 0;
v1 = 0;
v2z = 0;
v2 = 0;

tick:
v0 = input
v1z = v1
v2zz = v2z
v2z = v2
vhp = v0 - d*v1z - v2z
v1 = v1z + g*(vhp)
v2 = v2z + g*(v1)
low  = 0.25*(v2 + v2zz) + 0.5*v2z
band = nrm*0.5*(v1 + v1z)
high = nrm*nrm*vhp
output = m0*high + m1*band + m2*low
v0z = v0
```

Mixing SVF, damping limit 1, gain adjustment, averaging FIR low pass and inside structure on band pass

This seems the most natural form as you get one of the averaging FIRs right inside the difference equations, so when you add non-linearities it will sound a bit better as it is low passed before hitting the non-linearity. The core filter psuedo code is very symmetric in nature so is appealling

Difference equations

```
svffeeqn =
  {vhp == v0 - d1 v1z - v2z, v1 == v1z + g1 (vhp), v2 == v2z + g1 (1 / 2 (v1 + v1z)),
    hmix == (m0 (vhp) + m1 (1 / 2 (v1 + v1z)) + m2 (1 / 2 (v2 + v2z))) / v0} /. sub;
svffemix[m0_, m1_, m2_] := Evaluate[
  hmix /. Solve[svffeeqn, {hmix}, {vhp, v0, v1, v2}][[1]] // FullSimplify]
svffelp = svffemix[0, 0, 1]
svffebp = svffemix[0, 1, 0]
svffehp = svffemix[1, 0, 0]

svftreq = {vhp == v0 - d2 v1 - v2, 0 == g2 (vhp - 0) - (v1 - 0) + ic1eq,
    0 == g2 (v1) - (v2 - 0) + ic2eq, ic1eq == (2 (v1 - 0) - ic1eq) z^-1,
    ic2eq == (2 (v2 - 0) - ic2eq) z^-1, hmix == (m0 vhp + m1 v1 + m2 v2) / v0} /. sub;
svftrmix[m0_, m1_, m2_] := Evaluate[
  hmix /. Solve[svftreq, {hmix}, {vhp, v0, v1, v2, ic1eq, ic2eq}][[1]] //
   FullSimplify]
svftrlp = svftrmix[0, 0, 1]
svftrbp = svftrmix[0, 1, 0]
svftrhp = svftrmix[1, 0, 0]
```

$$\frac{g1^2\,(1+z)^2}{2\left(2\,d1\,g1\,(-1+z)+2\,(-1+z)^2+g1^2\,(1+z)\right)}$$

$$\frac{g1\,(-1+z)\,(1+z)}{2\,d1\,g1\,(-1+z)+2\,(-1+z)^2+g1^2\,(1+z)}$$

$$\frac{2\,(-1+z)^2}{2\,d1\,g1\,(-1+z)+2\,(-1+z)^2+g1^2\,(1+z)}$$

$$\frac{g2^2\,(1+z)^2}{(-1+z)^2+g2^2\,(1+z)^2+d2\,g2\left(-1+z^2\right)}$$

$$\frac{g2\,(-1+z)\,(1+z)}{(-1+z)^2+g2^2\,(1+z)^2+d2\,g2\left(-1+z^2\right)}$$

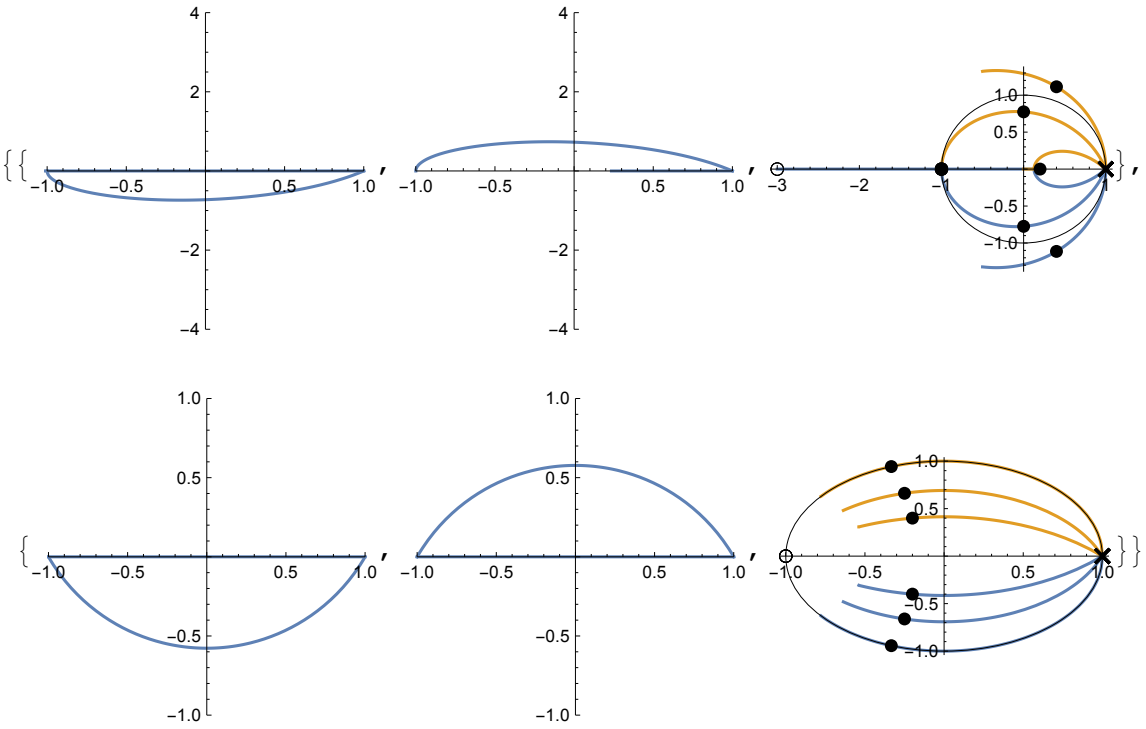
$$\frac{(-1+z)^2}{(-1+z)^2+g2^2\,(1+z)^2+d2\,g2\left(-1+z^2\right)}$$

Plot of Root Locus to match roots

```
{tfelp, sfelp} = TransferFunctionAndPoles[svffelp]
{ttrlp, strlp} = TransferFunctionAndPoles[svftrlp]
PlotTransferFunctionAndPoles[tfelp, sfelp, ttrlp, strlp]
```

$$\left\{\left(\frac{g1^2\ (1+z)^2}{2\left(2\,d1\,g1\ (-1+z)+2\ (-1+z)^2+g1^2\ (1+z)\right)}\right)_{\frac{1}{2}}^{\mathcal{T}},\right.$$
$$\left\{\frac{1}{4}\left(4-2\,d1\,g1-g1^2-g1\sqrt{-16+4\,d1^2+4\,d1\,g1+g1^2}\right),\right.$$
$$\left.\left.\frac{1}{4}\left(4-2\,d1\,g1-g1^2+g1\sqrt{-16+4\,d1^2+4\,d1\,g1+g1^2}\right)\right\}\right\}$$

$$\left\{\left(\frac{g2^2\ (1+z)^2}{(-1+z)^2+g2^2\ (1+z)^2+d2\,g2\left(-1+z^2\right)}\right)_{\frac{1}{2}}^{\mathcal{T}},\right.$$
$$\left.\left\{-\frac{-1+g2^2+\sqrt{\left(-4+d2^2\right)g2^2}}{1+d2\,g2+g2^2},\ \frac{1-g2^2+\sqrt{\left(-4+d2^2\right)g2^2}}{1+d2\,g2+g2^2}\right\}\right\}$$



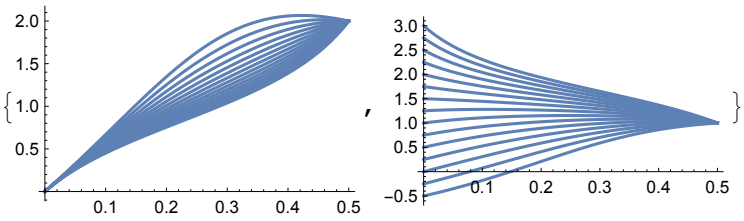
Solve to transform poles of FE to poles of TR

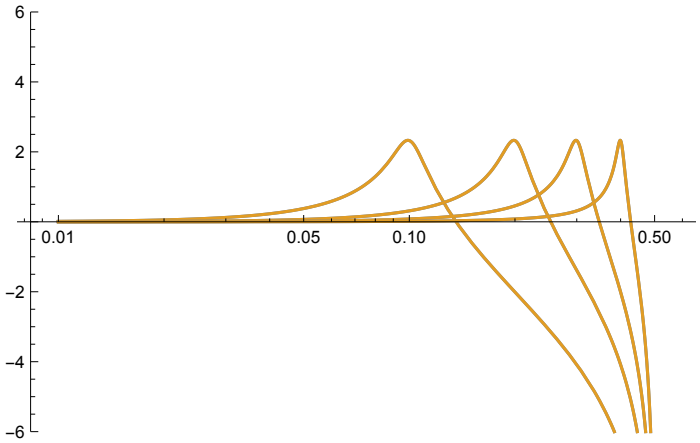
```
poleorderfelp = {1, 2};
poles = MatchPoles[svffelp, svftrlp, sfelp, strlp, poleorderfelp]
gainhp = MatchGain[svffehp, svftrhp, poles]
gainbp = Sqrt[gainhp]
PlotPoleMapping[poles]
PlotResponses[svffemix[0, 0, 1], svftrmix[0, 0, 1], poles, 1, 0.2]
PlotResponses[svffemix[gainhp, 0, 0], svftrmix[1, 0, 0], poles, 1, 0.2]
PlotResponses[svffemix[0, gainbp, 0], svftrmix[0, 1, 0], poles, 1, 0.2]
PlotResponses[svffemix[gainhp, 0, 1], svftrmix[1, 0, 1], poles, 1, 0.2]
PlotResponses[svffemix[-gainhp, 0, 1], svftrmix[-1, 0, 1], poles, 1, 0.2]
```

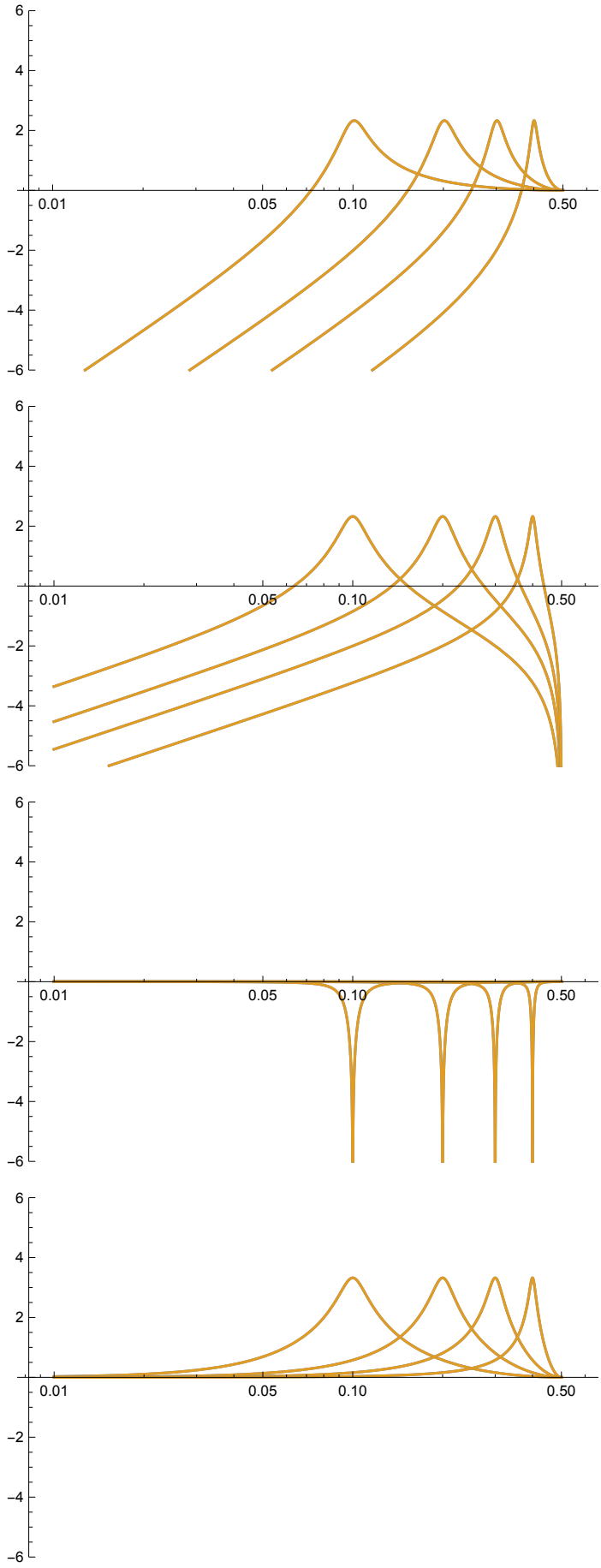
$$\left\{g1 \to \frac{2\ g2}{\sqrt{1 + d2\ g2 + g2^2}},\ d1 \to \frac{d2 + g2}{\sqrt{1 + d2\ g2 + g2^2}}\right\}$$

$$\frac{1}{1 + g2\ (d2 + g2)}$$

$$\sqrt{\frac{1}{1 + g2\ (d2 + g2)}}$$






6
4
2
−2
−4
−6
0.01
0.05
0.10
0.50

Psuedo Code

```
init:
g2 = tan(pi*cutoff/samplerate)
d2 = 2 - 2*res
nrm = 1/sqrt (1 + d2*g2 + g2*g2)
g = 2*g2*nrm
d = (g2+d2)*nrm
m0 = high pass mix
m1 = band pass mix
m2 = low pass mix

clear:
v0z = 0;
v1 = 0;
v2z = 0;
v2 = 0;

tick:
v0 = input
v1z = v1
v2z = v2
vhp = v0 - d*v1z - v2z
v1 = v1z + g*(vhp)
vbp = 0.5*(v1 + v1z)
v2 = v2z + g*(vbp)
vlp = 0.5*(v2 + v2z)
low = vlp
band = nrm*vbp
high = nrm*nrm*vhp
output = m0*high + m1*band + m2*low
v0z = v0
```

Mixing SVF, damping limit 2, gain adjustment and averaging FIRs on outputs

Probably not that useful as the mixing version requires extra delays to line up the high and band-pass outputs with the low pass, so you consume a bit more cpu, but still you get the benefit of more parallel execution

Difference equations

```
svffeeqn = {vhp == v0 - d1 v1z - v2z, v1 == v1z + g1 (vhp),
    v2 == v2z + g1 (v1z), hmix == (m0 (vhp z^-1) + m1 (1 / 2 (v1 + v1z) z^-1) +
        m2 (1 / 4 v2 + 1 / 2 v2z + 1 / 4 v2zz)) / v0} /. sub;
svffemix[m0_, m1_, m2_] := Evaluate[hmix /.
    Solve[svffeeqn, {hmix}, {vhp, v0, v1, v2}][[1]] // FullSimplify]
svffelp = svffemix[0, 0, 1]
svffebp = svffemix[0, 1, 0]
svffehp = svffemix[1, 0, 0]

svftreq = {vhp == v0 - d2 v1 - v2, 0 == g2 (vhp - 0) - (v1 - 0) + ic1eq,
    0 == g2 (v1) - (v2 - 0) + ic2eq, ic1eq == (2 (v1 - 0) - ic1eq) z^-1,
    ic2eq == (2 (v2 - 0) - ic2eq) z^-1, hmix == (m0 vhp + m1 v1 + m2 v2) / v0} /. sub;
svftrmix[m0_, m1_, m2_] := Evaluate[
  hmix /. Solve[svftreq, {hmix}, {vhp, v0, v1, v2, ic1eq, ic2eq}][[1]] //
   FullSimplify]
svftrlp = svftrmix[0, 0, 1]
svftrbp = svftrmix[0, 1, 0]
svftrhp = svftrmix[1, 0, 0]
```

$$\frac{g1^2\ (1+z)^2}{4\left(g1^2+d1\ g1\ (-1+z)+(-1+z)^2\right)\ z}$$

$$\frac{g1\ (-1+z)\ (1+z)}{2\left(g1^2+d1\ g1\ (-1+z)+(-1+z)^2\right)\ z}$$

$$\frac{(-1+z)^2}{\left(g1^2+d1\ g1\ (-1+z)+(-1+z)^2\right)\ z}$$

$$\frac{g2^2\ (1+z)^2}{(-1+z)^2+g2^2\ (1+z)^2+d2\ g2\ \left(-1+z^2\right)}$$

$$\frac{g2\ (-1+z)\ (1+z)}{(-1+z)^2+g2^2\ (1+z)^2+d2\ g2\ \left(-1+z^2\right)}$$

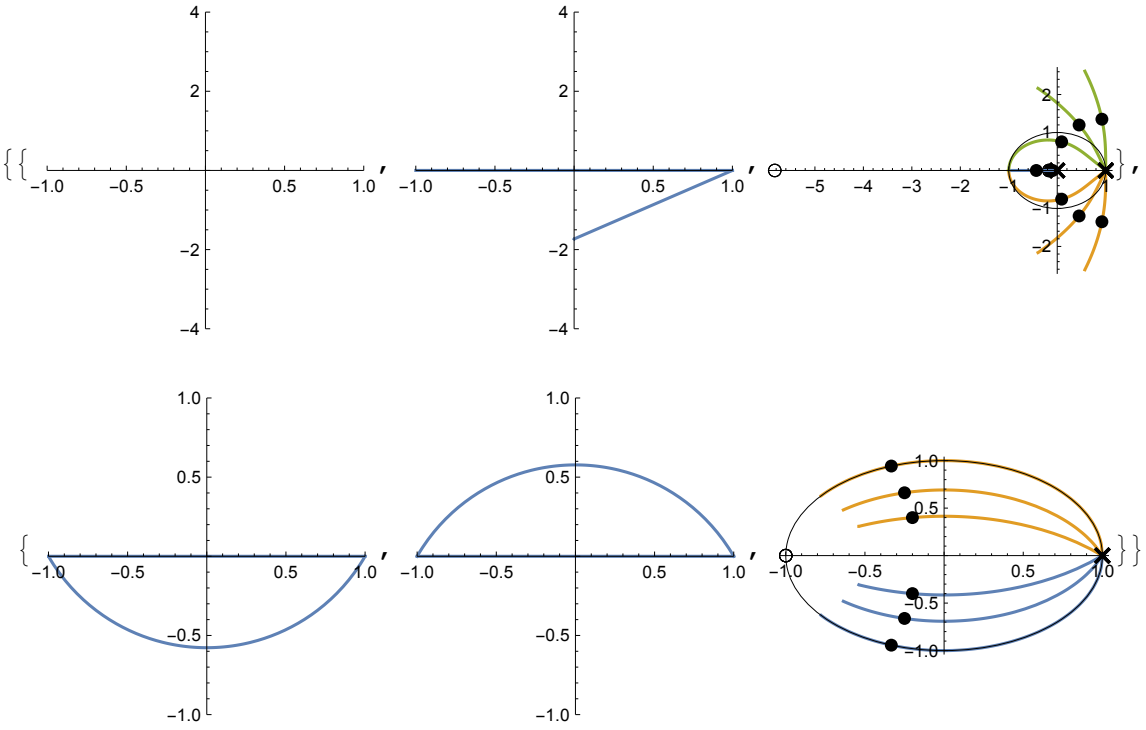
$$\frac{(-1+z)^2}{(-1+z)^2+g2^2\ (1+z)^2+d2\ g2\ \left(-1+z^2\right)}$$

Plot of Root Locus to match roots

```
{tfelp, sfelp} = TransferFunctionAndPoles[svffelp]
{ttrlp, strlp} = TransferFunctionAndPoles[svftrlp]
PlotTransferFunctionAndPoles[tfelp, sfelp, ttrlp, strlp]
```

$$\left\{\left(\frac{g1^2\ (1+z)^2}{4\ \left(g1^2+d1\ g1\ (-1+z)+(-1+z)^2\right)\ z}\right)^{\mathcal{T}}_{\frac{1}{2}},\ \left\{0,\ \frac{1}{2}\left(2-d1\ g1-\sqrt{-4+d1^2}\ g1\right),\ \frac{1}{2}\left(2-d1\ g1+\sqrt{-4+d1^2}\ g1\right)\right\}\right\}$$

$$\left\{\left(\frac{g2^2\ (1+z)^2}{(-1+z)^2+g2^2\ (1+z)^2+d2\ g2\ \left(-1+z^2\right)}\right)^{\mathcal{T}}_{\frac{1}{2}},\ \left\{-\frac{-1+g2^2+\sqrt{\left(-4+d2^2\right)\ g2^2}}{1+d2\ g2+g2^2},\ \frac{1-g2^2+\sqrt{\left(-4+d2^2\right)\ g2^2}}{1+d2\ g2+g2^2}\right\}\right\}$$



Solve to transform poles of FE to poles of TR

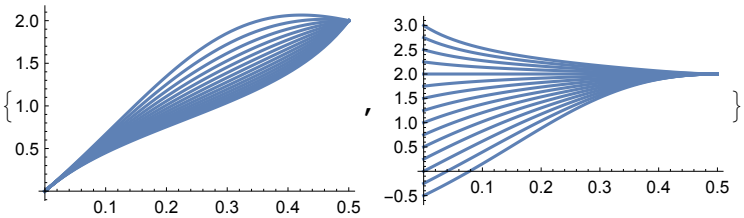
```
poleorderfelp = {2, 3};
poles = MatchPoles[svffelp, svftrlp, sfelp, strlp, poleorderfelp]
gainlp = MatchGain[svffehp, svftrhp, poles]
gainhp = MatchGain[svffehp, svftrhp, poles] / g2
gainbp = MatchGain[svffebp, svftrbp, poles] / g2
PlotPoleMapping[poles]
PlotResponses[svffemix[0, 0, 1], svftrmix[0, 0, 1], poles, 1, 0.2]
PlotResponses[svffemix[gainhp, 0, 0], svftrmix[1, 0, 0], poles, 1, 0.2]
PlotResponses[svffemix[0, gainbp, 0], svftrmix[0, 1, 0], poles, 1, 0.2]
PlotResponses[svffemix[gainhp, 0, 1], svftrmix[1, 0, 1], poles, 1, 0.2]
PlotResponses[svffemix[-gainhp, 0, 1], svftrmix[-1, 0, 1], poles, 1, 0.2]
```

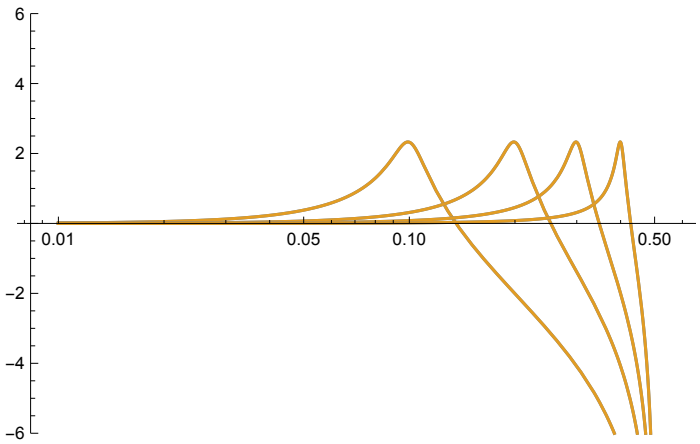
$$\left\{g1 \to \frac{2\ g2}{\sqrt{1 + d2\ g2 + g2^2}},\ d1 \to \frac{d2 + 2\ g2}{\sqrt{1 + d2\ g2 + g2^2}}\right\}$$

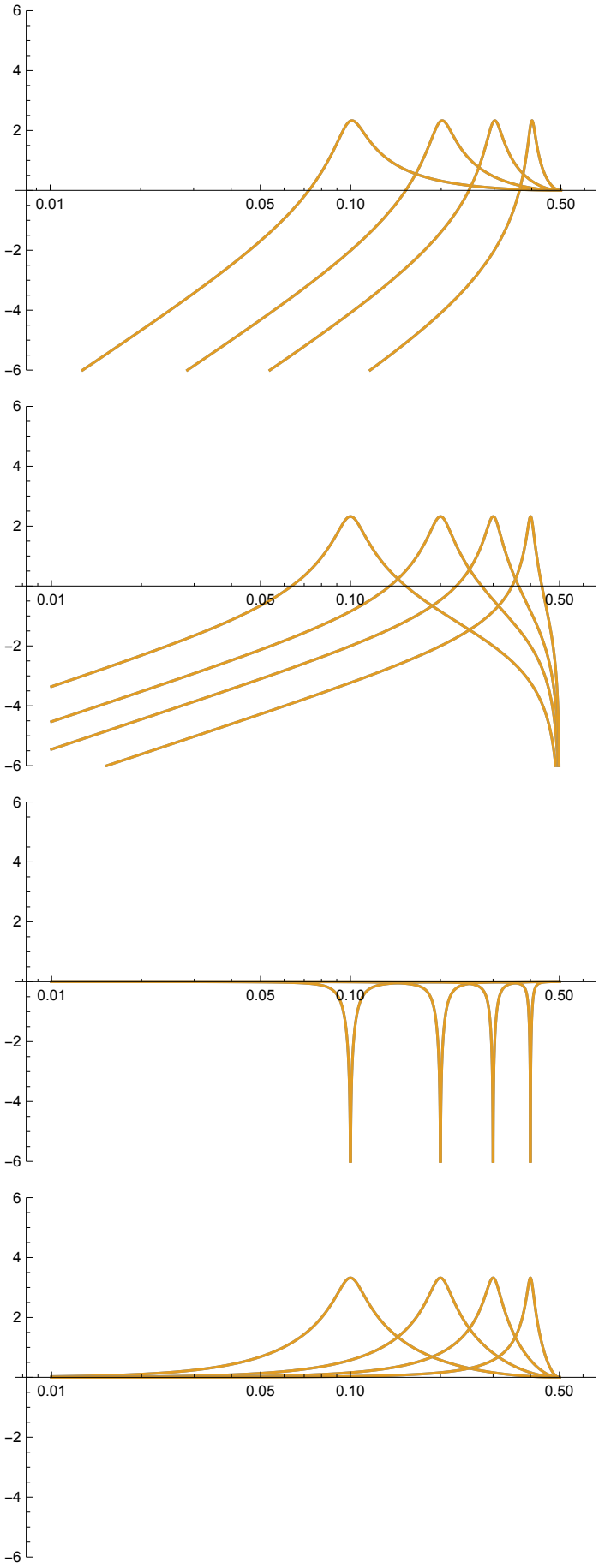
$$\frac{g2}{1 + g2\ (d2 + g2)}$$

$$\frac{1}{1 + g2\ (d2 + g2)}$$

$$\frac{1}{\sqrt{1 + g2\ (d2 + g2)}}$$






6
4
2
−2
−4
−6
0.01
0.05
0.10
0.50

Psuedo Code

```
init:
g2 = tan(pi*cutoff/samplerate)
d2 = 2 - 2*res
nrm = 1/sqrt (1 + d2*g2 + g2*g2)
g = 2*g2*nrm
d = g + d2*nrm
m0 = high pass mix
m1 = band pass mix
m2 = low pass mix

clear:
v0z = 0;
vhp = 0;
v1z = 0;
v1 = 0;
v2z = 0;
v2 = 0;

tick:
v0 = input
vhpz = vhp
v1zz = v1z
v1z = v1
v2zz = v2z
v2z = v2
vhp = v0 - d*v1z - v2z
v1 = v1z + g*(vhp)
v2 = v2z + g*(v1z)
low  = 0.25*(v2 + v2zz) + 0.5*v2z
band = nrm*0.5*(v1z + v1zz)
high = nrm*nrm*vhpz
output = m0*high + m1*band + m2*low
v0z = v0
```