APPENDIX A

EXAMPLE *MATHEMATICA* NOTEBOOKS

Filter Design

For Signal Processing

Using MATLAB and *Mathematica*

Miroslav D. Lutovac, Ph. D.
Dejan V. Tošić, Ph. D.
Brian L. Evans, Ph. D.

A.1 Analysis by Transform Method of Analog LTI Circuits

Miroslav D. Lutovac, Dejan V. Tosic and Brian L. Evans
lutovac@iritel.bg.ac.yu tosic@galeb.etf.bg.ac.yu bevans@ece.utexas.edu

A.1.1 References

1. G. S. Moschytz and P. Horn,
Active Filter Design Handbook
For Use with Programmable Pocket Calculators and Mini Computers.
John Wiley and Sons, Ltd., New York, 1981, pp. 38–39

2. R. P. Sallen and E. L. Key,
"A practical method of designing RC active filters,"
IRE Trans. Circuit Theory, CT–2, Mar. 1955, pp. 74–85.

3. D.V. Tosic, M.D. Lutovac, B.L. Evans, I.M. Markoski,
"A tool for symbolic analysis and design of analog active filters,"
5th Int. Workshop, Symbolic Methods, Applications, Circuit Design,
SMACD'98, Kaiserslautern, Germany, pp. 71–74, Oct. 1998.

4. D. V. Tosic, M. D. Lutovac,
"Symbolic computation of impulse, step and sine responses of linear time-invariant systems,"
Proc. 10th Int. Symp. Theor. Electrical Engineering,
ISTET99, Magdeburg, Germany, Sep. 1999, pp. 653–657.

5. D. V. Tosic,
"SALECAS - a package for symbolic analysis of linear circuits and systems,"
4th Int. Workshop, Symbolic Methods, Applications, Circuit Design,
SMACD, Leuven, Belgium, pp. 227–230, Oct. 1996.

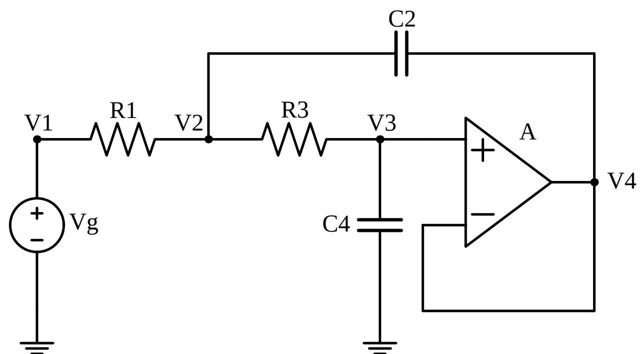
6. M. D. Lutovac, D. V. Tosic and B. L. Evans,
"Advanced Filter Design for Signal Processing using MATLAB and Mathematica,"
http://iritel.iritel.bg.ac.yu/˜lutovac/www/afdhome.htm
http://galeb.etf.bg.ac.yu/˜tosic/afdhome.htm
http://www.ece.utexas.edu/˜bevans/

A.1.2 Initialization

```
SetDirectory[HomeDirectory[]];
<<Calculus`LaplaceTransform`
<<afd\math\m\clearall.m
<<afd\math\m\drawafil.m
<<afd\math\m\drawacc.m
```

A.1.3 Schematic

```
DrawLPSK[0, 0, 1/2, 1.25, 8];
```



A.1.4 Circuit Analysis

Reduced Modified Nodal Analysis (RMNA)

```
CircuitEquations = {
 V1 == Vg
,(V2-V1)/R1 + (V2-V3)/R3 + (V2-V4)*(s*C2) == 0
,(V3-V2)/R3 + V3*(s*C4) == 0
,V3 == V4
};
NodeVoltages = {V1,V2,V3,V4};
CircuitResponse = Together[Flatten[
 Solve[CircuitEquations,NodeVoltages]
]];
```

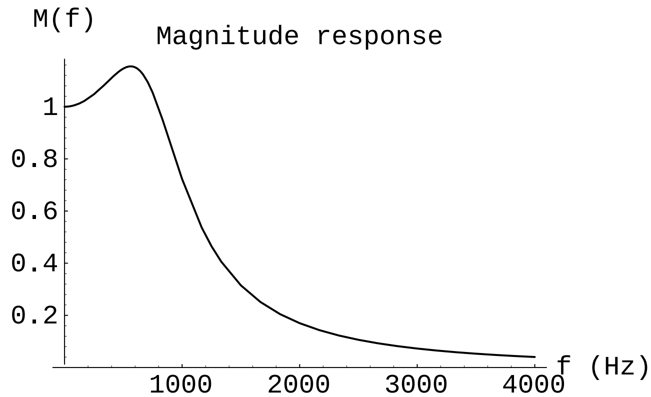
A.1.5 Response

```
V1s = Simplify[V1 /. CircuitResponse];
Collect[Numerator[%],s]/Collect[Denominator[%],s]

Vg

V2s = Simplify[V2 /. CircuitResponse];
Collect[Numerator[%],Vg]/Collect[Denominator[%],s]

           (1 + C4 R3 s) Vg
---------------------------------------
                                      2
1 + (C4 R1 + C4 R3) s + C2 C4 R1 R3 s
```

```
V3s = Simplify[V3 /. CircuitResponse];
Collect[Numerator[%],s]/Collect[Denominator[%],s]

                  Vg
--------------------------------------
                                     2
1 + (C4 R1 + C4 R3) s + C2 C4 R1 R3 s

V4s = Simplify[V4 /. CircuitResponse];
Collect[Numerator[%],s]/Collect[Denominator[%],s]

                  Vg
--------------------------------------
                                     2
1 + (C4 R1 + C4 R3) s + C2 C4 R1 R3 s
```

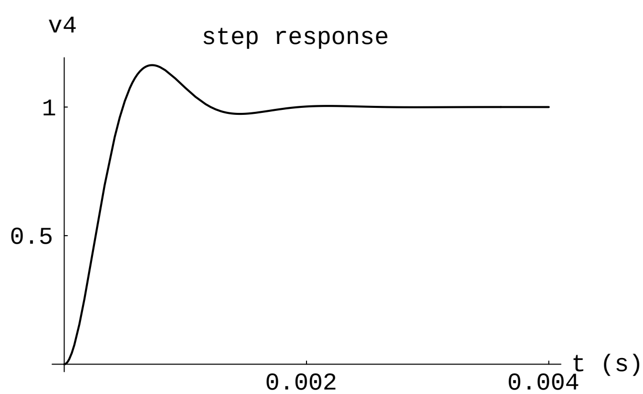
■ A.1.6 Voltage Transfer Function

```
H = V4s/Vg;
Collect[Numerator[%],s]/Collect[Denominator[%],s]

                  1
--------------------------------------
                                     2
1 + (C4 R1 + C4 R3) s + C2 C4 R1 R3 s

M = Abs[H] /. {C2->4*C, C4->C, R1->R, R3->R} ./ s -> I*2*Pi*f

                  1
Abs[---------------------------------]
                            2 2  2 2
     1 + 4 * C f Pi R - 16 C F Pi R

Plot[ {M ./ {C->10^(-8), R->10^4}}, {f,0,4000}
, PlotRange -> All
, AxesLabel -> {"f (Hz)", "M(f)"}
, AxesOrigin -> {0,0}
, PlotLabel -> "Magnitude response"];
```



■ A.1.7 Step Response

```
v4t = InverseLaplaceTransform[
H/s //. {C2->4*C, C4->C, R1->R, R3->R}
, s, t];
v4t //Together //Simplify

        Sqrt[3] t          Sqrt[3] t
    Cos[---------]     Sin[---------]
          4 C R              4 C R
1 - -------------- - ------------------
       t/(4 C R)             t/(4 C R)
      E              Sqrt[3] E

Plot[v4t /. {C->10^(-8), R->10^4}
, {t,0,4*10^(-3)}
, PlotRange -> All
, AxesLabel -> {"t (s)", "v4"}
, Ticks -> {{0,0.002,0.004},{0,0.5,1}}
, PlotLabel -> "step response"];
```



A.2 Analysis by Transform Method of Discrete-Time LTI Systems

Miroslav D. Lutovac, Dejan V. Tosic and Brian L. Evans
lutovac@iritel.bg.ac.yu tosic@galeb.etf.bg.ac.yu bevans@ece.utexas.edu

A.2.1 References

1. Alan Oppenheim, Ronald Schafer, "Digital Signal Processing," Prentice-Hall, Englewood Cliffs, New Jersey, 1975.

2.Sanjit Mitra, James Kaiser, "Handbook for Digital Signal Processing," John Wiley, New York, 1993, pp. 127–128.

3. D. V. Tosic, M. D. Lutovac, "Symbolic computation of impulse, step and sine responses of linear time-invariant systems," Proc. 10th Int. Symp. Theor. Electrical Engineering, ISTET99, Magdeburg, Germany, Sep. 1999, pp. 653–657.

4. John M. Novak and Brian L. Evans, "Mathematica, Signals and Systems," Georgia Tech Research Corp., Atlanta, Georgia, 1995.

5. D. V. Tosic, M. D. Lutovac, and I. M. Markoski, "Symbolic derivation of transfer functions of discrete-time systems," ISTET'97, Palermo, Italia, 9–11 June 1997, pp. 311–314.

6. D. V. Tosic, "SALECAS - a package for symbolic analysis of linear circuits and systems," SMACD, Leuven, Belgium, pp. 227–230, Oct. 1996.

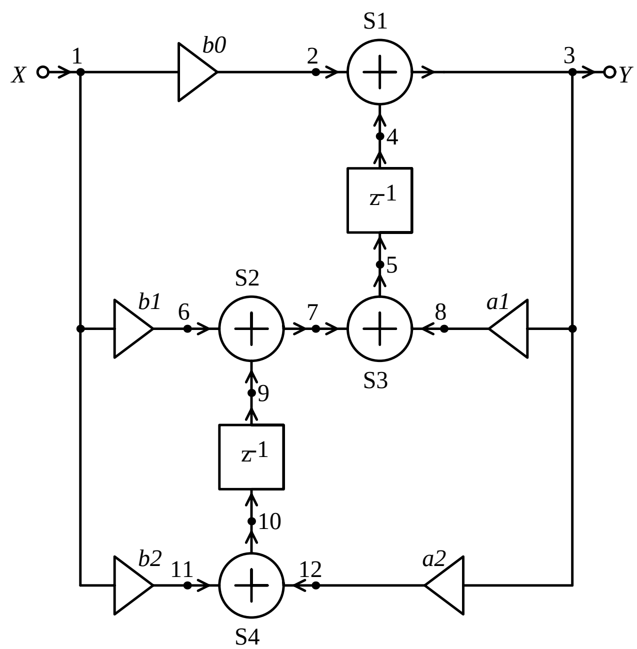
7. M. D. Lutovac, D. V. Tosic and B. L. Evans, "Advanced Filter Design for Signal Processing using MATLAB and Mathematica,"
http://iritel.iritel.bg.ac.yu/~lutovac/www/afdhome.htm
http://galeb.etf.bg.ac.yu/~tosic/afdhome.htm
http://www.ece.utexas.edu/~bevans/

A.2.2 Initialization

```
SetDirectory[HomeDirectory[]];
<<afd\math\m\clearall.m
AppendTo[$Path, "APPS"];"
Needs["SignalProcessing`Master`"]
<<afd\math\m\drawdfil.m
<<afd\math\m\drawiirf.m
```

A.2.3 Block Diagram

```
DrawTDF2[0,0,1,1/0.8,10];
```



A.2.4 Analysis (v=1/z)

```
ElementEquations = {
  Y1 == X
, Y2 == b0*Y1 + Xb0
, Y3 == Y2 + Y4
, Y4 == v*Y5
, Y5 == Y7 + Y8
, Y6 == b1*Y1 + Xb1
, Y7 == Y6 + Y9
, Y8 == a1*Y3 + Xa1
, Y9 ==  v*Y10
, Y10 == Y11 + Y12
, Y11 == b2*Y1 + Xb2
, Y12 == a2*Y3 + Xa2
};
```

```
NodeSignals = {Y1,Y2,Y3,Y4,Y5,Y6,Y7,Y8,Y9,Y10,Y11,Y12};
Response = Flatten[Solve[ElementEquations,NodeSignals]];
Y = Together[Y3/.Response]

                        2              2                       2
-(b0 X) - b1 v X - b2 v  X - v Xa1 - v  Xa2 - Xb0 - v Xb1 - v  Xb2
------------------------------------------------------------------
                                        2
                        -1 + a1 v + a2 v
```

■ A.2.5 Transfer Function and Noise Transfer Functions

```
Hv   = Y /. {X -> 1, Xa1 -> 0, Xa2 -> 0, Xb0 -> 0, Xb1 -> 0, Xb2 -> 0};
Ha1v = Y /. {X -> 0, Xa1 -> 1, Xa2 -> 0, Xb0 -> 0, Xb1 -> 0, Xb2 -> 0};
Ha2v = Y /. {X -> 0, Xa1 -> 0, Xa2 -> 1, Xb0 -> 0, Xb1 -> 0, Xb2 -> 0};
Hb0v = Y /. {X -> 0, Xa1 -> 0, Xa2 -> 0, Xb0 -> 1, Xb1 -> 0, Xb2 -> 0};
Hb1v = Y /. {X -> 0, Xa1 -> 0, Xa2 -> 0, Xb0 -> 0, Xb1 -> 1, Xb2 -> 0};
Hb2v = Y /. {X -> 0, Xa1 -> 0, Xa2 -> 0, Xb0 -> 0, Xb1 -> 0, Xb2 -> 1};
  H = Collect[-Numerator[Hv],v] / (-Collect[Denominator[Hv],v]);
Ha1 = Collect[-Numerator[Ha1v],v]/(-Collect[Denominator[Ha1v],v]);
Ha2 = Collect[-Numerator[Ha2v],v]/(-Collect[Denominator[Ha2v],v]);
Hb0 = Collect[-Numerator[Hb0v],v]/(-Collect[Denominator[Hb0v],v]);
Hb1 = Collect[-Numerator[Hb1v],v]/(-Collect[Denominator[Hb1v],v]);
Hb2 = Collect[-Numerator[Hb2v],v]/(-Collect[Denominator[Hb2v],v]);
v2invz = {v->z^"-1", v^2->z^("-2")};
Print["H(z) = ", H /. v2invz ]
Print["Ha1(z) = ", Ha1 /. v2invz ]
Print["Ha2(z) = ", Ha2 /. v2invz ]
Print["Hb0(z) = ", Hb0 /. v2invz ]
Print["Hb1(z) = ", Hb1 /. v2invz ]
Print["Hb2(z) = ", Hb2 /. v2invz ]

                -1       -2
       b0 + b1 z   + b2 z
H(z) = --------------------
               -1       -2
       1 - a1 z   - a2 z
                  -1
                 z
Ha1(z) = -------------------
                 -1       -2
         1 - a1 z   - a2 z
                  -2
                 z
Ha2(z) = -------------------
                 -1       -2
         1 - a1 z   - a2 z
```

```
                 1
Hb0(z) = -------------------
                -1       -2
         1 - a1 z   - a2 z
                 -1
                z
Hb1(z) = -------------------
                -1       -2
         1 - a1 z   - a2 z
                 -2
                z
Hb2(z) = -------------------
                -1       -2
         1 - a1 z   - a2 z
```

■ A.2.6 Complex Response in Terms of v=z^(-1)

```
Y3z = Collect[-Numerator[Y],X]/(-Denominator[Y])
```

```
                 2                2                       2
(b0 + b1 v + b2 v ) X + v Xa1 + v  Xa2 + Xb0 + v Xb1 + v  Xb2
-------------------------------------------------------------
                                    2
                      1 - a1 v - a2 v
```

■ A.2.7 Transfer Function

```
H3z = H /. {b0->1, b1->2, b2->1, a1->0, a2->-1/2, v->z^(-1)}
```

```
     -2   2
1 + z   + -
          z
-----------
      1
 1 + ----
        2
     2 z
```

■ A.2.8 Impulse Response

```
y3n = InverseZTransform[H3z,z,n]
```

```
     -I    n
-((-------)  (1 - 2 I Sqrt[2]) DiscreteStep[n])
   Sqrt[2]
----------------------------------------------- -
                      2
     I    n
 (-------)  (1 + 2 I Sqrt[2]) DiscreteStep[n]
  Sqrt[2]
 -------------------------------------------- + 2 KroneckerDelta[n]
                     2
```

```
y3nDS = ComplexExpand[Coefficient[y3n,DiscreteStep[n]]]
```

```
      n Pi
  Cos[----]
       2          1/2 - n/2     n Pi
-(---------) + 2 2           Sin[----]
     n/2                          2
    2
```

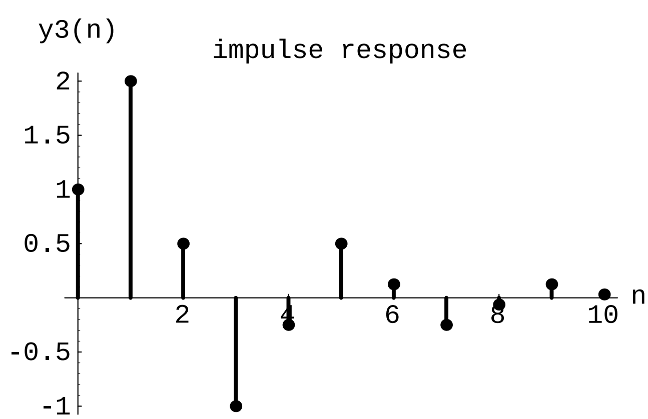
```
y3nKD = Coefficient[y3n,KroneckerDelta[n]]
```

```
2
```

```
h3n = y3nDS*DiscreteStep[n] + y3nKD*KroneckerDelta[n]
```

```
                                              n Pi
                                          Cos[----]
                                               2          1/2 - n/2     n Pi
2 KroneckerDelta[n] + DiscreteStep[n] (-(---------) + 2 2           Sin[----])
                                              n/2                          2
                                             2
```

```
DiscreteSignalPlot[h3n
,{n,0,10}
,AxesLabel -> {"n","y3(n)"}
,PlotLabel -> "impulse response"
];
```



```
Table[{n,h3n}, {n,0,10}]
% //TableForm //N
```

```
                     1                  1        1       1         1
{{0, 1}, {1, 2}, {2, -}, {3, -1}, {4, -(-)}, {5, -}, {6, -}, {7, -(-)},
                     2                  4        2       8         4

      1         1        1
{8, -(--)}, {9, -}, {10, --}}
      16        8        32
```

```
0     1.
1.    2.
2.    0.5
3.    -1.
4.    -0.25
5.    0.5
6.    0.125
7.    -0.25
8.    -0.0625
9.    0.125
10.   0.03125
```

A.3 Switched Capacitor Filter - Mode 3a
Analysis and Design

Miroslav D. Lutovac, Dejan V. Tosic and Brian L. Evans
lutovac@iritel.bg.ac.yu tosic@galeb.etf.bg.ac.yu bevans@ece.utexas.edu

A.3.1 References

1. Linear Technology Corporation
Monolithic Filter Handbook, 1990.

2. M. D. Lutovac, D. Novakovic, I. Markoski,
"Selective SC-filters with low passive sensitivity,"
Electronics Letters vol. 33, no. 8, pp. 674–675, Apr. 1997.

3. M.D.Lutovac, D. V. Tosic, D.Novakovic,
"Programmable low-pass/high-pass SC-filters,"
Proc.9th Conf. MELECON '98, Tel-Aviv, Israel May 1998, pp.673–677

4. D.V. Tosic, M.D. Lutovac, B.L. Evans, I.M. Markoski,
"A tool for symbolic analysis and design of analog active filters,"
5th Int. Workshop, Symbolic Methods, Applications, Circuit Design,
SMACD'98, Kaiserslautern, Germany, pp. 71–74, Oct. 1998.

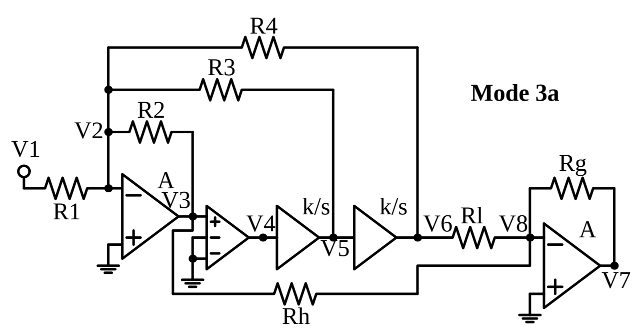
5. M. D. Lutovac, D. V. Tosic and B. L. Evans,
"Advanced Filter Design for Signal Processing using MATLAB and Mathematica,"
http://iritel.iritel.bg.ac.yu/˜lutovac/www/afdhome.htm
http://galeb.etf.bg.ac.yu/˜tosic/afdhome.htm
http://www.ece.utexas.edu/˜bevans/

A.3.2 Initialization

```
SetDirectory[HomeDirectory[]];
<<afd\math\m\clearall.m
<<afd\math\m\drawafil.m
<<afd\math\m\drawasc.m
```

A.3.3 Schematic

```
DrawMode3a[0,0,1/2,1/0.8,8];
```



A.3.4 Circuit Analysis

Reduced Modified Nodal Analysis

```
CircuitEquations = {
  V1 == Vg
, (V2-V1)/R1 + (V2-V3)/R2 + (V2-V5)/R3 + (V2-V6)/R4 == 0
, V3 == -A*V2
, V4 == V3
, V5 == V4*k/s
, V6 == V5*k/s
, (V8-V6)/Rl + (V8-V3)/Rh + (V8-V7)/Rg == 0
, V7 == -A*V8
};
NodeVoltages = {V1,V2,V3,V4,V5,V6,V7,V8};
CircuitResponse = Together[Flatten[
 Solve[CircuitEquations,NodeVoltages]
]];
Print["V1 = ", V1 /. CircuitResponse]
Print["V7 = ", V7 /. CircuitResponse]
```

```
V1 = Vg
         2                2                 2
V7 = -((A  R2 R3 R4 s (-(k  Rg Rh) - Rg Rl s ) Vg) /
    ((Rg Rh + Rg Rl + Rh Rl + A Rh Rl)
                                                3
      ((R1 R2 R3 + R1 R2 R4 + R1 R3 R4 + R2 R3 R4) s  -
               2                2
        A (-(k  R1 R2 R3 s) + s  (-(k R1 R2 R4) - R1 R3 R4 s)))))
```

A.3.5 Voltage Transfer Function

```
H = V7/V1 /. CircuitResponse //Together //Simplify;
H3a = Limit[H, A->Infinity];
Print["H(s) = ",
 Factor[Collect[Numerator[H3a],s]]/Factor[Collect[Denominator[H3a],s]]
]
```

```
                            2          2
              Rg R2 R3 R4 (k  Rh + Rl s )
H(s) = ------------------------------------------
                  2                            2
       Rh Rl R1 (k  R2 R3 + k R2 R4 s + R3 R4 s )
```

A.3.6 Definitions and Procedures

```
PoleQpole[H_,s_] := Module[{den,fp,Qp},
  den = Denominator[H];
  fp = Sqrt[Coefficient[den,s,0]/Coefficient[den,s,2]]/(2*Pi);
  Qp = (Coefficient[den,s,2]/Coefficient[den,s,1])*(2*Pi*fp);
  Simplify[{fp, Qp}]];
```

```
ZeroQzero[H_,s_] := Module[{fz,num,Qz0},
  num = Numerator[H];
  Qz0 = (Coefficient[num,s,2]/Coefficient[num,s,1]);
  fz = Sqrt[Coefficient[num,s,0]/Coefficient[num,s,2]]/(2*Pi);
  Simplify[{fz, Qz0*fz}]];
Sensitivity[F_,x_] := (x/F)*D[F,x];
GSP[F_,A_] := Limit[A*Sensitivity[F,A],A->Infinity]//Simplify;
PrintLabeledList[expressions_List,labels_List] := Map[
 Print[#[[1]]," = ",#[[2]]]&
,Transpose[{labels,expressions}]
];
```

A.3.7 Poles, Zeros, Q-Factors

```
{fp,Qp} = Simplify[PoleQpole[H,s]];
PrintLabeledList[{fp,Qp},{"fp","Qp"}];
```

```
                                 2
                              A k  R1 R2 R3
     Sqrt[-------------------------------------------------------]
          R1 R2 R3 + R1 R2 R4 + R1 R3 R4 + A R1 R3 R4 + R2 R3 R4
fp = ------------------------------------------------------------
                                 2 Pi
                                  2
                               A k  R1 R2 R3
Qp = (Sqrt[-------------------------------------------------------]
           R1 R2 R3 + R1 R2 R4 + R1 R3 R4 + A R1 R3 R4 + R2 R3 R4
    (R1 R2 R3 + R1 R2 R4 + R1 R3 R4 + A R1 R3 R4 + R2 R3 R4)) / (A k R1 R2 R4)
```

```
fp0 = Limit[fp, A -> Infinity];
Qp0 = Simplify[Limit[Qp, A -> Infinity]/.k->1];
PrintLabeledList[{fp0,Qp0},{"fp","Qp"}];
```

```
          2
         k  R2
     Sqrt[-----]
          R4
fp = -----------
        2 Pi
            R2
     R3 Sqrt[--]
            R4
Qp = -----------
         R2
```

```
{fz,Qz} = Simplify[ZeroQzero[H,s]];
PrintLabeledList[{fz,Qz},{"fz","Qz"}];

                                 1
Power::infy: Infinite expression - encountered.
                                 0
           2
          k  Rh
     Sqrt[-----]
           Rl
fz = -----------
        2 Pi
Qz = ComplexInfinity
```

■ A.3.8 Gain-Sensitivity Product (GSP)

```
GSPfp = GSP[fp,A];
GSPQp = GSP[Qp,A];
PrintLabeledList[{GSPfp,GSPQp},{"GSPfp","GSPQp"}];

        1    R2     R2     R2
GSPfp = - + ---- + ---- + ----
        2   2 R1   2 R3   2 R4
          1     R2     R2     R2
GSPQp = -(-) - ---- - ---- - ----
          2    2 R1   2 R3   2 R4
```

■ A.3.9 Design

▪ Find Element Values

```
DesignMode3a[K_,Qp_,wp_,wz_,fclk_,P_:100,R1_:R1nom,R2_:R2nom,Rh_:Rhnom]:=
 Module[{R3,R4,Rl,Rg},
 R4 = R2*(2*Pi*fclk/(P*wp))^2;
 R3 = Qp*Sqrt[R2*R4];
 Rl = Rh*(2*Pi*fclk/(P*wz))^2;
 Rg = K*Rh*R1/R2;
{R1,R2,R3,R4,Rl,Rh,Rg}
];
{R1,R2,R3,R4,Rl,Rh,Rg} = Together[
                         DesignMode3a[K,Q,W,Z,Fc,P,R1n,R2n,Rhn]];
PrintLabeledList[{R1,R2,R3,R4,Rl,Rh,Rg}
                ,{"R1","R2","R3","R4","Rl","Rh","Rg"}];

R1 = R1n
R2 = R2n
                2   2    2
              Fc  Pi  R2n
R3 = 2 Q Sqrt[------------]
                 2  2
                P  W
```

```
          2    2
     4 Fc   Pi   R2n
R4 = -------------
           2  2
          P  W
          2    2
     4 Fc   Pi   Rhn
R1 = -------------
           2  2
          P  Z
Rh = Rhn
     K Rhn R1n
Rg = ---------
        R2n
```

■ **Test**

```
H3atest  = Simplify[ExpandAll[Together[Limit[H , A -> Infinity]] /.
 {Sqrt[x_^2*y_^2/z_^2] -> x*y/z, Sqrt[x_^2*y_^2*p_^2/z_^2] -> x*y*p/z
, Sqrt[x_^2*y_^2*p_^2/(z_^2*n_^2)] -> x*y*p/(z*n)} ]];
num = Numerator[H3atest];
den = Denominator[H3atest];
numlist = CoefficientList[num,s];
denlist = CoefficientList[den,s];
K3at =(numlist[[3]]/denlist[[3]]);
H3at =  K3at *
Simplify[num/numlist[[3]]]/Simplify[den/denlist[[3]]] /. k-> 2*Pi*Fc/P
```

```
     2    2
 K (s  + Z )
-------------
 2   s W    2
s  + --- + W
      Q
```

Design Examples

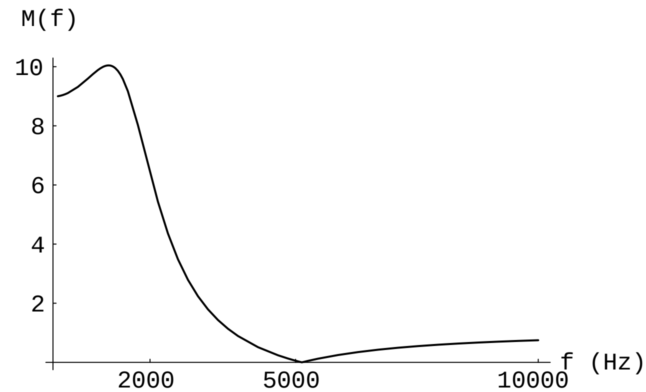
```
values = {K ->1, Q -> 1.0349, W -> 2*Pi*1710.9457, Z -> 2*Pi*5129.3034
  , Fc -> 256.*10^3, P -> 100, R12 ->23.16*10^3
  , R22 ->10.*10^3, Rh2 ->238.6*10^3} //N;
h1 = H  /. k-> 2*Pi*Fc/P /.values //N;
H3atest = Limit[h1, A->Infinity];
H3atest =H3atest /. Sqrt[x_^2] -> x;
num = Numerator[H3atest];
den = Denominator[H3atest];
numlist = CoefficientList[num,s];
denlist = CoefficientList[den,s];
K3at =(numlist[[3]]/denlist[[3]]);
H3at =  K3at * Simplify[num/numlist[[3]]]/Simplify[den/denlist[[3]]]
```

```
                 9        2
   1. (1.03867 10  + 1. s )
-------------------------------
          8                   2
1.15567 10  + 10387.7 s + 1. s
```

```
Rexample1 = N[{R1,R2,R3,R4,Rl,Rh,Rg}/. values] /. Sqrt[R2n^2]->R2n;
PrintLabeledList[Rexample1,{"R1","R2","R3","R4","Rl","Rh","Rg"}];
```

```
R1 = R1n
R2 = R2n
R3 = 1.54847 R2n
R4 = 2.23876 R2n
Rl = 0.249094 Rhn
Rh = Rhn
     1. Rhn R1n
Rg = ----------
        R2n
```

```
Plot[{Abs[H3at] /. s -> I*2*Pi*f}
, {f, 100, 10000}
,PlotRange -> All
,Ticks -> {{0,2000,5000,10000},{0,2,4,6,8,10}}
, AxesLabel -> {"f (Hz)","M(f)"}
];
```



A.3.10 Optimization

Find R2/R1 for Low Gain-Sensitivity Product

```
sf = (Simplify[N[Together[GSPfp /. values]]
     /. Sqrt[x_^2] -> x/. Sqrt[x_^2*y_^2] -> x*y ]
     /. Sqrt[R2n^2] -> R2n)
```

```
          0.5 R2n
1.04624 + -------
            R1n
```

Remark

We have to choose R1 > R2 to minimize GSP.

A.4 OTA-C General Biquad Analysis and Design

Miroslav D. Lutovac, Dejan V. Tosic and Brian L. Evans
lutovac@iritel.bg.ac.yu tosic@galeb.etf.bg.ac.yu bevans@ece.utexas.edu

■ A.4.1 References

1. Wai-Kai-Chen, Ed.,
"The Circuits and Filters Handbook," p. 2479,
"CRC Press," "Boca Raton, Florida," "1995."

2. D.V. Tosic, M.D. Lutovac, B.L. Evans, I.M. Markoski,
"A tool for symbolic analysis and design of analog active filters,"
5th Int. Workshop, Symbolic Methods, Applications, Circuit Design,
SMACD'98, Kaiserslautern, Germany, pp. 71–74, Oct. 1998.

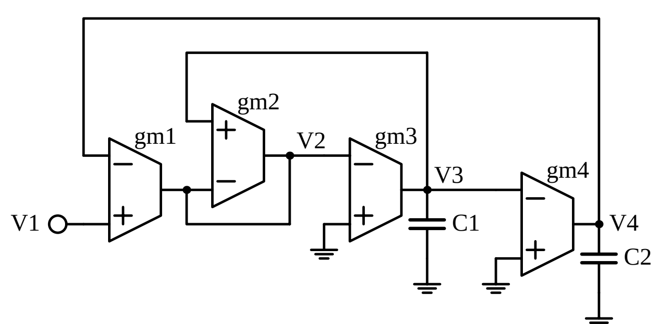
3. M. D. Lutovac, D. V. Tosic and B. L. Evans,
"Advanced Filter Design for Signal Processing using MATLAB and Mathematica,"
http://iritel.iritel.bg.ac.yu/~lutovac/www/afdhome.htm
http://galeb.etf.bg.ac.yu/~tosic/afdhome.htm
http://www.ece.utexas.edu/~bevans/

■ A.4.2 Initialization

```
SetDirectory[HomeDirectory[]];
<<afd\math\m\clearall.m
<<afd\math\m\drawafil.m
<<afd\math\m\drawaota.m
```

■ A.4.3 Schematic

```
DrawOTAb[0, 0, 1/2, 1/08, 8]
```



A.4.4 Circuit Analysis

Reduced Modified Nodal Analysis

```
CircuitEquations = {
  V1 == Vg
, -(V1-V4)*gm1 - (V3-V2)*gm2 == 0
, -(-V2)*gm3 + V3/(1/(s*C1)) == 0
, -(-V3)*gm4 + V4/(1/(s*C2)) == 0
};

NodeVoltages = {V1,V2,V3,V4};
CircuitResponse = Together[Flatten[
 Solve[CircuitEquations,NodeVoltages]
]];
```

A.4.5 Voltage Transfer Function

```
H = V4/V1 /. CircuitResponse //Together //Simplify;
Print["H(s) = ", H]
Hhp = V2/V1 /. CircuitResponse //Together //Simplify;
Print["Hhp(s) = ",  Hhp]
Hbp = V3/V1 /. CircuitResponse //Together //Simplify;
Print["Hbp(s) = ", Hbp]
```

```
                      gm1 gm3 gm4
H(s) = -----------------------------------------
                                              2
       gm1 gm3 gm4 + C2 gm2 gm3 s + C1 C2 gm2 s
                                   2
                       C1 C2 gm1 s
Hhp(s) = -----------------------------------------
                                                2
         gm1 gm3 gm4 + C2 gm2 gm3 s + C1 C2 gm2 s
                         C2 gm1 gm3 s
Hbp(s) = -(-----------------------------------------)
                                                  2
           gm1 gm3 gm4 + C2 gm2 gm3 s + C1 C2 gm2 s
```

A.4.6 Definitions and Procedures

```
PoleQpole[H_,s_] := Module[{den,fp,Qp},
  den = Denominator[H];
  fp = Sqrt[Coefficient[den,s,0]/Coefficient[den,s,2]]/(2*Pi);
  Qp = (Coefficient[den,s,2]/Coefficient[den,s,1])*(2*Pi*fp);
  Simplify[{fp, Qp}]];
ZeroQzero[H_,s_] := Module[{fz,num,Qz0},
  num = Numerator[H];
  Qz0 = (Coefficient[num,s,2]/Coefficient[num,s,1]);
  fz = Sqrt[Coefficient[num,s,0]/Coefficient[num,s,2]]/(2*Pi);
  Simplify[{fz, Qz0*fz}]];
```

```
PrintLabeledList[expressions_List,labels_List] := Map[
 Print[#[[1]]," = ",#[[2]]]&
,Transpose[{labels,expressions}]
];
```

A.4.7 Poles, Zeros, Q-Factors

```
{fp,Qp} = Simplify[PoleQpole[H,s]];
Klp = H /. s-> 0
PrintLabeledList[{fp,Qp},{"fp","Qp"}];

1
          gm1 gm3 gm4
     Sqrt[-----------]
           C1 C2 gm2
fp = -----------------
           2 Pi
             gm1 gm3 gm4
     C1 Sqrt[-----------]
              C1 C2 gm2
Qp = --------------------
             gm3
```

A.4.8 Design

Find Element Values

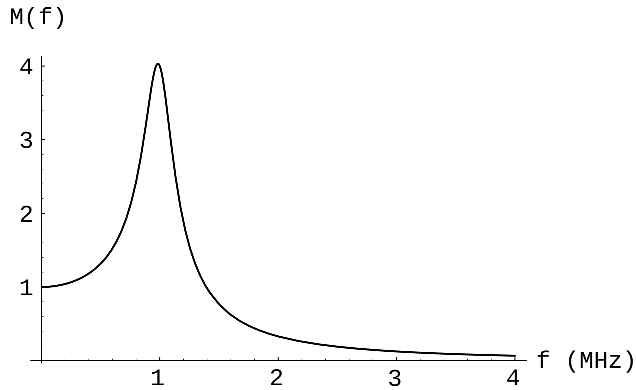
```
DesignOTA1[Qp_,wp_,C1_,C2_,gm1_,gm2_] :=
 Module[{gm3,gm4},
 gm3 = C1*wp/Qp;
 gm4 = C2*wp*Qp*gm2/gm1;
{C1,C2,gm1,gm2,gm3,gm4}
];
{C1,C2,gm1,gm2,gm3,gm4} = Together[DesignOTA1[Q,W,c1,c2,g1,g2]];
PrintLabeledList[{C1,C2,gm1,gm2,gm3,gm4},{"C1","C2","gm1","gm2","gm3","gm4"}];

C1 = c1
C2 = c2
gm1 = g1
gm2 = g2
      c1 W
gm3 = ----
       Q
      c2 g2 Q W
gm4 = ---------
         g1
```

A.4.9 Test

```
Simplify[H]

         2
      Q W
-----------------
   2            2
Q s  + s W + Q W
```

Design Examples

```
values = {Q -> 4., W -> N[2*Pi*10^6] ,
  c1 -> 10.*10^(-12), c2 ->10.*10^(-12), g1 -> g , g2 -> g} //N;
h1 = Together[H  /. values] /. 1. -> 1;
Print["gm3 = ",10^3*gm3 /. values, " mS "]
Print["gm4 = ",10^3*gm4 /. values, " mS "]
h = (Numerator[h1]/g)/ (Simplify[Denominator[h1]/g])

gm3 = 0.015708 mS
gm4 = 0.251327 mS
                    13
          3.94784 10
--------------------------------
          13            6      2
3.94784 10   + 1.5708 10  s + s

PrintLabeledList[N[{Q,W/(2*Pi)} /. values]
,{"Qp","fp (Hz)"}];
Print["------------------------"]
Rexample1 = N[{C1*10^(12),C2*10^(12),gm1,gm2,gm3*10^(3),gm4*10^(3)}/. values] ;
PrintLabeledList[Rexample1
                ,{"C1 (pF)","C2 (pF)","gm1","gm2","gm3 (mS)","gm4 (mS)"}];

Qp = 4.
              6
fp (Hz) = 1. 10
------------------------
C1 (pF) = 10.
C2 (pF) = 10.
gm1 = g
gm2 = g
gm3 (mS) = 0.015708
gm4 (mS) = 0.251327

Plot[{Abs[h] /. s -> I*2*Pi*f*10^6}
, {f, 0.01, 4}
,PlotRange -> All
, AxesLabel -> {"f (MHz)","M(f)"}
];
```


M(f)
4
3
2
1
1
2
3
4
f (MHz)

A.5 Lowpass-Medium-Q-Factor Active RC Filter Analysis and Design

Miroslav D. Lutovac, Dejan V. Tosic and Brian L. Evans
lutovac@iritel.bg.ac.yu tosic@galeb.etf.bg.ac.yu bevans@ece.utexas.edu

A.5.1 References

1. G. S. Moschytz and P. Horn,
Active Filter Design Handbook
For Use with Programmable Pocket Calculators and Mini Computers
John Wiley and Sons, Ltd., New York, 1981, pp. 38–39.

2. R. P. Sallen and E. L. Key,
"A practical method of designing RC active filters,"
IRE Trans. Circuit Theory, CT–2, Mar. 1955, pp. 74–85.

3. D.V. Tosic, M.D. Lutovac, B.L. Evans, I.M. Markoski,
"A tool for symbolic analysis and design of analog active filters,"
5th Int. Workshop, Symbolic Methods, Applications, Circuit Design,
SMACD'98, Kaiserslautern, Germany, pp. 71–74, Oct. 1998.

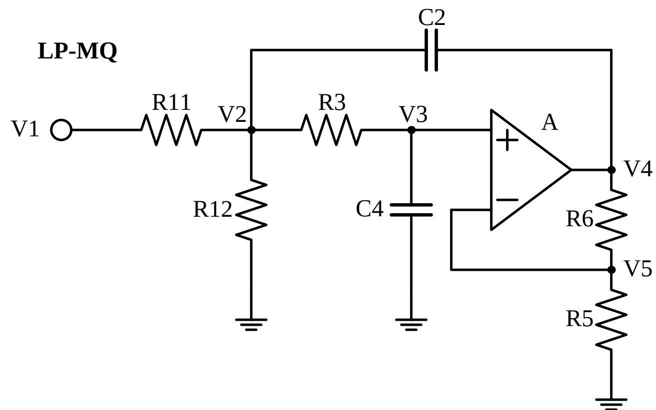
4. M. D. Lutovac, D. V. Tosic and B. L. Evans,
"Advanced Filter Design for Signal Processing using MATLAB and Mathematica,"
http://iritel.iritel.bg.ac.yu/˜lutovac/www/afdhome.htm
http://galeb.etf.bg.ac.yu/˜tosic/afdhome.htm
http://www.ece.utexas.edu/˜bevans/

A.5.2 Initialization

```
SetDirectory[HomeDirectory[]];
<<afd\math\m\clearall.m
<<afd\math\m\drawafil.m
<<afd\math\m\drawarc.m
```

■ A.5.3 Schematic

```
DrawLPMQ[0,0,1/2,1/0.8];
```



■ A.5.4 Circuit Analysis

■ Reduced Modified Nodal Analysis

```
CircuitEquations = {V1 == Vg
, (V2-V1)/R11 + V2/R12 + (V2-V3)/R3 + (V2-V4)/(1/(s*C2)) == 0
, (V3-V2)/R3 + V3/(1/(s*C4)) == 0
, (V5-V4)/R6 + V5/R5 == 0
, (V3-V5)*A == V4};
NodeVoltages = {V1,V2,V3,V4,V5};
CircuitResponse = Together[Flatten[
 Solve[CircuitEquations,NodeVoltages]
]];
Print["V1 = ", V1 /. CircuitResponse]
Print["V4 = ", V4 /. CircuitResponse]
```

```
V1 = Vg
        2
V4 = -((A  R12 R3 R5 Vg) /
      2
     (A  C2 R11 R12 R3 R5 s - (-R5 - A R5 - R6) (1 + C4 R3 s)
        (R11 R12 + R11 R3 + R12 R3 + C2 R11 R12 R3 s) +
       (-R5 - A R5 - R6) (R11 R12 + A C2 R11 R12 R3 s))) -
  (A R12 R3 (-R5 - A R5 - R6) Vg) /
     2
   (A  C2 R11 R12 R3 R5 s - (-R5 - A R5 - R6) (1 + C4 R3 s)
      (R11 R12 + R11 R3 + R12 R3 + C2 R11 R12 R3 s) +
     (-R5 - A R5 - R6) (R11 R12 + A C2 R11 R12 R3 s))
```

A.5.5 Voltage Transfer Function

```
H = V4/V1 /. CircuitResponse //Together //Simplify;
Ha = Limit[H, A->Infinity];
Print["H(s) = ",
 Collect[Numerator[Ha],s]/Collect[Denominator[Ha],s]]
```

```
H(s) = (R12 (R5 + R6)) /
  (R11 R5 + R12 R5 + (C4 R11 R12 R5 + C4 R11 R3 R5 + C4 R12 R3 R5 - C2 R11 R12 R6) s +
                          2
    C2 C4 R11 R12 R3 R5 s )
```

A.5.6 Definitions and Procedures

```
PoleQpole[H_,s_] := Module[{den,fp,Qp},
  den = Denominator[H];
  fp = Sqrt[Coefficient[den,s,0]/Coefficient[den,s,2]]/(2*Pi);
  Qp = (Coefficient[den,s,2]/Coefficient[den,s,1])*(2*Pi*fp);
  Simplify[{fp, Qp}]];
ZeroQzero[H_,s_] := Module[{fz,num,Qz0},
  num = Numerator[H];
  Qz0 = (Coefficient[num,s,2]/Coefficient[num,s,1]);
  fz = Sqrt[Coefficient[num,s,0]/Coefficient[num,s,2]]/(2*Pi);
  Simplify[{fz, Qz0*fz}]];
Sensitivity[F_,x_] := (x/F)*D[F,x];
GSP[F_,A_] := Limit[A*Sensitivity[F,A],A->Infinity]//Simplify;
PrintLabeledList[expressions_List,labels_List] := Map[
 Print[#[[1]]," = ",#[[2]]]&
,Transpose[{labels,expressions}]
];
```

A.5.7 Poles, Zeros, Q-Factors

```
{fp,Qp} = Simplify[PoleQpole[H,s]];
PrintLabeledList[{fp,Qp},{"fp","Qp"}];
```

```
             R11 + R12
     Sqrt[----------------]
          C2 C4 R11 R12 R3
fp = ----------------------
              2 Pi
                              R11 + R12
Qp = (C2 C4 R11 R12 Sqrt[----------------] R3 (R5 + A R5 + R6)) /
                         C2 C4 R11 R12 R3
  (C2 R11 R12 R5 + C4 R11 R12 R5 + A C4 R11 R12 R5 + C4 R11 R3 R5 + A C4 R11 R3 R5 +
    C4 R12 R3 R5 + A C4 R12 R3 R5 + C2 R11 R12 R6 - A C2 R11 R12 R6 + C4 R11 R12 R6 +
    C4 R11 R3 R6 + C4 R12 R3 R6)
```

■ A.5.8 Gain-Sensitivity Product (GSP)

```
GSPfp = GSP[fp,A];
GSPQp = GSP[Qp,A];
PrintLabeledList[{GSPfp,GSPQp},{"GSPfp","GSPQp"}];

GSPfp = 0
                                                    2
                           C2 R11 R12 (R5 + R6)
GSPQp = ----------------------------------------------------------------
        R5 (C4 R11 R12 R5 + C4 R11 R3 R5 + C4 R12 R3 R5 - C2 R11 R12 R6)
```

■ A.5.9 Design

■ Find Element Values

```
DesignLPMQ[K_,Qp_,wp_,P_,C2x_,C4x_,R5x_] :=  Module[
{C2,C4,R1,R11,R12,R3,R5,R6,Ko},
 C2 = C2x;
 C4 = C4x;
 R1 = 1/(wp*Sqrt[C2x*C4x*P]);
 R3 = P*R1;
 R5 = R5x;
 R6 = R5*((1+P)*C4/C2-Sqrt[P*C4/C2]/Qp);
 Ko = 1+R6/R5;
 R11 = R1*Ko/K;
 R12 = R1*Ko/(Ko-K);
{R11,R12,C2,R3,C4,R5,R6}
];
{R11,R12,C2,R3,C4,R5,R6} = Together[DesignLPMQ[K,Q,W,P,C2x,C4x,R5x]];
```

■ A.5.10 Design Example

```
values = {K -> 1., Q -> 7.5, W -> 2*Pi*2500., P -> 1.5333078
 , C2x -> 33.*10^(-9), C4x -> 10.*10^(-9), R5x -> 6800.} //N;
PrintLabeledList[N[{K,Q,W/(2*Pi),P} /. values]
,{"K","Qp","fp (Hz)","P"} ];
Print["----------------"]
PrintLabeledList[Together[{R11,R12,C2*10^9,R3,C4*10^9,R5,R6} /. values],
{"R11 (ohm)","R12 (ohm)","C2 (nF)","R3 (ohm)","C4 (nF)","R5 (ohm)","R6 (ohm)"}];

K = 1.
Qp = 7.5
fp (Hz) = 2500.
P = 1.53331
----------------
R11 (ohm) = 4745.54
R12 (ohm) = 7011.9
C2 (nF) = 33.
R3 (ohm) = 4339.48
C4 (nF) = 10.
R5 (ohm) = 6800.
R6 (ohm) = 4602.13
```

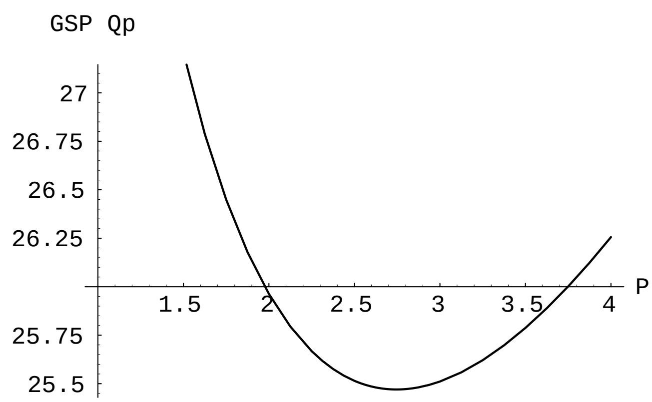
A.5.11 Optimization

Find P for Low Gain-Sensitivity Product

```
values = {K -> 1.476288, Q -> 7.5, W -> 2*Pi*2500.
 , C2x -> 100.*10^(-9), C4x -> 15.*10^(-9), R5x -> 10000.} //N;
gspQp = Together[Simplify[GSPQp] /. values]
```

```
                                           2
0.435711 (7.66667 - 0.344265 Sqrt[P] + 1. P)
--------------------------------------------
                   Sqrt[P]
```

```
P1 = 1.;
P2 = 4.;
Plot[gspQp /. values
, {P, P1, P2}
, AxesLabel -> {"P","GSP Qp"}
];
```




```
{GSPmin,Pset} = FindMinimum[gspQp,{P,P1,P2}]
```

```
{25.4701, {P -> 2.74571}}
```

```
PrintLabeledList[N[{K,Q,W/(2*Pi),P} /. values /.Pset]
,{"K","Qp","fp (Hz)","P"} ];
Print["----------------"]
PrintLabeledList[Together[{R11,R12,C2*10^9,R3,C4*10^9,R5,R6} /. values /.Pset],
{"R11 (ohm)","R12 (ohm)","C2 (nF)","R3 (ohm)","C4 (nF)","R5 (ohm)","R6 (ohm)"}];
```

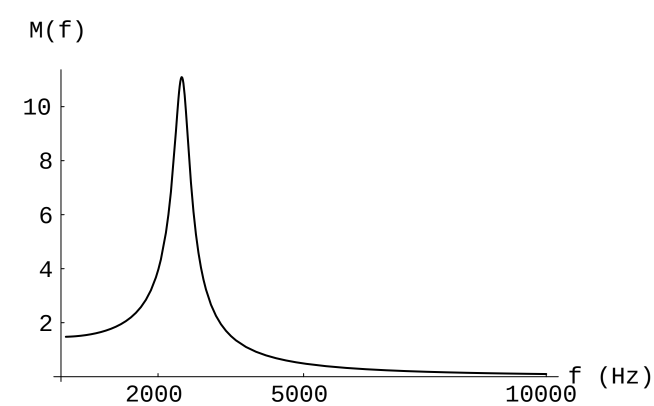
```
K = 1.47629
Qp = 7.5
fp (Hz) = 2500.
P = 2.74571
----------------
R11 (ohm) = 991.99
```

```
                        9
R12 (ohm) = 1.81368 10
C2 (nF) = 100.
R3 (ohm) = 2723.72
C4 (nF) = 15.
R5 (ohm) = 10000.
R6 (ohm) = 4762.89
```

```
Hhpmq = Together[Ha  /.Pset/. values] ;
num = Numerator[Hhpmq];
den = Denominator[Hhpmq];
numlist = CoefficientList[num,s];
denlist = CoefficientList[den,s];
Hhpmq = (1/denlist[[3]]) * num/(den/denlist[[3]])//Factor
```

```
                     8
         3.64259 10
-----------------------------
         8                  2
2.4674 10  + 2094.4 s + 1. s
```

```
Plot[{Abs[Hhpmq] /. s -> N[I*2*Pi*f]}
, {f, 100, 10000}
, PlotRange -> All
, Ticks -> {{0,2000,5000,10000},{0,2,4,6,8,10}}
, AxesLabel -> {"f (Hz)","M(f)"} ];
```



- **Remark**

We choose P=2.7457 as a good choice because we obtain small GSP and 1/R12=0.

A.6 General Purpose High-Q-Factor RC Filter Analysis and Design

Miroslav D. Lutovac, Dejan V. Tosic and Brian L. Evans
lutovac@iritel.bg.ac.yu tosic@galeb.etf.bg.ac.yu bevans@ece.utexas.edu

■ A.6.1 References

1. G. S. Moschytz and P. Horn,
Active Filter Design Handbook
For Use with Programmable Pocket Calculators and Mini Computers
John Wiley and Sons, Ltd., New York, 1981 pp. 64–65.

2. W. J. Kerwin, L. P. Huelsman, and R. W. Newcomb,
State-variable synthesis for insensitive integrated circuit transfer functions,
IEEE J. Solid-State Circuits, SC–2, pp. 87–92, September, 1967.

3. D.V. Tosic, M.D. Lutovac, B.L. Evans, I.M. Markoski,
"A tool for symbolic analysis and design of analog active filters,"
5th Int. Workshop, Symbolic Methods, Applications, Circuit Design,
SMACD'98, Kaiserslautern, Germany, pp. 71–74, Oct. 1998.

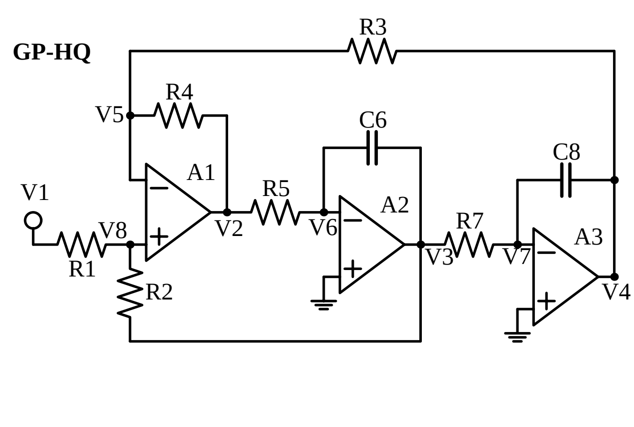
4. M. D. Lutovac, D. V. Tosic and B. L. Evans,
"Advanced Filter Design for Signal Processing using MATLAB and Mathematica,"
http://iritel.iritel.bg.ac.yu/˜lutovac/www/afdhome.htm
http://galeb.etf.bg.ac.yu/˜tosic/afdhome.htm
http://www.ece.utexas.edu/˜bevans/

■ A.6.2 Initialization

```
SetDirectory[HomeDirectory[]];
<<afd\math\m\clearall.m
<<afd\math\m\drawafil.m
<<afd\math\m\drawarc.m
```

■ A.6.3 Schematic

```
DrawGPHQ[0,0,1/2,1/0.8];
```



A.6.4 Circuit Analysis

Reduced Modified Nodal Analysis

```
CircuitEquations = {V1 == Vg
, V2 == A*(V8 - V5)
, V3 == -A*V6
, V4 == -A*V7
, (V5-V2)/R4 + (V5-V4)/R3 == 0
, (V6-V2)/R5 + (V6-V3)/(1/(s*C6)) ==0
, (V7-V3)/R7 + (V7-V4)/(1/(s*C8)) ==0
, (V8-V1)/R1 + (V8-V3)/R2 ==0};
NodeVoltages = {V1,V2,V3,V4,V5,V6,V7,V8};
CircuitResponse = Together[Flatten[
 Solve[CircuitEquations,NodeVoltages]
]];
```

A.6.5 Voltage Transfer Function

```
H = V4/V1 /. CircuitResponse //Together //Simplify;
Ha = Limit[H, A->Infinity];
Print["H(s) = ",
 Collect[Numerator[Ha],s]/Collect[Denominator[Ha],s]]
```

```
H(s) = (R2 (R3 + R4)) /
  (R1 R4 + R2 R4 + (C8 R1 R3 R7 + C8 R1 R4 R7) s +
                                                  2
    (C6 C8 R1 R3 R5 R7 + C6 C8 R2 R3 R5 R7) s )
```

```
Klp = Factor[Ha /. s->0]
```

```
R2 (R3 + R4)
------------
(R1 + R2) R4
```

```
Hh = V2/V1 /. CircuitResponse //Together //Simplify;
Hha = Limit[Hh, A->Infinity];
        Print["Hh(s) = ",
 Collect[Numerator[Hha],s]/Collect[Denominator[Hha],s]]
```

```
                                   2
Hh(s) = (C6 C8 R2 (R3 + R4) R5 R7 s ) /
  (R1 R4 + R2 R4 + (C8 R1 R3 R7 + C8 R1 R4 R7) s +
                                                  2
    (C6 C8 R1 R3 R5 R7 + C6 C8 R2 R3 R5 R7) s )
```

```
Khp = Factor[Limit[Hha , s->Infinity]]
```

```
R2 (R3 + R4)
------------
(R1 + R2) R3
```

```
Hb = V3/V1 /. CircuitResponse //Together //Simplify;
Hba = Limit[Hb, A->Infinity];
Print["Hb(s) = ",
 Collect[Numerator[Hba],s]/Collect[Denominator[Hba],s]]

Hb(s) = -((C8 R2 (R3 + R4) R7 s) /
    (R1 R4 + R2 R4 + (C8 R1 R3 R7 + C8 R1 R4 R7) s +
                                                    2
      (C6 C8 R1 R3 R5 R7 + C6 C8 R2 R3 R5 R7) s ))

num = Numerator[Hba];
den = Denominator[Hba];
numlist = CoefficientList[num,s];
denlist = CoefficientList[den,s];
Hbp = Factor[numlist[[2]]/denlist[[2]]]

  R2
-(--)
  R1
```

A.6.6 Definitions and Procedures

```
PoleQpole[H_,s_] := Module[{den,fp,Qp},
  den = Denominator[H];
  fp = Sqrt[Coefficient[den,s,0]/Coefficient[den,s,2]]/(2*Pi);
  Qp = (Coefficient[den,s,2]/Coefficient[den,s,1])*(2*Pi*fp);
  Simplify[{fp, Qp}]];
ZeroQzero[H_,s_] := Module[{fz,num,Qz0},
  num = Numerator[H];
  Qz0 = (Coefficient[num,s,2]/Coefficient[num,s,1]);
  fz = Sqrt[Coefficient[num,s,0]/Coefficient[num,s,2]]/(2*Pi);
  Simplify[{fz, Qz0*fz}]];
Sensitivity[F_,x_] := (x/F)*D[F,x];
GSP[F_,A_] := Limit[A*Sensitivity[F,A],A->Infinity]//Simplify;
GSPepsA[F_,epsA_] := -(1/epsA)*Sensitivity[F,epsA]//Together;
PrintLabeledList[expressions_List,labels_List] := Map[
 Print[#[[1]]," = ",#[[2]]]&
,Transpose[{labels,expressions}]
];
```

A.6.7 Poles, Zeros, Q-Factors

```
{fp,Qp} = Simplify[PoleQpole[H,s]];
fp0 = Together[Limit[fp,A-> Infinity]];
Qp0 = Limit[Qp,A-> Infinity];
PrintLabeledList[{fp0,Qp0},{"fp","Qp"}];
```

```
             R4
     Sqrt[--------------]
          C6 C8 R3 R5 R7
fp = --------------------
             2 Pi
                                   R4
     C6 (R1 + R2) R3 R5 Sqrt[--------------]
                              C6 C8 R3 R5 R7
Qp = ---------------------------------------
                 R1 (R3 + R4)
```

■ A.6.8 Gain-Sensitivity Product (GSP)

```
fpepsA = Together[fp /. A->1/e];
QpepsA = Together[Qp /. A->1/e];
GSPQp = Simplify[GSPepsA[QpepsA,e] /. e->0];
GSPfp = Simplify[GSPepsA[fpepsA,e] /. e->0];
PrintLabeledList[{GSPfp,GSPQp},{"GSPfp","GSPQp"}];
```

```
               2                                   2        2
        -(R1 R3 ) + 2 R1 R3 R4 + 3 R2 R3 R4 + R1 R4  + R2 R4
GSPfp = -----------------------------------------------------
                        2 R3 (R1 R4 + R2 R4)
                2   2                      2                2   2              2   3
GSPQp = (2 C6 R1  R3  R4 R5 + 4 C6 R1 R2 R3  R4 R5 + 2 C6 R2  R3  R4 R5 - C8 R1  R3  R7 -
         2   2                      2                2   2                2      2
    C8 R1  R3  R4 R7 + 3 C8 R1 R2 R3  R4 R7 + 2 C8 R2  R3  R4 R7 - 3 C8 R1  R3 R4  R7 -
                    2           2   3                 3
    2 C8 R1 R2 R3 R4  R7 - C8 R1  R4  R7 - C8 R1 R2 R4  R7) /
  (2 R3 (R1 R4 + R2 R4) (C8 R1 R3 R7 + C8 R1 R4 R7))
```

■ A.6.9 Design

■ Find Element Values

```
DesignGPHQ[Qp_,wp_,Cx_,Rx_] :=
 Module[{C6,C8,R1,R2,R3,R4,R5,R7,Ro},
 C6 = Cx;
 C8 = Cx;
 Ro = 1/(wp*Cx);
 R1 = Rx;
 R3 = Rx;
 R5 = Rx;
 R7 = Rx;
 R4 = Rx*(Rx/Ro)^2;
 R2 = Rx*(Qp*(1+R4/Rx)/Sqrt[R4/Rx]-1);
 {R1,R2,R3,R4,R5,C6,R7,C8}];
 {R1,R2,R3,R4,R5,C6,R7,C8} = DesignGPHQ[Q,Wp,Co,Rd];
```

■ Design Examples

```
values = {Q -> 6., Wp -> 2*Pi*1500.
, Co -> 68.*10^(-9), Rd -> 1800.} //N;
gspQpfp = Together[Abs[GSPQp/2]+Abs[Q*GSPfp] /. values];
```

```
PrintLabeledList[N[{Q,Wp/(2*Pi),gspQpfp} /. values]
,{"Qp","fp (Hz)","GSP"} ];
Print["------------------"]
PrintLabeledList[Together[{R1,R2,R3,R4,R5,C6*10^9,R7,C8*10^9} /. values]
,{"R1 (ohm)","R2 (ohm)","R3 (ohm)","R4 (ohm)","R5 (ohm)","C6 (nF)"
,"R7 (ohm)","C8 (nF)"}];
```

```
Qp = 6.
fp (Hz) = 1500.
GSP = 17.1412
------------------
R1 (ohm) = 1800.
R2 (ohm) = 20020.9
R3 (ohm) = 1800.
R4 (ohm) = 2395.4
R5 (ohm) = 1800.
C6 (nF) = 68.
R7 (ohm) = 1800.
C8 (nF) = 68.
```

```
values = {Q -> 25., Wp -> 2*Pi*1500.
, Co -> 22.*10^(-9), Rd -> 4700.} //N;
gspQpfp = Together[Abs[GSPQp/2]+Abs[Q*GSPfp] /. values];
PrintLabeledList[N[{Q,Wp/(2*Pi),gspQpfp} /. values]
,{"Qp","fp (Hz)","GSP"} ];
Print["------------------"]
PrintLabeledList[Together[{R1,R2,R3,R4,R5,C6*10^9,R7,C8*10^9} /. values]
,{"R1 (ohm)","R2 (ohm)","R3 (ohm)","R4 (ohm)","R5 (ohm)","C6 (nF)"
,"R7 (ohm)","C8 (nF)"}];
```

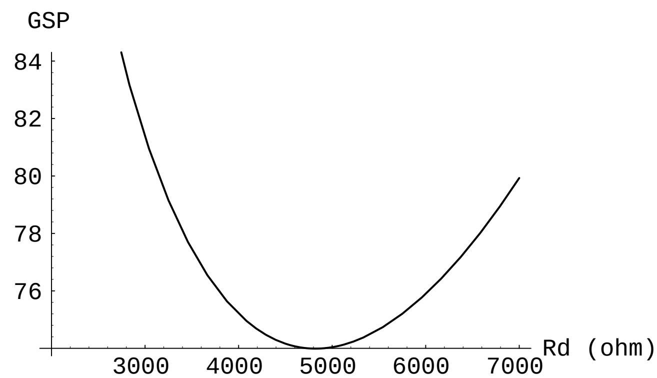
```
Qp = 25.
fp (Hz) = 1500.
GSP = 74.014
------------------
R1 (ohm) = 4700.
R2 (ohm) = 230378.
R3 (ohm) = 4700.
R4 (ohm) = 4463.56
R5 (ohm) = 4700.
C6 (nF) = 22.
R7 (ohm) = 4700.
C8 (nF) = 22.
```

A.6.10 Optimization

Find Rd for Low Gain-Sensitivity Product

```
values = {Q -> 25., Wp -> 2*Pi*1500.
, Co -> 22.*10^(-9)} //N;
gspQpfp1 = Together[Abs[GSPQp/2.] + Abs[Q*GSPfp] /. values] //Simplify;
```

```
r1 = 2000;
r2 = 7000;
Plot[{gspQpfp1}
, {Rd, r1, r2}
, AxesLabel -> {"Rd (ohm)","GSP"}
];
```



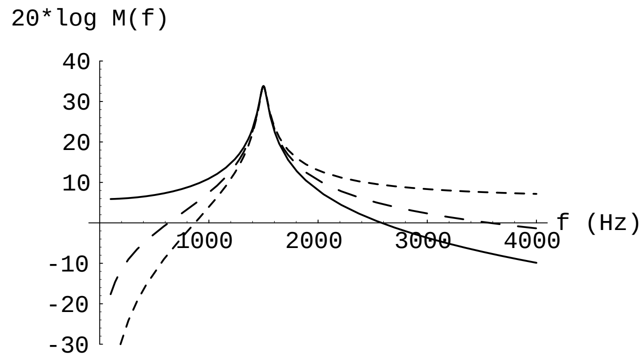

```
{Gspmin,Rset} = FindMinimum[gspQpfp1, {Rd, r1, r2}]
```

```
{73.99, {Rd -> 4822.22}}
```

```
values = {Q -> 25., Wp -> 2*Pi*1500.
, Co -> 22.*10^(-9)} //N;
gspQpfp = Together[Abs[GSPQp/2]+Abs[Q*GSPfp] /. values /. Rset];
PrintLabeledList[N[{Q,Wp/(2*Pi),gspQpfp} /. values/. Rset]
,{"Qp","fp (Hz)","GSP"} ];
Print["------------------"]
PrintLabeledList[Together[{R1,R2,R3,R4,R5,C6*10^9,R7,C8*10^9} /. values/.Rset]
,{"R1 (ohm)","R2 (ohm)","R3 (ohm)","R4 (ohm)","R5 (ohm)","C6 (nF)"
,"R7 (ohm)","C8 (nF)"}];
```

```
Qp = 25.
fp (Hz) = 1500.
GSP = 73.99
------------------
R1 (ohm) = 4822.22
R2 (ohm) = 236289.
R3 (ohm) = 4822.22
R4 (ohm) = 4820.9
R5 (ohm) = 4822.22
C6 (nF) = 22.
R7 (ohm) = 4822.22
C8 (nF) = 22.
```

```
Hlp = Ha /. values /. Rset//Together;
num = Numerator[Hlp];
den = Denominator[Hlp];
numlist = CoefficientList[num,s];
denlist = CoefficientList[den,s];
Hlp = 1/denlist[[3]] * num/(den/denlist[[3]])//Factor

                     8
          1.74124 10
-------------------------------
          7                   2
8.88264 10  + 376.991 s + 1. s

Hhp = Hha /. values /. Rset//Together;
num = Numerator[Hhp];
den = Denominator[Hhp];
numlist = CoefficientList[num,s];
denlist = CoefficientList[den,s];
Hhp = 1/denlist[[3]] * num/(den/denlist[[3]])//Factor

                   2
          1.95973 s
-------------------------------
          7                   2
8.88264 10  + 376.991 s + 1. s

Hbp = Hba /. values /. Rset//Together;
num = Numerator[Hbp];
den = Denominator[Hbp];
numlist = CoefficientList[num,s];
denlist = CoefficientList[den,s];
Hbp = 1/denlist[[3]] * num/(den/denlist[[3]])//Factor

          -18472.6 s
-------------------------------
          7                   2
8.88264 10  + 376.991 s + 1. s

AHlp = 20*Log[10,Abs[Hlp] /. s -> N[I*2*Pi*f]];
AHhp = 20*Log[10,Abs[Hhp] /. s -> N[I*2*Pi*f]];
AHbp = 20*Log[10,Abs[Hbp] /. s -> N[I*2*Pi*f]];
Plot[{AHlp, AHhp, AHbp}
, {f, 100, 4000}
,PlotRange -> {-30,40}
,PlotStyle -> {Dashing[{}],
               Dashing[{.02}],
               Dashing[{.04}]}
, AxesLabel -> {"f (Hz)","20*log M(f)"}
];
```


20*log M(f)
40
30
20
10
-10
-20
-30
1000
2000
3000
4000
f (Hz)

A.7 Advanced Analog Filter Design Case Studies

Miroslav D. Lutovac, Dejan V. Tosic and Brian L. Evans
lutovac@iritel.bg.ac.yu tosic@galeb.etf.bg.ac.yu bevans@ece.utexas.edu

A.7.1 References

1. D.V. Tosic, M.D.Lutovac, B.L.Evans,
"Advanced filter design,"
Proc. IEEE Asilomar Conf. Signal, Systems, Computer,
Nov. 1997, pp.710–715.

2. M. D. Lutovac, D. V. Tosic and B. L. Evans,
"Advanced Filter Design for Signal Processing using MATLAB and Mathematica,"
http://iritel.iritel.bg.ac.yu/˜lutovac/www/afdhome.htm
http://galeb.etf.bg.ac.yu/˜tosic/afdhome.htm
http://www.ece.utexas.edu/˜bevans/

A.7.2 Initialization

```
SetDirectory[HomeDirectory[]];
<<afd\math\m\clearall.m
<<graphics‘graphics‘
```

A.7.3 Notation

a – selectivity factor
ap – maximum passband loss, dB, of realized filter
Ap – maximum passband loss, dB, in specification
as – minimum stopband loss, dB, of realized filter
As – minimum stopband loss, dB, in specification
A2a(n,Ap,As) – minimum selectivity factor from attenuation spec
A2K(A) – characteristic function in terms of attenuation in dB
e – ripple factor
fp – passband edge [Hz] of realized filter
Fp – passband edge [Hz] in specification
fs – stopband edge [Hz] of realized filter
Fs – stopband edge [Hz] in specification
Ke(n,a,e,x) – characteristic function
L(n,a) – discrimination factor
n – order
nbut(Fp,Fs,Ap,As) minimum Butterworth order from specification
ncheb(Fp,Fs,Ap,As) minimum Chebyshev order from specification
nellip(Fp,Fs,Ap,As) minimum order from specification
q(k) – modular constant
R(n,a,x) – elliptic rational function
S(n,a,e) – list of transfer function poles
S(n,a,e,i) – transfer function pole

SA – attenuation specification
SK – characterictic function specification
X(n,a) – list of zeros of elliptic rational function
X(n,a,i) – zero of elliptic rational function
Z(n,a,e) – zeta function

A.7.4 Definitions and Procedures

```
X[n_Integer,a_,i_Integer] :=  -JacobiCD[
 (2*i-1)*EllipticK[1/a^2]/n, 1/a^2
];
X[n_Integer,a_,i_Integer] := 0 /; And[i==(n+1)/2,OddQ[n]];
X[n_Integer, a_] :=  X[n,a,#]& /@ Range[n];
L[n_Integer,a_] := Block[{i,r},
 If[EvenQ[n],
  r = (1/a^n)*Product[(a^2 - X[n,a,i]^2)^2, {i,n/2}]/
             Product[(1   - X[n,a,i]^2)^2, {i,n/2}];,
  r = (1/a^(n-2))*Product[(a^2 - X[n,a,i]^2)^2, {i,(n-1)/2}]/
                 Product[(1   - X[n,a,i]^2)^2, {i,(n-1)/2}];
 ];
 r
];
R[n_Integer,a_,x_] := Block[{i,r,r0},
 If[EvenQ[n],
  r = Product[x^2 -     X[n,a,i]^2, {i,n/2}]/
      Product[x^2 - a^2/X[n,a,i]^2, {i,n/2}];
  r0 = Product[1 -     X[n,a,i]^2, {i,n/2}]/
       Product[1 - a^2/X[n,a,i]^2, {i,n/2}];,
  r = x*Product[x^2 -     X[n,a,i]^2, {i,(n-1)/2}]/
        Product[x^2 - a^2/X[n,a,i]^2, {i,(n-1)/2}];
  r0 = Product[1 -     X[n,a,i]^2, {i,(n-1)/2}]/
       Product[1 - a^2/X[n,a,i]^2, {i,(n-1)/2}];
 ];
 r/r0
];
Z[n_Integer,a_,e_] :=  JacobiSN[
 InverseJacobiSN[1/Sqrt[1+e^2],1-1/(L[n,a])^2]*
 EllipticK[1-1/a^2]/EllipticK[1-1/(L[n,a])^2],1-1/a^2
];
S[n_Integer, a_, e_, i_Integer] := Block[
{den,num,numim,numre,x,z},
 x = X[n,a,i];
 z = Z[n,a,e];
 numre = -z*Sqrt[1 - z^2]*Sqrt[1 - x^2]*Sqrt[1 - x^2/a^2];
 numim = x*Sqrt[1 - (1-1/a^2)*z^2];
 num = numre + I*numim;
```

```
 den = 1 - (1 - x^2/a^2)*z^2;
 num/den
];
S[n_Integer, a_, e_] :=  S[n,a,e,#]& /@ Range[n];
A2K[A_] := Sqrt[1 - 10^(-A/10)]/10^(-A/20);
Ke[n_Integer, a_, e_, x_] := e*Abs[R[n,a,x]];
```

A.7.5 Specification

```
SA = {3000., 3225., 0.2, 40.};
{Fp, Fs, Ap, As} = SA;
Kp = A2K[Ap];
Ks = A2K[As];
SK = {Fp, Fs, Kp, Ks}

{3000., 3225., 0.217091, 99.995}
```

A.7.6 Minimum Order

```
nellip[Fp_,Fs_,Ap_,As_] := Block[
 {num, den,
  k = Fp/Fs,
  L = Sqrt[(-1 + 10^(As/10))/(-1 + 10^(Ap/10))]},
 num = EllipticK[1-1/L^2]/EllipticK[1/L^2];
 den = EllipticK[1-k^2]/EllipticK[k^2];
 Ceiling[num/den//N]
];
ncheb[Fp_,Fs_,Ap_,As_] := Block[
 {L = Sqrt[(-1 + 10^(As/10))/(-1 + 10^(Ap/10))],
  aspec = Fs/Fp},
 Ceiling[ArcCosh[L]/ArcCosh[aspec]//N]
];
nbut[Fp_,Fs_,Ap_,As_] := Block[
 {L = Sqrt[(-1 + 10^(As/10))/(-1 + 10^(Ap/10))],
  aspec = Fs/Fp},
 Ceiling[Log[10,L]/Log[10,aspec]//N]
];
{nellip[Fp,Fs,Ap,As], ncheb[Fp,Fs,Ap,As], nbut[Fp,Fs,Ap,As]}
nmin = nellip[Fp,Fs,Ap,As];
nmax = 2*nmin;
nlist = Range[nmin,nmax]

{8, 18, 85}
{8, 9, 10, 11, 12, 13, 14, 15, 16}
```

■ A.7.7 Range of Selectivity Factor and Ripple Factor

```
q[k_] := Block[{c,e,r,s,t},
 If[k<=1/Sqrt[2.0],
   t = (1/2)*(1 - (1-k^2)^(1/4))/(1 + (1-k^2)^(1/4));,
   t = (1/2)*(1 - Sqrt[k])/(1 + Sqrt[k]);
 ];
 e = {1,5, 9, 13,  17,   21,    25,     29,      33,       37};
 c = {1,2,15,150,1707,20910,268616,3567400,48555069,673458874};
 s = Sum[c[[i]]*(t^e[[i]]),{i,Length[e]}];
 If[k<=1/Sqrt[2.0],
   r = s;,
   r = Exp[Pi^2/Log[s]];
 ];
 N[r]
];
A2a[n_,Ap_,As_] := Block[
 {m, num, den, terms=9, L, qL},
 L = Sqrt[(-1 + 10^(As/10))/(-1 + 10^(Ap/10))];
 qL = q[1/L]^(1/n);
 num = 1 + 2*Sum[(-1)^m*(qL)^(m^2), {m,1,terms}];
 den = 1 + 2*Sum[(qL)^(m^2), {m,1,terms}];
 1/Sqrt[1 - (num/den)^4]
];
amin8  = A2a[nmin,Ap,As];
amax8  = a/. FindRoot[R[nmin,a,Fs/Fp]==Ks/Kp, {a,Fs/Fp,1.1}];
amin9  = A2a[9,Ap,As];
amax9  = a/. FindRoot[R[9,a,Fs/Fp]==Ks/Kp, {a,amax8,1.1}];
amin10 = A2a[10,Ap,As];
amax10 = a/. FindRoot[R[10,a,Fs/Fp]==Ks/Kp, {a,amax9,1.2}];
amin11 = A2a[11,Ap,As];
amax11 = a/. FindRoot[R[11,a,Fs/Fp]==Ks/Kp, {a,amax10,1.2}];
amin12 = A2a[12,Ap,As];
amax12 = a/. FindRoot[R[12,a,Fs/Fp]==Ks/Kp, {a,amax11,1.2}];
amin13 = A2a[13,Ap,As];
amax13 = a/. FindRoot[R[13,a,Fs/Fp]==Ks/Kp, {a,amax12,1.3}];
amin14 = A2a[14,Ap,As];
amax14 = a/. FindRoot[R[14,a,Fs/Fp]==Ks/Kp, {a,amax13,1.4}];
amin15 = A2a[15,Ap,As];
amax15 = a/. FindRoot[R[15,a,Fs/Fp]==Ks/Kp, {a,amax14,1.6}];
amin16 = A2a[16,Ap,As];
amax16 = a/. FindRoot[R[16,a,Fs/Fp]==Ks/Kp, {a,amax15,1.9}];
aminlist = {amin8,amin9,amin10,amin11,amin12,amin13,amin14,amin15,amin16};
amaxlist = {amax8,amax9,amax10,amax11,amax12,amax13,amax14,amax15,amax16};
eminlist = Table[Ks/L[n,amaxlist[[n-8+1]]], {n,nmin,nmax}];
{emin8,emin9,emin10,emin11,emin12,emin13,emin14,emin15,emin16} = eminlist;
emax = Kp;
emaxlist = Table[Kp,{nmax-nmin+1}];
```

```
TableForm[Transpose[{nlist,aminlist,amaxlist,eminlist,emaxlist}]
, TableHeadings->{{},{"n","amin","amax","emin","emax"}}]
```

n	amin	amax	emin	emax
8	1.04285	1.08323	0.0757872	0.217091
9	1.022	1.09807	0.0184689	0.217091
10	1.01135	1.12013	0.00368669	0.217091
11	1.00587	1.15172	0.000578651	0.217091
12	1.00304	1.19663	0.0000676012	0.217091
13	1.00158	1.26158	$5.39161\ 10^{-6}$	0.217091
14	1.00082	1.35951	$2.53319\ 10^{-7}$	0.217091
15	1.00042	1.51914	$5.28479\ 10^{-9}$	0.217091
16	1.00022	1.82219	$2.51893\ 10^{-11}$	0.217091

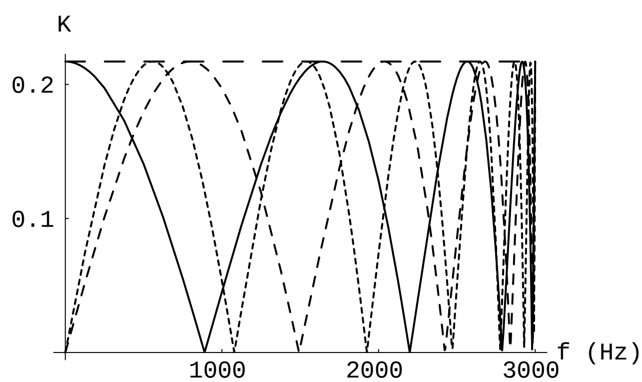
A.7.8 Range of Edge Frequencies

```
fpminlist = Fs / amaxlist;
{fpmin8,fpmin9,fpmin10,fpmin11,fpmin12,fpmin13,
 fpmin14,fpmin15,fpmin16} = fpminlist;
fpmaxlist = Fs / aminlist;
{fpmax8,fpmax9,fpmax10,fpmax11,fpmax12,fpmax13,
 fpmax14,fpmax15,fpmax16} = fpmaxlist;
fsminlist = Fp * aminlist;
{fsmin8,fsmin9,fsmin10,fsmin11,fsmin12,fsmin13,
 fsmin14,fsmin15,fsmin16} = fsminlist;
fsmaxlist = Fp * amaxlist;
{fsmax8,fsmax9,fsmax10,fsmax11,fsmax12,fsmax13,
 fsmax14,fsmax15,fsmax16} = fsmaxlist;
TableForm[Transpose[{nlist,fpminlist,fpmaxlist,fsminlist,fsmaxlist}]
, TableHeadings->{{},{"n","fpmin","fpmax","fsmin","fsmax (Hz)"}}]
```

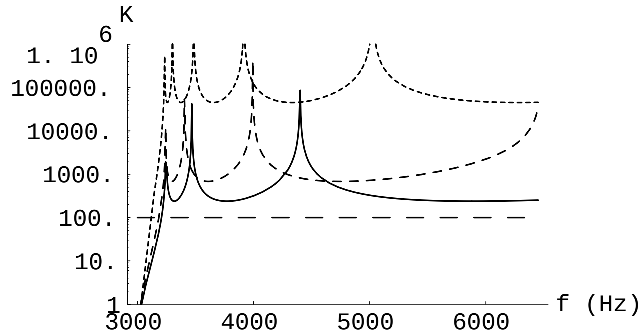
n	fpmin	fpmax	fsmin	fsmax (Hz)
8	2977.2	3092.49	3128.55	3249.7
9	2936.97	3155.57	3066.01	3294.21
10	2879.14	3188.79	3034.06	3360.38
11	2800.16	3206.17	3017.62	3455.16
12	2695.06	3215.22	3009.13	3589.89
13	2556.32	3219.92	3004.73	3784.74
14	2372.18	3222.36	3002.45	4078.53
15	2122.91	3223.63	3001.27	4557.42
16	1769.85	3224.29	3000.66	5466.56

■ A.7.9 Design D1

```
Plot[Evaluate[{Kp
              , Ke[ 8, Fs/Fp, emax, f/Fp]
              , Ke[ 9, Fs/Fp, emax, f/Fp]
              , Ke[13, Fs/Fp, emax, f/Fp]
             }]
, {f,0,Fp}
, AxesLabel -> {"f (Hz)", "K"}
, PlotStyle -> {Dashing[{0.04}],
                Dashing[{}],
                Dashing[{0.02}],
                Dashing[{0.01}]
               }
, Ticks -> {{1000, 2000, 3000}, {0, 0.1, 0.2}}
, PlotRange -> All];
```

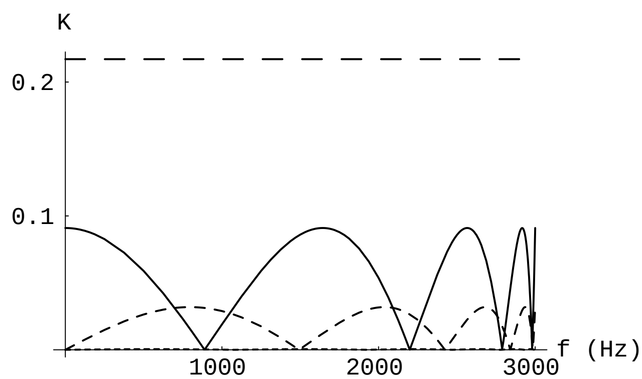



```
LogPlot[Evaluate[{Ks
              , Ke[ 8, Fs/Fp, emax, f/Fp]
              , Ke[ 9, Fs/Fp, emax, f/Fp]
              , Ke[13, Fs/Fp, emax, f/Fp]
             }]
, {f,Fp,2*Fs}
, AxesLabel -> {"f (Hz)", "K"}
, PlotStyle -> {Dashing[{0.04}],
                Dashing[{}],
                Dashing[{0.02}],
                Dashing[{0.01}]
               }
, Ticks -> {{3000, 4000, 5000, 6000}, Automatic}
, PlotRange -> {1,10^6}];
```



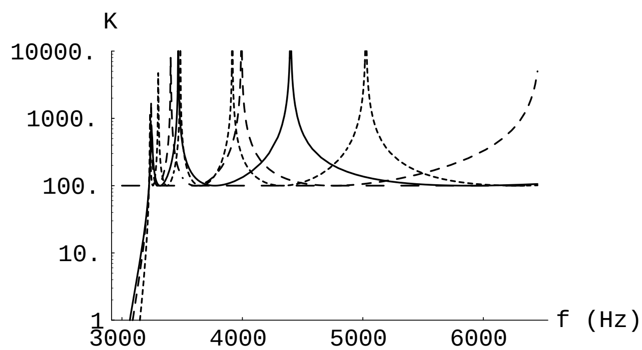
■ A.7.10 Design D2

```
Plot[Evaluate[{Kp
              , Ke[ 8, Fs/Fp, Ks/L[ 8,Fs/Fp], f/Fp]
              , Ke[ 9, Fs/Fp, Ks/L[ 9,Fs/Fp], f/Fp]
              , Ke[13, Fs/Fp, Ks/L[13,Fs/Fp], f/Fp]
             }]
, {f,0,Fp}
, AxesLabel -> {"f (Hz)", "K"}
, PlotStyle -> {Dashing[{0.04}],
                Dashing[{}],
                Dashing[{0.02}],
                Dashing[{0.01}]
               }
, Ticks -> {{1000, 2000, 3000}, {0, 0.1, 0.2}}
, PlotRange -> All];
```



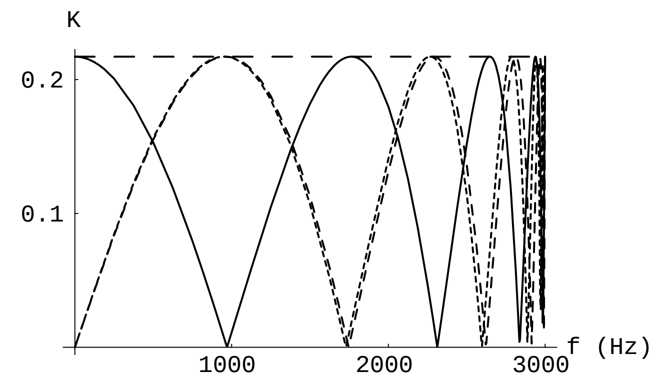

```
LogPlot[Evaluate[{Ks
              , Ke[ 8, Fs/Fp, Ks/L[ 8,Fs/Fp], f/Fp]
              , Ke[ 9, Fs/Fp, Ks/L[ 9,Fs/Fp], f/Fp]
              , Ke[13, Fs/Fp, Ks/L[13,Fs/Fp], f/Fp]
             }]
, {f,Fp,2*Fs}
```

```
, AxesLabel -> {"f (Hz)", "K"}
, PlotStyle -> {Dashing[{0.04}],
               Dashing[{}],
               Dashing[{0.02}],
               Dashing[{0.01}]
              }
, Ticks -> {{3000, 4000, 5000, 6000}, Automatic}
, PlotRange -> {1,10^4}];
```

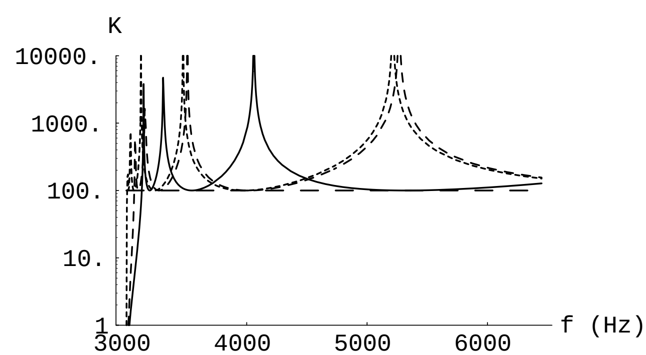


■ A.7.11 Design D3a

```
Plot[Evaluate[{Kp
             , Ke[ 8, amin8,  emax, f/Fp]
             , Ke[ 9, amin9,  emax, f/Fp]
             , Ke[13, amin13, emax, f/Fp]
            }]
, {f,0,Fp}
, AxesLabel -> {"f (Hz)", "K"}
, PlotStyle -> {Dashing[{0.04}],
               Dashing[{}],
               Dashing[{0.02}],
               Dashing[{0.01}]
              }
, Ticks -> {{1000, 2000, 3000}, {0, 0.1, 0.2}}
, PlotRange -> All];
```

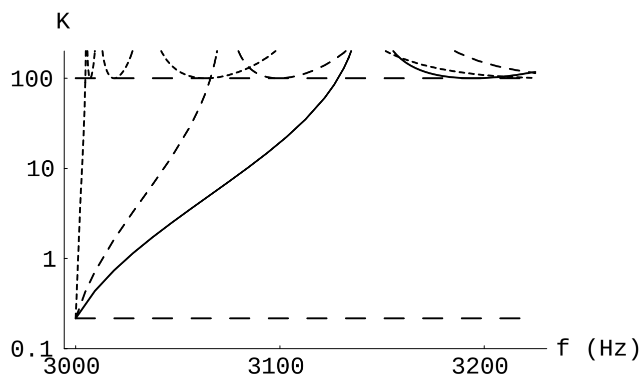



```
LogPlot[Evaluate[{Ks
              , Ke[ 8, amin8,  emax, f/Fp]
              , Ke[ 9, amin9,  emax, f/Fp]
              , Ke[13, amin13, emax, f/Fp]
             }]
, {f,Fp,2*Fs}
, AxesLabel -> {"f (Hz)", "K"}
, PlotStyle -> {Dashing[{0.04}],
                Dashing[{}],
                Dashing[{0.02}],
                Dashing[{0.01}]
               }
, Ticks -> {{3000, 4000, 5000, 6000}, Automatic}
, PlotRange -> {1,10^4}];
```



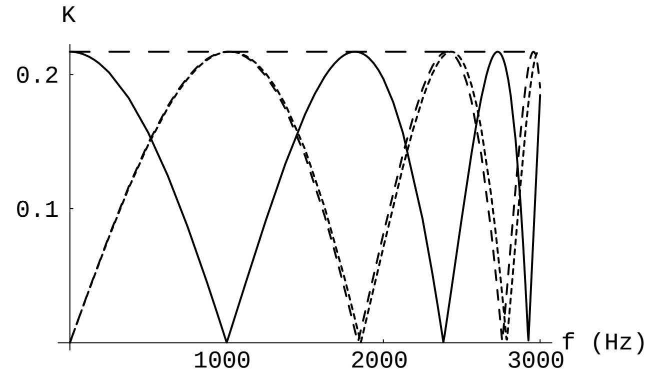

```
LogPlot[Evaluate[{Kp,Ks
              , Ke[ 8, amin8,  emax, f/Fp]
              , Ke[ 9, amin9,  emax, f/Fp]
              , Ke[13, amin13, emax, f/Fp]
             }]
, {f,Fp,Fs}
, AxesLabel -> {"f (Hz)", "K"}
```

```
, PlotStyle -> {Dashing[{0.04}],
               Dashing[{0.04}],
               Dashing[{}],
               Dashing[{0.02}],
               Dashing[{0.01}]
              }
, Ticks -> {{3000, 3100, 3200}, {0.1, 1, 10, 100}}
, PlotRange -> {0.1,200}];
```

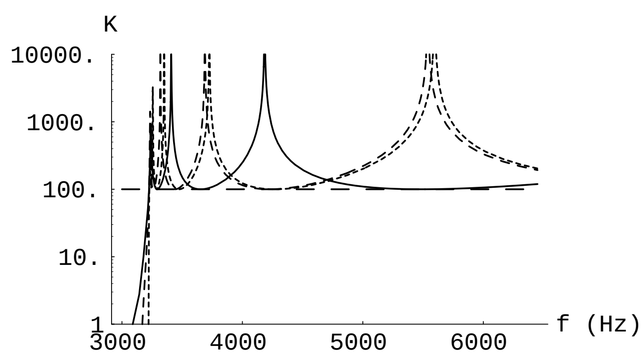


■ A.7.12 Design D3b

```
Plot[Evaluate[{Kp
              , Ke[ 8, amin8,  emax, f/fpmax8]
              , Ke[ 9, amin9,  emax, f/fpmax9]
              , Ke[13, amin13, emax, f/fpmax13]
             }]
, {f,0,Fp}
, AxesLabel -> {"f (Hz)", "K"}
, PlotStyle -> {Dashing[{0.04}],
               Dashing[{}],
               Dashing[{0.02}],
               Dashing[{0.01}]
              }
, Ticks -> {{1000, 2000, 3000}, {0, 0.1, 0.2}}
, PlotRange -> All];
```

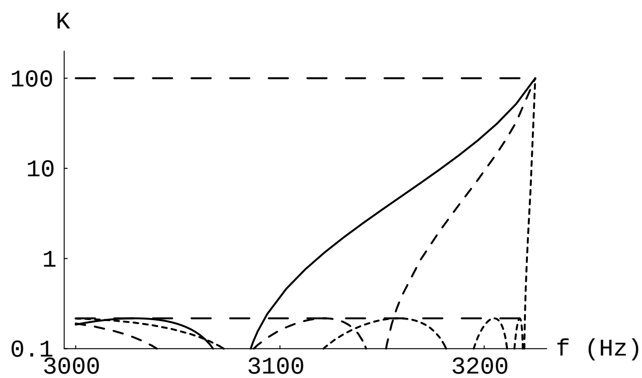



```
LogPlot[Evaluate[{Ks
              , Ke[ 8, amin8,  emax, f/fpmax8]
              , Ke[ 9, amin9,  emax, f/fpmax9]
              , Ke[13, amin13, emax, f/fpmax13]
             }]
, {f,Fp,2*Fs}
, AxesLabel -> {"f (Hz)", "K"}
, PlotStyle -> {Dashing[{0.04}],
                Dashing[{}],
                Dashing[{0.02}],
                Dashing[{0.01}]
               }
, Ticks -> {{3000, 4000, 5000, 6000}, Automatic}
, PlotRange -> {1,10^4}];
```



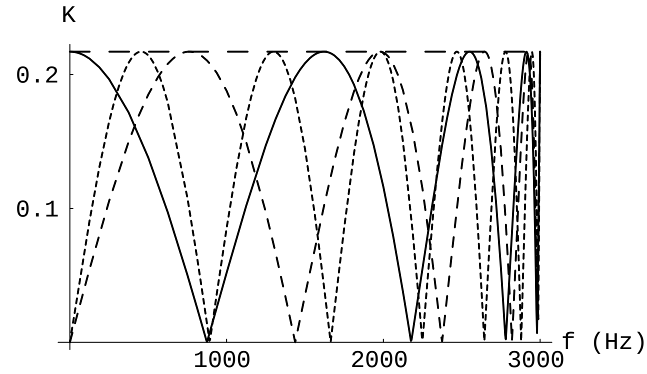

```
LogPlot[Evaluate[{Kp,Ks
              , Ke[ 8, amin8,  emax, f/fpmax8]
              , Ke[ 9, amin9,  emax, f/fpmax9]
              , Ke[13, amin13, emax, f/fpmax13]
             }]
, {f,Fp,Fs}
, AxesLabel -> {"f (Hz)", "K"}
```

```
, PlotStyle -> {Dashing[{0.04}],
               Dashing[{0.04}],
               Dashing[{}],
               Dashing[{0.02}],
               Dashing[{0.01}]
              }
, Ticks -> {{3000, 3100, 3200}, {0.1, 1, 10, 100}}
, PlotRange -> {0.1,200}];
```

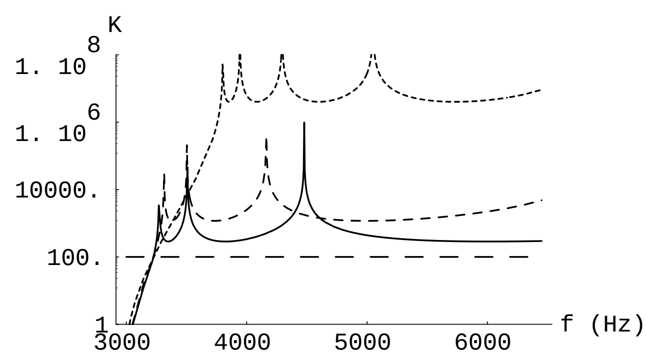


■ A.7.13 Design D4a

```
Plot[Evaluate[{Kp
              , Ke[ 8, amax8,  emax, f/Fp]
              , Ke[ 9, amax9,  emax, f/Fp]
              , Ke[13, amax13, emax, f/Fp]
             }]
, {f,0,Fp}
, AxesLabel -> {"f (Hz)", "K"}
, PlotStyle -> {Dashing[{0.04}],
               Dashing[{}],
               Dashing[{0.02}],
               Dashing[{0.01}]
              }
, Ticks -> {{1000, 2000, 3000}, {0, 0.1, 0.2}}
, PlotRange -> All];
```



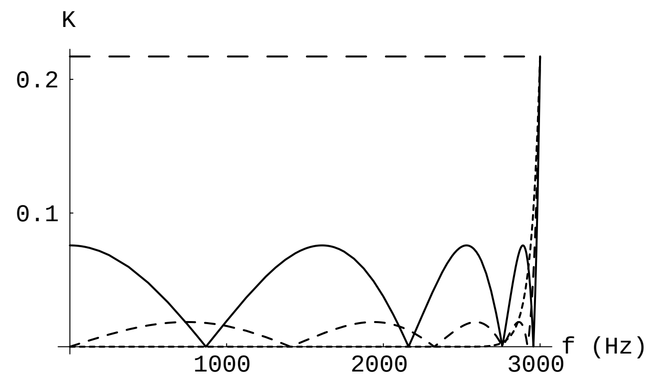

```
LogPlot[Evaluate[{Ks
              , Ke[ 8, amax8,  emax, f/Fp]
              , Ke[ 9, amax9,  emax, f/Fp]
              , Ke[13, amax13, emax, f/Fp]
             }]
, {f,Fp,2*Fs}
, AxesLabel -> {"f (Hz)", "K"}
, PlotStyle -> {Dashing[{0.04}],
                Dashing[{}],
                Dashing[{0.02}],
                Dashing[{0.01}]
               }
, Ticks -> {{3000, 4000, 5000, 6000}, Automatic}
, PlotRange -> {1,10^8}];
```



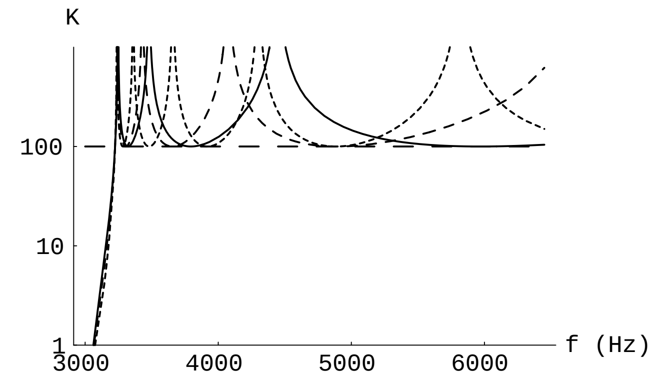
■ A.7.14 Design D4b

```
Plot[Evaluate[{Kp
              , Ke[ 8, amax8,  emin8,  f/(Fs/amax8)]
              , Ke[ 9, amax9,  emin9,  f/(Fs/amax9)]
              , Ke[13, amax13, emin13, f/(Fs/amax13)]
             }]
, {f,0,Fp}
```

```
, AxesLabel -> {"f (Hz)", "K"}
, PlotStyle -> {Dashing[{0.04}],
               Dashing[{}],
               Dashing[{0.02}],
               Dashing[{0.01}]
              }
, Ticks -> {{1000, 2000, 3000}, {0, 0.1, 0.2}}
, PlotRange -> All];
```




```
LogPlot[Evaluate[{Ks
              , Ke[ 8, amax8,  emin8,  f/(Fs/amax8)]
              , Ke[ 9, amax9,  emin9,  f/(Fs/amax9)]
              , Ke[13, amax13, emin13, f/(Fs/amax13)]
             }]
, {f,Fp,2*Fs}
, AxesLabel -> {"f (Hz)", "K"}
, PlotStyle -> {Dashing[{0.04}],
               Dashing[{}],
               Dashing[{0.02}],
               Dashing[{0.01}]
              }
, Ticks -> {{3000, 4000, 5000, 6000}, {1,10,100}}
, PlotRange -> {1,10^3}];
```


K
100
10
1
3000
4000
5000
6000
f (Hz)

A.8 Classical Digital Filters

Transpose Direct Form II

IIR Second-Order Realization

Miroslav D. Lutovac, Dejan V. Tosic and Brian L. Evans

lutovac@iritel.bg.ac.yu tosic@galeb.etf.bg.ac.yu bevans@ece.utexas.edu

A.8.1 References

1. Alan Oppenheim, Ronald Schafer, "Digital Signal Processing,"
Prentice-Hall, Englewood Cliffs, New Jersey, 1975.

2. Sanjit Mitra, James Kaiser, "Handbook for Digital Signal Processing,"
John Wiley, New York, 1993, pp. 127–128.

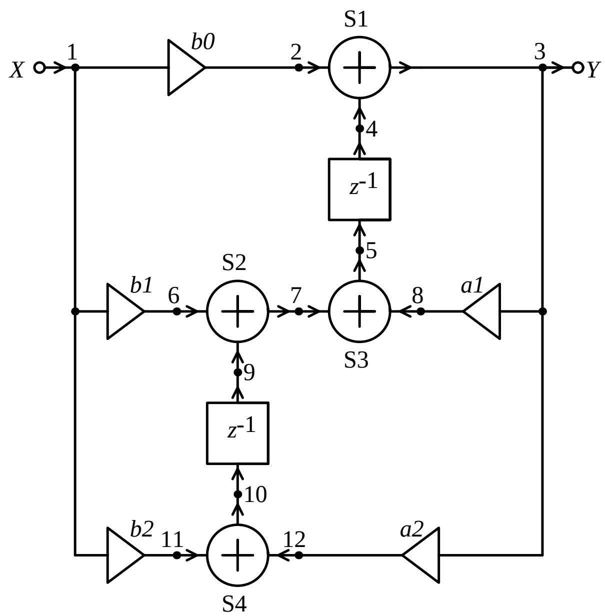
3. M. D. Lutovac, D. V. Tosic and B. L. Evans,
"Advanced Filter Design for Signal Processing using MATLAB and Mathematica,"
http://iritel.iritel.bg.ac.yu/~lutovac/www/afdhome.htm
http://galeb.etf.bg.ac.yu/~tosic/afdhome.htm
http://www.ece.utexas.edu/~bevans/

A.8.2 Initialization

```
SetDirectory[HomeDirectory[]];
<<afd\math\m\clearall.m
<<afd\math\m\drawdfil.m
<<afd\math\m\drawiirf.m
```

A.8.3 Block Diagram

```
DrawTDF2[0,0,1,1/0.8,10];
```



A.8.4 Analysis (v=1/z)

```
ElementEquations = {
  Y1 == X
, Y2 == b0*Y1 + Xb0
, Y3 == Y2 + Y4
, Y4 == v*Y5
, Y5 == Y7 + Y8
, Y6 == b1*Y1 + Xb1
, Y7 == Y6 + Y9
, Y8 == a1*Y3 + Xa1
, Y9 ==  v*Y10
, Y10 == Y11 + Y12
, Y11 == b2*Y1 + Xb2
, Y12 == a2*Y3 + Xa2
};
NodeSignals = {Y1,Y2,Y3,Y4,Y5,Y6,Y7,Y8,Y9,Y10,Y11,Y12};
Response = Flatten[Solve[ElementEquations,NodeSignals]];
Y = Together[Y3/.Response];
Hv = Y /. {X -> 1, Xa1 -> 0, Xa2 -> 0, Xb0 -> 0, Xb1 -> 0, Xb2 -> 0};
Hz = Hv /. v -> 1/z //Together;
Print["H(v) = ", Hv]
Print["H(z) = ", Hz]
```

```
                       2
       -b0 - b1 v - b2 v
H(v) = ------------------
                      2
       -1 + a1 v + a2 v
                       2
       -b2 - b1 z - b0 z
H(z) = ------------------
                     2
         a2 + a1 z - z
```

■ A.8.5 Transfer Function and Noise Transfer Functions

```
Ha1v = Y /. {X -> 0, Xa1 -> 1, Xa2 -> 0, Xb0 -> 0, Xb1 -> 0, Xb2 -> 0};
Ha2v = Y /. {X -> 0, Xa1 -> 0, Xa2 -> 1, Xb0 -> 0, Xb1 -> 0, Xb2 -> 0};
Hb0v = Y /. {X -> 0, Xa1 -> 0, Xa2 -> 0, Xb0 -> 1, Xb1 -> 0, Xb2 -> 0};
Hb1v = Y /. {X -> 0, Xa1 -> 0, Xa2 -> 0, Xb0 -> 0, Xb1 -> 1, Xb2 -> 0};
Hb2v = Y /. {X -> 0, Xa1 -> 0, Xa2 -> 0, Xb0 -> 0, Xb1 -> 0, Xb2 -> 1};
H = Collect[Numerator[Hv],v]/Collect[Denominator[Hv],v];
Ha1 = Collect[Numerator[Ha1v],v]/Collect[Denominator[Ha1v],v];
Ha2 = Collect[Numerator[Ha2v],v]/Collect[Denominator[Ha2v],v];
Hb0 = Collect[Numerator[Hb0v],v]/Collect[Denominator[Hb0v],v];
Hb1 = Collect[Numerator[Hb1v],v]/Collect[Denominator[Hb1v],v];
Hb2 = Collect[Numerator[Hb2v],v]/Collect[Denominator[Hb2v],v];
v2invz = {v->z^"-1", v^2->z^("-2"), v^3->z^("-3"), v^4->z^("-4")};
Print["H(z) = ", H /. v2invz ]
Print["Ha1(z) = ", Ha1 /. v2invz ]
Print["Ha2(z) = ", Ha2 /. v2invz ]
Print["Hb0(z) = ", Hb0 /. v2invz ]
Print["Hb1(z) = ", Hb1 /. v2invz ]
Print["Hb2(z) = ", Hb2 /. v2invz ]
```

```
                 -1       -2
       -b0 - b1 z   - b2 z
H(z) = ---------------------
                -1       -2
       -1 + a1 z   + a2 z
                     -1
                    z
Ha1(z) = -(--------------------)
                     -1       -2
           -1 + a1 z   + a2 z
                     -2
                    z
Ha2(z) = -(--------------------)
                     -1       -2
           -1 + a1 z   + a2 z
```

```
                   1
Hb0(z) = -(--------------------)
                   -1       -2
          -1 + a1 z    + a2 z
                   -1
                  z
Hb1(z) = -(--------------------)
                   -1       -2
          -1 + a1 z    + a2 z
                   -2
                  z
Hb2(z) = -(--------------------)
                   -1       -2
          -1 + a1 z    + a2 z
```

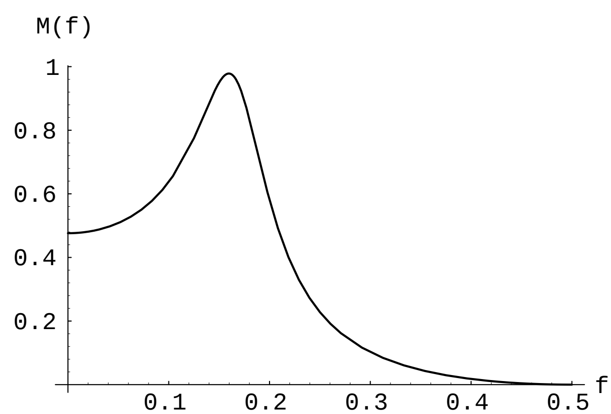
A.8.6 VQNR Variance of Quantization Noise Due to Rounding the Output of the Multiplier

```
VQNR[H_,z_Symbol,a_Symbol,b_Symbol
    ,r_Symbol,theta_Symbol] := Module[
   {ax, bx, d0, d1, d2, denH2, H0, H0inv, H2, numH2
   ,res0, res1, res2, rtheta2ab, sumres, var=Infinity
   ,z1, z2},
   H0 = Together[H] /. {a -> ax, b -> bx};
   H0inv = Together[H0 /. z->1/z];
   H2 = Together[Cancel[z^2*H0]];
   numH2 = Collect[Numerator[H2],z];
   denH2 = Collect[Denominator[H2],z];
   If[Exponent[denH2,z] == 2
   ,{d0,d1,d2} = CoefficientList[denH2,z];
    {z1,z2} = Flatten[Solve[denH2==0,z]];
    res0 = D[H2*H0inv,{z,2}]/2 /. z->0;
    res1 = numH2*H0inv/(d2*(z-(z/.z2))*z^3) /. z1;
    res2 = numH2*H0inv/(d2*(z-(z/.z1))*z^3) /. z2;
    sumres = Simplify[res0 + res1 + res2];
    rtheta2ab = Solve[{d0 == d2*(r^2),
                      d1 == d2*(-2*r*Cos[theta])}
                ,{ax,bx}] //Flatten;
    var = Together[sumres /. rtheta2ab];
   ,Print["Error in denominator!   ", denH2];
   ];
 var]
Haz = Together[Ha2 /. v -> 1/z]
vara = Simplify[VQNR[Haz,z,a1,a2,r,theta], Trig -> True]
```

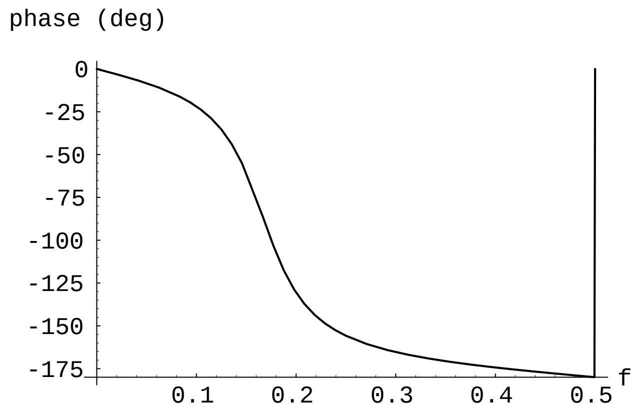
```
       1
-(-------------)
              2
  a2 + a1 z - z
                     2
              -1 - r
-------------------------------------
      2        4      2
(-1 + r ) (1 + r  - 2 r  Cos[2 theta])
```

■ A.8.7 Frequency Response (z=Exp[I*2*Pi*f])

```
values = {a1 -> 0.8, a2 -> -.64, b0 -> 1/10, b1 -> 2/10, b2 -> 1/10};
Hf = Hz /. z->Exp[I*2*Pi*f] /. values //N;
Plot[Evaluate[Abs[Hf]]
, {f, 0, 1/2}
, AxesLabel -> {"f", "M(f)"}
];
```

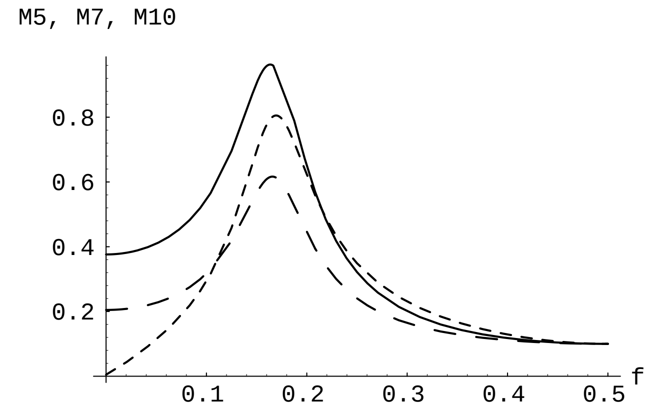



```
Plot[(Evaluate[Arg[Hf]])/Degree
, {f, 0, 1/2}
, AxesLabel -> {"f", "phase (deg)"}
, AxesOrigin -> {0, -180}
, PlotRange -> All
];
```



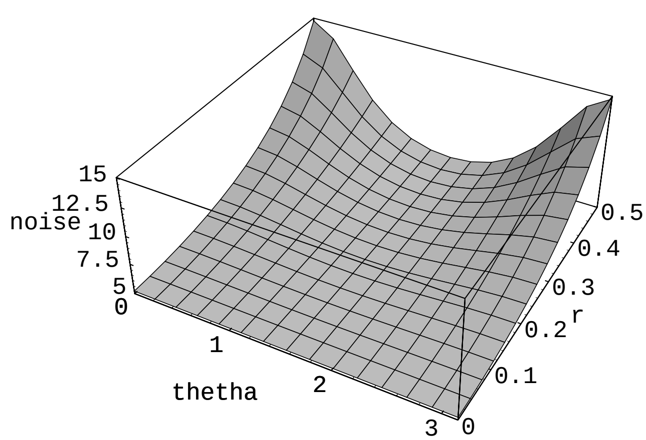
A.8.8 Dynamics
Magnitude Response of Partial Transfer Functions

```
excitation = {X -> 1, Xa1 -> 0, Xa2 -> 0, Xb0 -> 0, Xb1 -> 0, Xb2 -> 0};
values = {a1 -> 0.8, a2 -> -.64, b0 -> 1/10, b1 -> 2/10, b2 -> 1/10};
H7 = Together[Y7/.Response] /. excitation;
H7z = H7 /. v -> 1/z //Together;
H7f = H7z /. z->Exp[I*2*Pi*f] /. values //N;
H5 = Together[Y5/.Response] /. excitation;
H5z = H5 /. v -> 1/z //Together;
H5f = H5z /. z->Exp[I*2*Pi*f] /. values //N;
H10 = Together[Y10/.Response] /. excitation;
H10z = H10 /. v -> 1/z //Together;
H10f = H10z /. z->Exp[I*2*Pi*f] /. values //N;
Plot[Evaluate[
  {Abs[H5f], Abs[H7f], Abs[H10f]}]
, {f, 0, 1/2}
, AxesLabel -> {"f", "M5, M7, M10"}
, PlotRange -> All
, PlotStyle -> {Dashing[{}]
              , Dashing[{.02}]
              , Dashing[{.04}]
}
];
```



■ A.8.9 Relative Output Noise as a Function of Pole Position Due to All 5 Multipliers

```
valuesb = {b0 -> 1/10, b1 -> 2/10, b2 -> 1/10};
Plot3D[5*(vara /. valuesb)
, {theta, 0, Pi}
, {r, 0, 0.5}
, AxesLabel -> {"thetha", "r", "noise"}
, PlotRange -> All];
```

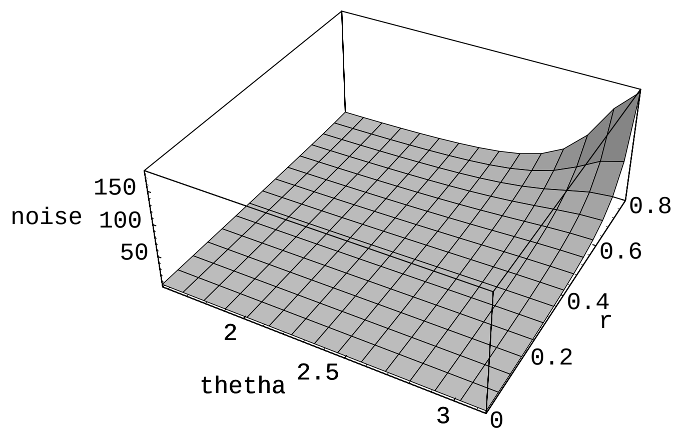


■ Remark

In this example all multipliers have equal noise variance, i.e. sum of variances is equal to 6*vara, where vara represents the noise variance due to the multiplier a2.

```
valuesb = {b0 -> 1/10, b1 -> 2/10, b2 -> 1/10};
Plot3D[5*(vara /. valuesb)
, {theta, Pi/2, Pi}
, {r, 0, .8}
```

```
, AxesLabel -> {"thetha", "r", "noise"}
, PlotRange -> All
(*, Ticks -> {{Pi/2,Pi},{0,1},{1,2,3,4}}*)
];
```



A.8.10 Filtering and Quantization

Filtering without Quantization

```
IIRTDF2[a1_, a2_, b0_, b1_, b2_, X_, d1_:0, d2_:0] := Module[
{vY5, vY10, Y, Y1, Y2, Y3, Y4, Y5, Y6, Y7, Y8, Y9, Y10, Y11,Y12},
 vY5 = d1;
 vY10 = d2;
 Y1  = X;
 Y4  = vY5;
 Y9  = vY10;
 Y2  = b0*Y1;
 Y6  = b1*Y1;
 Y11 = b2*Y1;
 Y3  = Y2 + Y4;
 Y8  = a1*Y3;
 Y12 = a2*Y3;
 Y7 = Y6 + Y9;
 Y5 = Y7 + Y8;
 Y10 = Y11 + Y12;
 Y = Y3;
{Y,Y5,Y10}];
```

Filtering with Quantization

```
Quantize[x_,b_:8,q_:0] := Module[{},
 If[q == 0
 , y = Round[x*2^b]/2^b;
 , y = Floor[2^nm*x]/2^nm ];
y];
```

```
QIIRTDF2[a1_, a2_, b0_, b1_, b2_, X_, d1_:0, d2_:0, bits_:8, mode_:0] := Module[
{vY5, vY10, Y, Y1, Y2, Y3, Y4, Y5, Y6, Y7, Y8, Y9, Y10, Y11,Y12},
 vY5 = d1;
 vY10 = d2;
 Y1  = X;
 Y4  = vY5;
 Y9  = vY10;
 Y2  = Quantize[b0*Y1, bits, mode];
 Y6  = Quantize[b1*Y1, bits, mode];
 Y11 = Quantize[b2*Y1, bits, mode];
 Y3  = Y2 + Y4;
 Y8  = Quantize[a1*Y3, bits, mode];
 Y12 = Quantize[a2*Y3, bits, mode];
 Y7 = Y6 + Y9;
 Y5 = Y7 + Y8;
 Y10 = Y11 + Y12;
 Y = Y3;
{Y,Y5,Y10}];
```

- **Filtering**

```
filterQIIRTDF2[a1_, a2_, b0_, b1_, b2_, Xdata_List, d1_:0, d2_:0
               , bits_Integer:8, mode_Integer:0] := Module[
{n, Qd1, Qd2, QYdata, QYn, QY3, Xn},
Qd1 = d1;
Qd2 = d2;
QYdata = {};
Do[Xn = Xdata[[n]];
   QY3 = QIIRTDF2[a1, a2, b0, b1, b2, Xn, Qd1, Qd2, bits, mode];
   {QYn, Qd1, Qd2} = QY3;
   AppendTo[QYdata,QYn];
,{n,1,Length[Xdata]}];
{QYdata,Qd1,Qd2}];
```

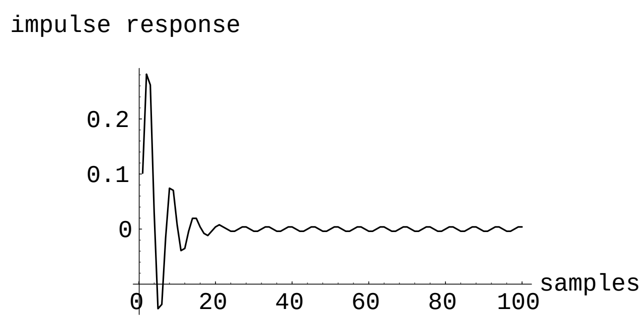
- **Example of Filtering**

```
values = {a1 -> 0.8, a2 -> -.64, b0 -> 0.1, b1 -> 0.2, b2 -> 0.1};
Hv /. values
{aD1,aD2,bD0,bD1,bD2} = {a1,a2,b0,b1,b2} /. values
Ndata = 100;               (* number of samples *)
Xdata = Table[0,{Ndata}]; Xdata[[1]] = 1;
{QYdata,Qd1,Qd2} = filterQIIRTDF2[aD1,aD2,bD0,bD1,bD2, Xdata, 0, 0, 8, 0]//N;
{Ydata,Qd1,Qd2} = filterQIIRTDF2[aD1,aD2,bD0,bD1,bD2, Xdata, 0, 0, 12, 0]//N;
```

```
                     2
-0.1 - 0.2 v - 0.1 v
---------------------
                    2
-1 + 0.8 v - 0.64 v
{0.8, -0.64, 0.1, 0.2, 0.1}
```

Impulse Response

```
ListPlot[QYdata
 , PlotJoined->True
 , PlotRange -> All
 , AxesOrigin -> {.1,-.1}
 , AxesLabel -> {"samples", "impulse response"}
];
```



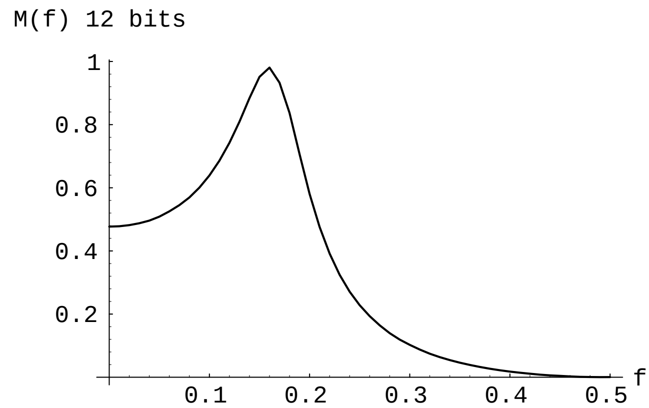
Spectrum of Impulse Response
Define Electrical Engineering FFT

```
EEfft[data_List] := InverseFourier[data]*Sqrt[Length[data]];
EEifft[data_List] := Fourier[data]/Sqrt[Length[data]];
spectrum  = EEfft[Ydata] //Chop;
qspectrum = EEfft[N[QYdata]] //Chop;
```

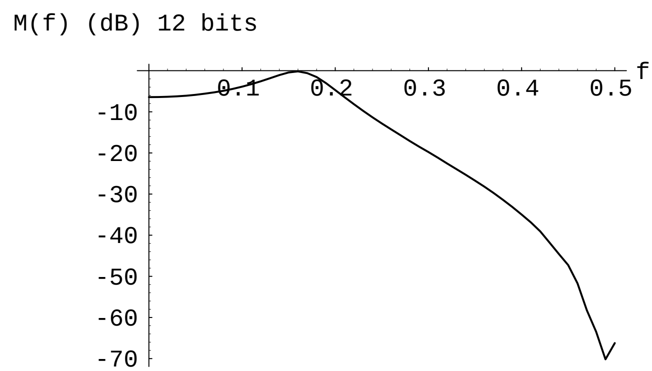
A.8.11 Verify Frequency Response by Spectrum of Impulse Response

12 Bits Filtering Quantization

```
pfs = Table[{(k-1)/Length[spectrum], Abs[spectrum[[k]]]}
            , {k,1,Length[spectrum]/2+1}];
ListPlot[pfs
, PlotJoined -> True
, PlotRange -> All
, AxesLabel -> {"f", "M(f) 12 bits"}
];
```

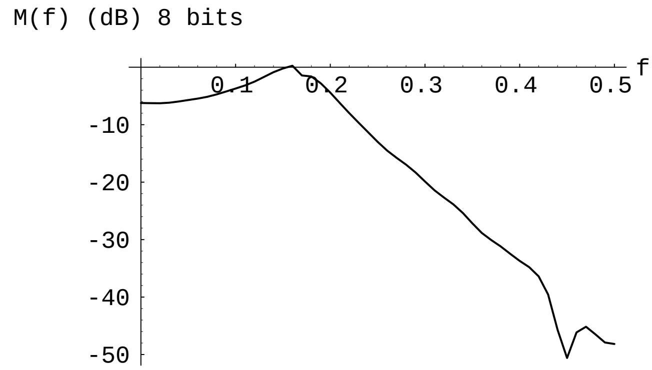



```
pfsdB = Table[{(ind-1)/Length[spectrum], 20*Log[10,Abs[spectrum[[ind]]]]}
              , {ind,1,Length[spectrum]/2+1}];
ListPlot[pfsdB
, PlotJoined -> True
, PlotRange -> All
, AxesLabel -> {"f", "M(f) (dB) 12 bits"}
];
```

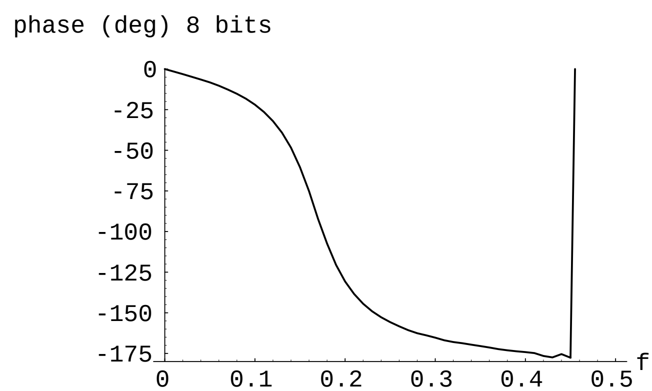


■ **8 Bits Filtering Quantization**

```
pfqsdB = Table[{(ind-1)/Length[qspectrum], 20*Log[10,Abs[qspectrum[[ind]]]]}
              , {ind,1,Length[qspectrum]/2+1}];
ListPlot[pfqsdB
, PlotJoined -> True
, PlotRange -> All
, AxesLabel -> {"f", "M(f) (dB) 8 bits"}
];
```

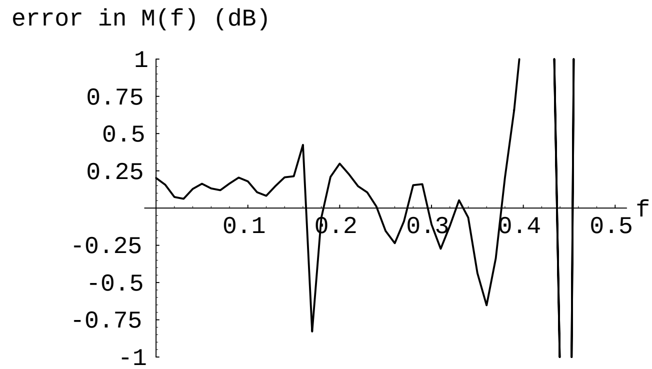



```
pfp = Table[{(ind-1)/Length[spectrum], Arg[spectrum[[ind]]]/Degree}
              , {ind,1,Length[qspectrum]/2+1}];
ListPlot[pfp
, PlotJoined -> True
, PlotRange -> {-180,0}
, AxesOrigin -> {0,-180}
, AxesLabel -> {"f", "phase (deg) 8 bits"}
];
```



Error in Magnitude Response Due to Quantization

```
deltapfs = Table[{(k-1)/Length[spectrum]
       , 20*Log[10,Abs[qspectrum[[k]]]] - 20*Log[10,Abs[spectrum[[k]]]]}
         , {k,1,Length[spectrum]/2+1}];
ListPlot[deltapfs
, PlotJoined -> True
, PlotRange -> {-1,1}
, AxesLabel -> {"f", "error in M(f) (dB)"}
];
```



error in M(f) (dB)
1
0.75
0.5
0.25
-0.25
-0.5
-0.75
-1
0.1
0.2
0.3
0.4
0.5
f

A.9 Advanced Digital Filter Design Case Studies

Miroslav D. Lutovac, Dejan V. Tosic and Brian L. Evans
lutovac@iritel.bg.ac.yu tosic@galeb.etf.bg.ac.yu bevans@ece.utexas.edu

■ A.9.1 References

1. D.V. Tosic, M.D.Lutovac, B.L.Evans,
"Advanced filter design,"
Proc. IEEE Asilomar Conf. Signal, Systems, Computer,
Nov. 1997, pp.710–715.

2. D.V.Tosic, M.D.Lutovac, B.L.Evans,
"Advanced digital filter design,"
Proc. European Conf. Circuit Theory Design ECCTD'99, Stresa, Italy,
Sep.1999,vol.2, pp.1323–1326.

3. M. D. Lutovac, D. V. Tosic and B. L. Evans,
"Advanced Filter Design for Signal Processing using MATLAB and Mathematica,"
http://iritel.iritel.bg.ac.yu/~lutovac/www/afdhome.htm
http://galeb.etf.bg.ac.yu/~tosic/afdhome.htm
http://www.ece.utexas.edu/~bevans/

■ A.9.2 Initialization

```
SetDirectory[HomeDirectory[]];
<<afd\math\m\clearall.m
<<afd\math\m\dfdellip.m
<<graphics`graphics`
```

■ A.9.3 Notation

```
!!afd\math\m\dfdnotat.m
```

a – selectivity factor
AdB – attenuation (dB) in terms of digital frequency f
ap – maximum passband attenuation, (dB), of realized filter
Ap – maximum passband attenuation, (dB), in specification
as – minimum stopband attenuation, (dB), of realized filter
As – minimum stopband attenuation, (dB), in specification
A2a(n,Ap,As) – minimum selectivity factor from attenuation spec
A2K(A) – characteristic function in terms of attenuation in dB
ba – coefficients of the second-order section
e – ripple factor
f – digital frequency (0<f<0.5)
fp – passband edge (0<fp<0.5) of realized filter
Fp – passband edge (0<Fp<0.5) in specification
fs – stopband edge (0<fs<0.5) of realized filter
Fs – stopband edge (0<Fs<0.5) in specification

hz – transfer function in z
Hz(n,a,e,Fp,z) – transfer function in z
Ke(n,a,e,x) – characteristic function
L(n,a) – discrimination factor
n – order
nbut(Fp,Fs,Ap,As) – minimum Butterworth order from specification
ncheb(Fp,Fs,Ap,As) – minimum Chebyshev order from specification
nellip(Fp,Fs,Ap,As) – minimum elliptic order from specification
nminQ – minimum order of elliptic minimal Q-factor design
q(k) – modular constant
Qfactor – Qfactor of the second-order section
rtan(f1,f2) – tan(pi*f1)/tan(pi*f2)
R(n,a,x) – elliptic rational function
S(n,a,e) – list of transfer function poles
S(n,a,e,i) – transfer function pole
SA – attenuation-limit specification
SK – characterictic-function-limit specification
xtan – tan(pi*Fs)/tan(pi*Fp)
X(n,a) – list of zeros of elliptic rational function
X(n,a,i) – zero of elliptic rational function
z – complex variable
Z(n,a,e) – zeta function
Zbl(Spole,Fp) – bilinear transformation

■ A.9.4 Definitions and Procedures

```
!!afd\math\m\dfdellip.m
(* DFDELLIP.M
   7:06PM  9/11/98
*)
A2a[n_,Ap_,As_] := Module[
 {m, num, den, terms=9, L, qL},
 L = Sqrt[(-1 + 10^(As/10))/(-1 + 10^(Ap/10))];
 qL = q[1/L]^(1/n);
 num = 1 + 2*Sum[(-1)^m*(qL)^(m^2), {m,1,terms}];
 den = 1 + 2*Sum[(qL)^(m^2), {m,1,terms}];
 1/Sqrt[1 - (num/den)^4]
];
A2K[A_] := Sqrt[1 - 10^(-A/10)]/10^(-A/20);
AdB[hz_,z_,f_] :=  -20*Log[10,Abs[hz /. z-> N[E^(I*2*Pi*f)] ]];
ba[n_,a_,e_,Fp_,z_] := Module[{i,g,t},
   {-2*Re[Zbl[S[n,a,e,1],Fp]], Abs[Zbl[S[n,a,e,1],Fp]]^2}
];
DGDelay[A_,B_,z_,f_] := Module[
```

```
{Ai,Bi,gA,gAd,gAn,nA,tAd,tAn,gB,gBd,gBn,nB,tBd,tBn},
   Ai = CoefficientList[A,z];
   Bi = CoefficientList[B,z];
   nA = Length[Ai];
   tAn = Join[{Sum[(i-1)*Ai[[i]]^2,{i,1,nA}]},
      Table[Cos[(k-1) w]*Sum[(2*i-k-1)*Ai[[i]]*Ai[[i-k+1]],{i,k,nA}],{k,2,nA}]];
   tAd = Join[{Sum[Ai[[i]]^2,{i,1,nA}]},
      Table[2*Cos[(k-1) w]*Sum[Ai[[i]]*Ai[[i-k+1]],{i,k,nA}],{k,2,nA}]];
  gAn = Sum[tAn[[i]],{i,1,nA}];
  gAd = Sum[tAd[[i]],{i,1,nA}];
  gA = Simplify[gAn/gAd];
  nB = Length[Bi];
  tBn = Join[{Sum[(i-1)*Bi[[i]]^2,{i,1,nB}]},
      Table[Cos[(k-1) w]*Sum[(2*i-k-1)*Bi[[i]]*Bi[[i-k+1]],{i,k,nB}],{k,2,nB}]];
  tBd = Join[{Sum[Bi[[i]]^2,{i,1,nB}]},
      Table[2*Cos[(k-1) w]*Sum[Bi[[i]]*Bi[[i-k+1]],{i,k,nB}],{k,2,nB}]];
  gBn = Sum[tBn[[i]],{i,1,nB}];
  gBd = Sum[tBd[[i]],{i,1,nB}];
  gB = Simplify[gBn/gBd];
 gB - gA +nA -nB /. z-> E^(I*2*N[Pi]*f) /. w-> 2*N[Pi]*f
];
Hz[n_,a_,e_,Fp_,z_] := Module[{i,g,t},
 If[EvenQ[n],
   g = Product[(2-2*Re[Zbl[I*a/X[n,a,i],Fp]])/
               (1-2*Re[Zbl[S[n,a,e,i],Fp]]
                +Abs[Zbl[S[n,a,e,i],Fp]]^2)
         ,{i,1,n/2}]*Sqrt[1+e^2];
   t = (1/g)*Product[(z^2-2*Re[Zbl[I*a/X[n,a,i],Fp]]*z+1)/
       (z^2-2*Re[Zbl[S[n,a,e,i],Fp]]*z
        +Abs[Zbl[S[n,a,e,i],Fp]]^2)
         ,{i,1,n/2}];,
   g = 2*Product[(1-2*Re[Zbl[I*a/X[n,a,i],Fp]]+1)/
       (1-2*Re[Zbl[S[n,a,e,i],Fp]]
         +Abs[Zbl[S[n,a,e,i],Fp]]^2)
         ,{i,1,(n-1)/2}]/(1-Zbl[S[n,a,e,(n+1)/2],Fp]);
   t = (1/g)*(z+1)*Product[(z^2-2*Re[Zbl[I*a/X[n,a,i],Fp]]*z+1)/
       (z^2-2*Re[Zbl[S[n,a,e,i],Fp]]*z
        +Abs[Zbl[S[n,a,e,i],Fp]]^2)
         ,{i,1,(n-1)/2}]/(z-Zbl[S[n,a,e,(n+1)/2],Fp]);
 ];
 t
];
Ke[n_Integer, a_, e_, x_] := e*Abs[R[n,a,x]];
```

```
L[n_Integer,a_] := Module[{i,r},
 If[EvenQ[n],
  r = (1/a^n)*Product[(a^2 - X[n,a,i]^2)^2, {i,n/2}]/
             Product[(1   - X[n,a,i]^2)^2, {i,n/2}];,
  r = (1/a^(n-2))*Product[(a^2 - X[n,a,i]^2)^2, {i,(n-1)/2}]/
                 Product[(1   - X[n,a,i]^2)^2, {i,(n-1)/2}];
 ];
 r
];
nbut[Fp_,Fs_,Ap_,As_] := Module[
 {L = Sqrt[(-1 + 10^(As/10))/(-1 + 10^(Ap/10))],
  aspec = Tan[Pi*Fs]/Tan[Pi*Fp]},
 Ceiling[Log[10,L]/Log[10,aspec]//N]
];
ncheb[Fp_,Fs_,Ap_,As_] := Module[
 {L = Sqrt[(-1 + 10^(As/10))/(-1 + 10^(Ap/10))],
  aspec = Tan[Pi*Fs]/Tan[Pi*Fp]},
 Ceiling[ArcCosh[L]/ArcCosh[aspec]//N]
];
nellip[Fp_,Fs_,Ap_,As_] := Module[
 {num, den,
  k = Tan[Pi*Fp]/Tan[Pi*Fs],
  L = Sqrt[(-1 + 10^(As/10))/(-1 + 10^(Ap/10))]},
 num = EllipticK[1-1/L^2]/EllipticK[1/L^2];
 den = EllipticK[1-k^2]/EllipticK[k^2];
 Ceiling[num/den//N]
];
nminQ[Fp_, Fs_, Ap_, As_] := Block[{ai, i, Kp, Ks, x, x1, x2},
 i = nellip[Fp,Fs,Ap,As];
 Kp = A2K[Ap];
 Ks = A2K[As];
 x1 = Tan[Pi*Fs]/Tan[Pi*Fp];
 x2 = x1^2;
 ai = x /. FindRoot[Sqrt[L[i, x]] == Ks, {x,x1,x2}];
 While[Ke[i, ai, 1/Ks, Fp/(Fs/ai)] > Kp,
  i += 1;
  x2 = ai;
  ai = x /. FindRoot[Sqrt[L[i, x]] == Ks, {x,x1,x2}];
 ];
 i
];
```

```
plotstyle012345 :=  PlotStyle -> {Dashing[{}],
                                  Dashing[{0.01}],
                                  Dashing[{0.02}],
                                  Dashing[{0.03}],
                                  Dashing[{0.04}],
                                  Dashing[{0.05}]}

plotPTS[MdB_,f_,spec_List] := Block[
{Fpass=spec[[1]], Fstop=spec[[2]],
 Apass=spec[[3]], Astop=spec[[4]], c1, c2, g1 ,g2, g3, M},
  M = MdB;
  c1 = PlotStyle -> {RGBColor[1,0,0],RGBColor[1,0,0],RGBColor[0,0,1]};
  c2 = PlotStyle -> {RGBColor[0,1,0],RGBColor[0,1,0],RGBColor[0,0,1]};
 g1 = Plot[{Apass,0,M}, {f,0,Fpass}, Evaluate[c1]
      , PlotRange -> All
      , Ticks -> {{0, Fp},{0,Ap/2,Ap}}
      , AxesOrigin->{-0.001,-0.01}];
 g2 = Plot[{Apass,Astop,M}, {f,Fpass,Fstop}, Evaluate[c2]
      , PlotRange -> All
      , Ticks -> {{Fp,Fs},{Ap, As/2, As}}
      , AxesOrigin->{Fpass-0.001,0}];
 g3 = Plot[{Astop,Astop,M}, {f,Fstop,0.5}, Evaluate[c1]
      , PlotRange -> {Astop,Astop+41}
      , Ticks -> {{Fs,0.5},{As, As+20, As+40}}
      , AxesOrigin->{Fstop-0.001,Astop-2}];
 Show[GraphicsArray[{g1,g2,g3}
]];
];

q[k_] := Module[{c,e,r,s,t},
 If[k<=1/Sqrt[2.0],
   t = (1/2)*(1 - (1-k^2)^(1/4))/(1 + (1-k^2)^(1/4));,
   t = (1/2)*(1 - Sqrt[k])/(1 + Sqrt[k]);
 ];
 e = {1,5, 9, 13,  17,   21,    25,     29,      33,       37};
 c = {1,2,15,150,1707,20910,268616,3567400,48555069,673458874};
 s = Sum[c[[i]]*(t^e[[i]]),{i,Length[e]}];
 If[k<=1/Sqrt[2.0],
   r = s;,
   r = Exp[Pi^2/Log[s]];
 ];
 N[r]
];

Qfactor[a_,b_] := Sqrt[(1+a+b)*(1+a-b)]/(2*(1-a));

rtan[fnum_,fden_,prec_:16] := N[Tan[Pi*fnum]/Tan[Pi*fden],prec];
```

```
R[n_Integer,a_,x_] := Module[{i,r,r0},
 If[EvenQ[n],
  r = Product[x^2 -     X[n,a,i]^2, {i,n/2}]/
      Product[x^2 - a^2/X[n,a,i]^2, {i,n/2}];
  r0 = Product[1 -     X[n,a,i]^2, {i,n/2}]/
       Product[1 - a^2/X[n,a,i]^2, {i,n/2}];,
  r = x*Product[x^2 -     X[n,a,i]^2, {i,(n-1)/2}]/
        Product[x^2 - a^2/X[n,a,i]^2, {i,(n-1)/2}];
  r0 = Product[1 -     X[n,a,i]^2, {i,(n-1)/2}]/
       Product[1 - a^2/X[n,a,i]^2, {i,(n-1)/2}];
 ];
 r/r0
];
```

```
S[n_Integer, a_, e_] :=  S[n,a,e,#]& /@ Range[n];
```

```
S[n_Integer, a_, e_, i_Integer] := Module[
{den,num,numim,numre,x,z},
 x = X[n,a,i];
 z = Z[n,a,e];
 numre = -z*Sqrt[1 - z^2]*Sqrt[1 - x^2]*Sqrt[1 - x^2/a^2];
 numim = x*Sqrt[1 - (1-1/a^2)*z^2];
 num = numre + I*numim;
 den = 1 - (1 - x^2/a^2)*z^2;
 num/den
];
```

```
X[n_Integer, a_] :=  X[n,a,#]& /@ Range[n];
```

```
X[n_Integer,a_,i_Integer] :=  -JacobiCD[
 (2*i-1)*EllipticK[1/a^2]/n, 1/a^2
];
X[n_Integer,a_,i_Integer] := 0 /; And[i==(n+1)/2,OddQ[n]];
```

```
Z[n_Integer,a_,e_] :=  JacobiSN[
 InverseJacobiSN[1/Sqrt[1+e^2],1-1/(L[n,a])^2]*
 EllipticK[1-1/a^2]/EllipticK[1-1/(L[n,a])^2],1-1/a^2
];
```

```
Zbl[sp_,Fp_] := (1+sp*(Tan[Pi*Fp]))/(1-sp*(Tan[Pi*Fp]));
```

■ A.9.5 Specification

```
SA = {0.2, 0.212, 0.2, 40.};
{Fp, Fs, Ap, As} = SA;
Kp = A2K[Ap];
Ks = A2K[As];
SK = {Fp, Fs, Kp, Ks}
```

```
{0.2, 0.212, 0.217091, 99.995}
```

■ A.9.6 Minimum Order

```
{nellip[Fp,Fs,Ap,As], ncheb[Fp,Fs,Ap,As], nbut[Fp,Fs,Ap,As]}
nmin = nellip[Fp,Fs,Ap,As];
nmax = 2*nmin;
nlist = Range[nmin,nmax]

{8, 18, 79}
{8, 9, 10, 11, 12, 13, 14, 15, 16}
```

■ A.9.7 Range of Selectivity Factor and Ripple Factor

```
xtan = rtan[Fs,Fp];
amin8  = A2a[nmin,Ap,As];
amax8  = a/. FindRoot[R[nmin,a,xtan]==Ks/Kp, {a,xtan,1.095}];
amin9  = A2a[9,Ap,As];
amax9  = a/. FindRoot[R[ 9,a,xtan]==Ks/Kp, {a,amax8,1.1}];
amin10 = A2a[10,Ap,As];
amax10 = a/. FindRoot[R[10,a,xtan]==Ks/Kp, {a,amax9,1.2}];
amin11 = A2a[11,Ap,As];
amax11 = a/. FindRoot[R[11,a,xtan]==Ks/Kp, {a,amax10,1.2}];
amin12 = A2a[12,Ap,As];
amax12 = a/. FindRoot[R[12,a,xtan]==Ks/Kp, {a,amax11,1.2}];
amin13 = A2a[13,Ap,As];
amax13 = a/. FindRoot[R[13,a,xtan]==Ks/Kp, {a,amax12,1.3}];
amin14 = A2a[14,Ap,As];
amax14 = a/. FindRoot[R[14,a,xtan]==Ks/Kp, {a,amax13,1.4}];
amin15 = A2a[15,Ap,As];
amax15 = a/. FindRoot[R[15,a,xtan]==Ks/Kp, {a,amax14,1.6}];
amin16 = A2a[16,Ap,As];
amax16 = a/. FindRoot[R[16,a,xtan]==Ks/Kp, {a,amax15,1.9}];
aminlist = {amin8,amin9,amin10,amin11,amin12,amin13,amin14,amin15,amin16};
amaxlist = {amax8,amax9,amax10,amax11,amax12,amax13,amax14,amax15,amax16};
eminlist = Table[Ks/L[n,amaxlist[[n-8+1]]], {n,nmin,nmax}];
{emin8,emin9,emin10,emin11,emin12,emin13,emin14,emin15,emin16} = eminlist;
emax = Kp;
emaxlist = Table[Kp,{nmax-nmin+1}];
TableForm[Transpose[{nlist,aminlist,amaxlist
                    ,ScientificForm/@eminlist,emaxlist}]
, TableHeadings->{{},{"n","amin","amax","emin","emax"}}]
```

n	amin	amax	emin	emax
8	1.04285	1.09245	$6.25782\ 10^{-2}$	0.217091
9	1.022	1.11016	$1.43176\ 10^{-2}$	0.217091
10	1.01135	1.13668	$2.62202\ 10^{-3}$	0.217091
11	1.00587	1.17518	$3.65611\ 10^{-4}$	0.217091

12	1.00304	1.23116	$3.61027\ 10^{-5}$	0.217091
13	1.00158	1.31502	$2.23759\ 10^{-6}$	0.217091
14	1.00082	1.44884	$6.96526\ 10^{-8}$	0.217091
15	1.00042	1.69029	$6.68381\ 10^{-10}$	0.217091
16	1.00022	2.26906	$4.39641\ 10^{-13}$	0.217091

■ A.9.8 Range of Edge Frequencies

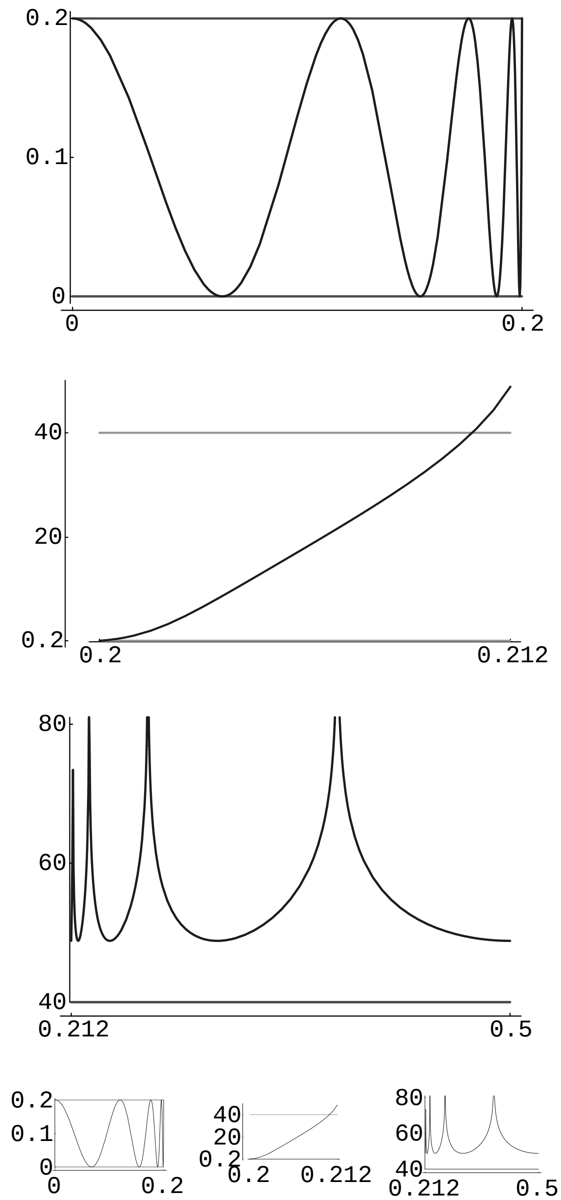
```
fpminlist = ArcTan[Tan[Pi*Fs] / amaxlist]/Pi //N;
{fpmin8,fpmin9,fpmin10,fpmin11,fpmin12,fpmin13,
 fpmin14,fpmin15,fpmin16} = fpminlist;
fpmaxlist = ArcTan[Tan[Pi*Fs] / aminlist]/Pi //N;
{fpmax8,fpmax9,fpmax10,fpmax11,fpmax12,fpmax13,
 fpmax14,fpmax15,fpmax16} = fpmaxlist;
fsminlist = ArcTan[Tan[Pi*Fp] * aminlist]/Pi //N;
{fsmin8,fsmin9,fsmin10,fsmin11,fsmin12,fsmin13,
 fsmin14,fsmin15,fsmin16} = fsminlist;
fsmaxlist = ArcTan[Tan[Pi*Fp] * amaxlist]/Pi //N;
{fsmax8,fsmax9,fsmax10,fsmax11,fsmax12,fsmax13,
 fsmax14,fsmax15,fsmax16} = fsmaxlist;
TableForm[Transpose[{nlist,fpminlist,fpmaxlist,fsminlist,fsmaxlist}]
, TableHeadings->{{},{"n","fpmin","fpmax","fsmin","fsmax"}}]
```

n	fpmin	fpmax	fsmin	fsmax
8	0.198485	0.205546	0.20639	0.213552
9	0.196065	0.208643	0.203305	0.216049
10	0.192535	0.210256	0.201712	0.21973
11	0.187606	0.211095	0.200887	0.224951
12	0.180824	0.21153	0.20046	0.23229
13	0.171447	0.211756	0.200239	0.242744
14	0.158187	0.211874	0.200124	0.258162
15	0.138517	0.211934	0.200064	0.282469
16	0.106119	0.211966	0.200033	0.326442

■ A.9.9 Transfer Functions and Plots of Attenuation in dB Versus Frequency

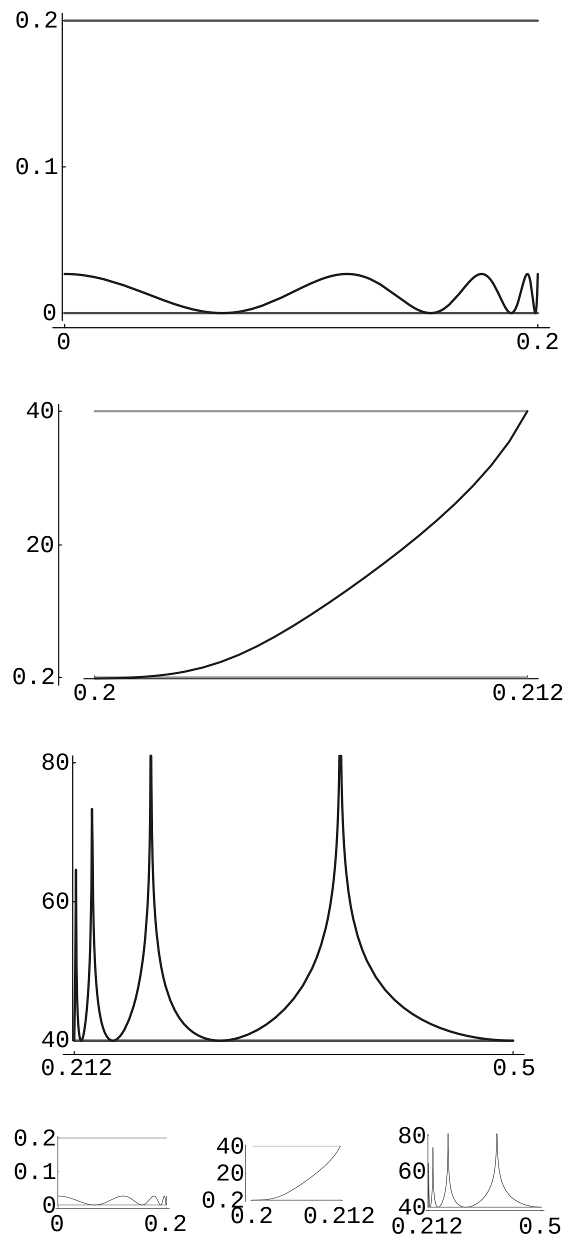
■ A.9.10 Design D1

```
hz8D1 = Hz[8, rtan[Fs,Fp], emax, Fp, z] //N;
AdB8D1 = AdB[hz8D1,z,f];
{b1n8,a1n8} = ba[8, rtan[Fs,Fp], emax, Fp, z] //N;
q1n8 = Qfactor[a1n8,b1n8];
plotPTS[AdB8D1,f,{0.2,0.212,0.2,40}]
```



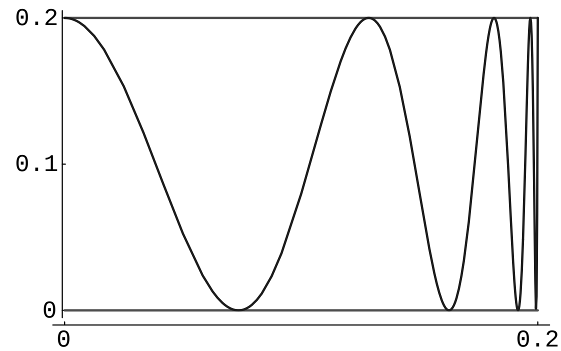
■ A.9.11 Design D2

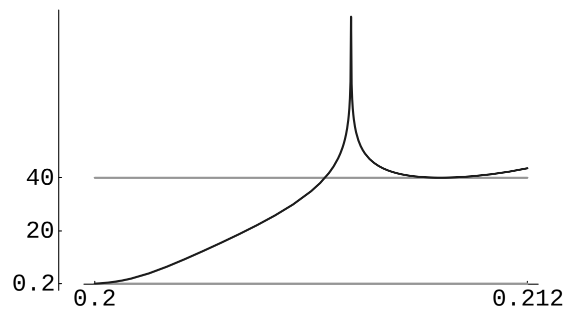
```
hz8D2 = Hz[8, rtan[Fs,Fp], Ks/L[8,rtan[Fs,Fp]], Fp, z] //N;
AdB8D2 = AdB[hz8D2,z,f];
{b2n8,a2n8} = ba[8, rtan[Fs,Fp], Ks/L[8,rtan[Fs,Fp]], Fp, z] //N;
q2n8 = Qfactor[a2n8,b2n8];
plotPTS[AdB8D2,f,{0.2,0.212,0.2,40}]
```

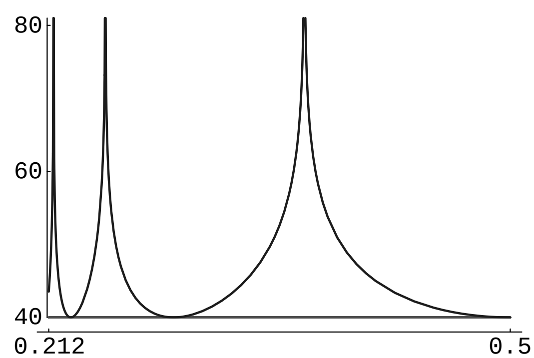


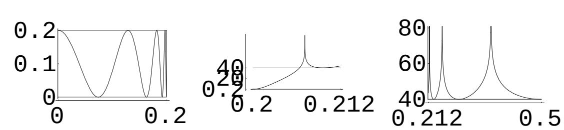
■ A.9.12 Design D3a

```
hz8D3a = Hz[8, amin8, emax, Fp, z] //N;
AdB8D3a = AdB[hz8D3a,z,f];
{b3an8,a3an8} = ba[8, amin8, emax, Fp, z] //N;
q3an8 = Qfactor[a3an8,b3an8];
plotPTS[AdB8D3a,f,{0.2,0.212,0.2,40}]
```



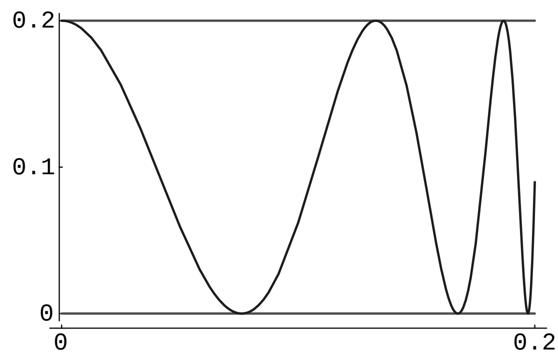


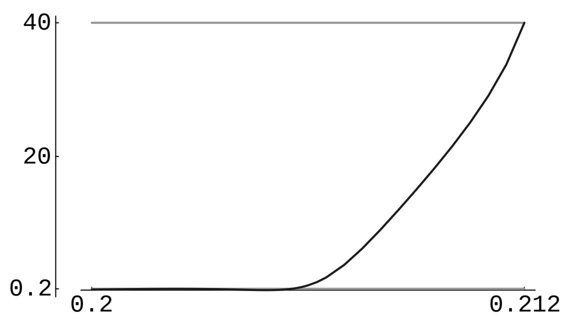


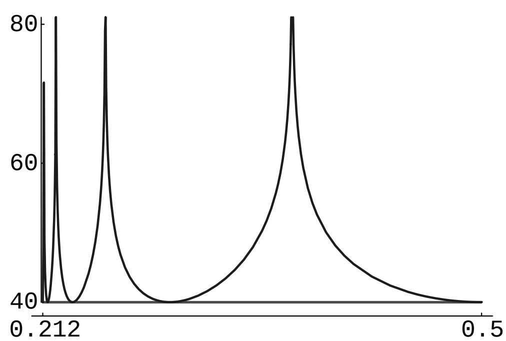


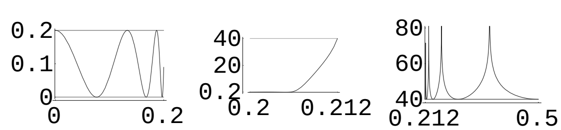
■ A.9.13 Design D3b

```
hz8D3b = Hz[8, amin8, emax, fpmax8, z] //N;
AdB8D3b = AdB[hz8D3b,z,f];
{b3bn8,a3bn8} = ba[8, amin8, emax, fpmax8, z] //N;
q3bn8 = Qfactor[a3bn8,b3bn8];
plotPTS[AdB8D3b,f,{0.2,0.212,0.2,40}]
```



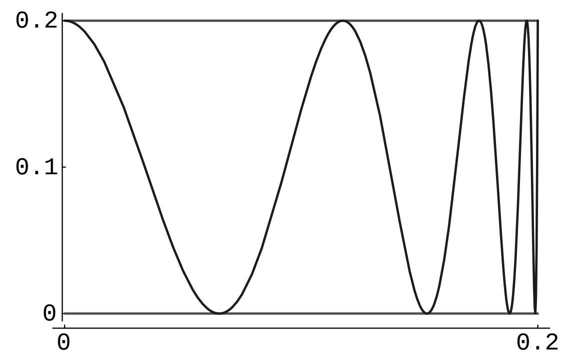


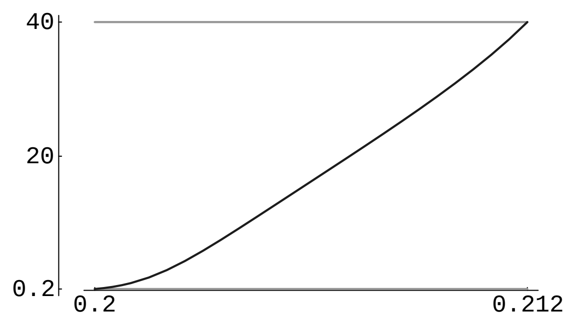


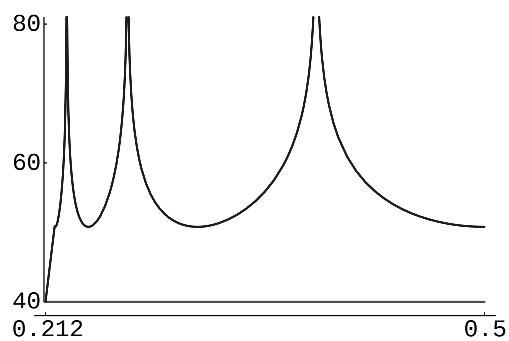


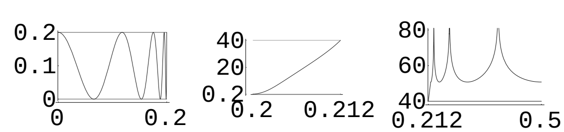
■ A.9.14 Design D4a

```
hz8D4a = Hz[8, amax8, emax, Fp, z] //N;
AdB8D4a = AdB[hz8D4a,z,f];
{b4an8,a4an8} = ba[8, amax8, emax, Fp, z] //N;
q4an8 = Qfactor[a4an8,b4an8];
plotPTS[AdB8D4a,f,{0.2,0.212,0.2,40}]
```



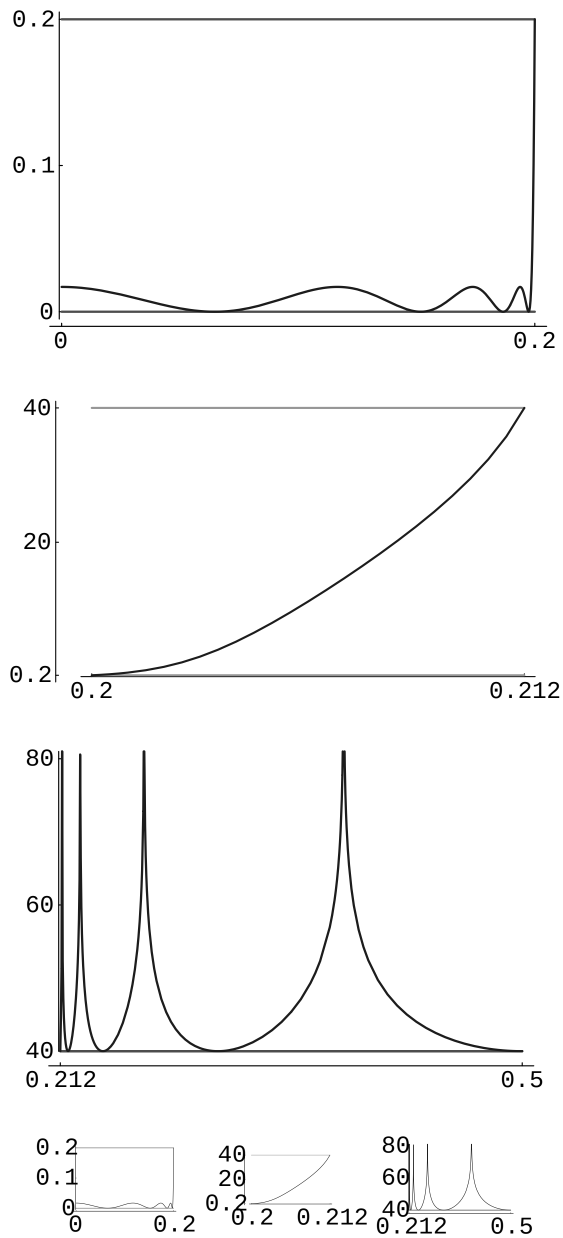






■ A.9.15 Design D4b

```
hz8D4b = Hz[8, amax8, emin8, fpmin8, z] //N;
AdB8D4b = AdB[hz8D4b,z,f];
{b4bn8,a4bn8} = ba[8, amax8, emin8, fpmin8, z] //N;
q4bn8 = Qfactor[a4bn8,b4bn8];
plotPTS[AdB8D4b,f,{0.2,0.212,0.2,40}]
```



■ A.9.16 Coefficients a and b, Q-Factor, and Sensitivity of the Second-Order Section with the Maximal Q-Factor

```
alist = {a1n8, a2n8, a3an8, a3bn8, a4an8, a4bn8};
blist = {b1n8, b2n8, b3an8, b3bn8, b4an8, b4bn8};
qlist = {q1n8, q2n8, q3an8, q3bn8, q4an8, q4bn8};
slist = 1/(1-alist)
```

```
{30.4061, 23.9766, 44.7132, 44.2529, 28.3548, 21.5643}
```

```
 TableForm[Transpose[{blist,alist,qlist,slist,slist^3}]
, TableHeadings->{{"D1","D2","D3a","D3b","D4a","D4b"}
   , {"b","a","Q-factor","1/(1-a)","1/(1-a)^3"}}]
```

	b	a	Q-factor	1/(1-a)	1/(1-a)^3
D1	-0.593746	0.967112	28.5112	30.4061	28111.3
D2	-0.565538	0.958293	22.4763	23.9766	13783.6
D3a	-0.600922	0.977635	42.1227	44.7132	89393.7
D3b	-0.534747	0.977403	42.1227	44.2529	86661.3
D4a	-0.592153	0.964733	26.5596	28.3548	22797.2
D4b	-0.572418	0.953627	20.1398	21.5643	10027.8

■ Remark

The expressions 1/(1-a) and 1/(1-a)^3 are used to estimate the magnitude response sensitivity to the transfer function coefficients.

A.10 Jacobi Elliptic Functions

Miroslav D. Lutovac, Dejan V. Tosic and Brian L. Evans
lutovac@iritel.bg.ac.yu tosic@galeb.etf.bg.ac.yu bevans@ece.utexas.edu

■ A.10.1 References

1. M.D.Lutovac, D.V. Tosic, I.M.Markoski,
"Symbolic computation of elliptic rational functions,"
5th Int. Workshop, Symbolic Methods, Applications, Circuit Design,
SMACD'98, Kaiserslautern, Germany,
Oct. 1998, pp.177–180.

2. M. Abramowitz and I. Stegun,
"Handbook of Mathematical Functions,"
Dover, New York, 1972.

3. M. D. Lutovac, D. V. Tosic and B. L. Evans,
"Advanced Filter Design for Signal Processing using MATLAB and Mathematica,"
http://iritel.iritel.bg.ac.yu/~lutovac/www/afdhome.htm
http://galeb.etf.bg.ac.yu/~tosic/afdhome.htm
http://www.ece.utexas.edu/~bevans/

■ A.10.2 Initialization

```
SetDirectory[HomeDirectory[]];
<<afd\math\m\clearall.m
```

■ A.10.3 Notation

cd[u,k] – Jacobi sine shifted
cn[u,k] – Jacobi cosine
k – modulus
K[k] – complete elliptic integral of the first kind
Kp[k] – complementary complete elliptic integral of the first kind
sn[u,k] – Jacobi sine

■ A.10.4 Definitions

```
sn[u_,k_] := JacobiSN[u,k^2];
cn[u_,k_] := JacobiCN[u,k^2];
cd[u_,k_] := JacobiCD[u,k^2];
dn[u_,k_] := JacobiDN[u,k^2];
invsn[v_,k_] := InverseJacobiSN[v,k^2];
K[k_] := EllipticK[k^2];
Kp[k_] := K[Sqrt[1-k^2]];
```

■ A.10.5 Basic properties

```
{sn[0,k], sn[K[k],k], cn[0,k], cn[K[k],k]}
```

```
{0, 1, 1, 0}
```

```
a = 1.1;
b = 0.9;
{sn[I*a,b],
 I*sn[a,Sqrt[1-b^2]]/cn[a,Sqrt[1-b^2]]}
{cn[I*a,b],
 1/cn[a,Sqrt[1-b^2]]}
{dn[I*a,b],
 dn[a,Sqrt[1-b^2]]/cn[a,Sqrt[1-b^2]]}
```

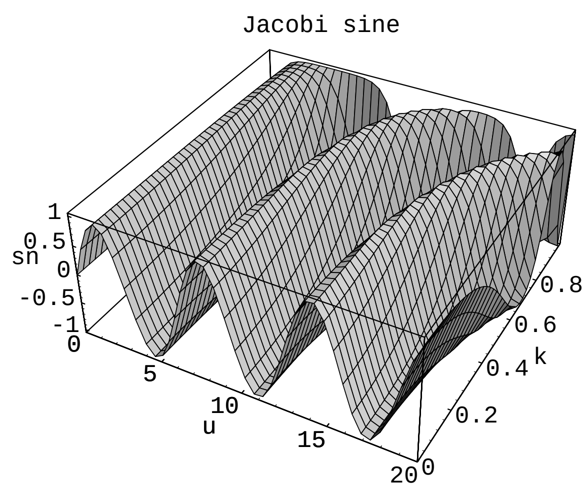
```
{1.81411 I, 1.81411 I}
{2.07147, 2.07147}
{1.9146, 1.9146}
```

```
c = sn[a,b]
d = invsn[c,b]
a - d
```

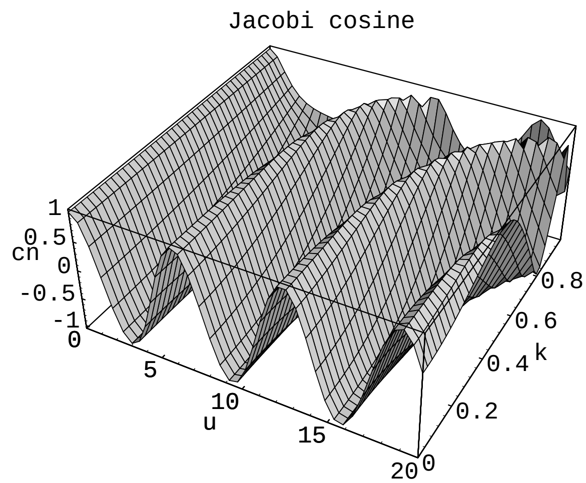
```
0.819407
1.1
           -16
6.66134 10
```

A.10.6 Plots

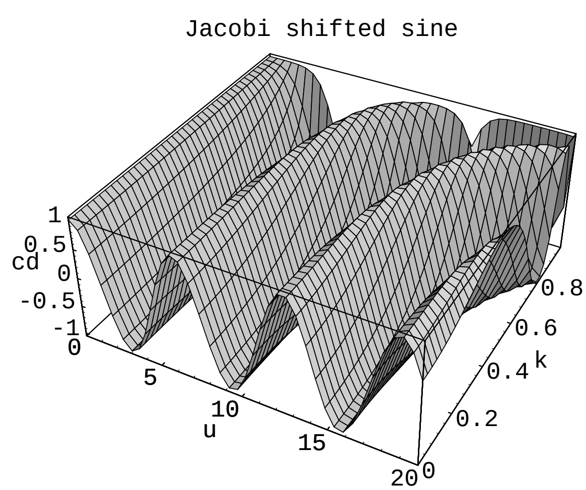
```
Plot3D[sn[u,k]
, {u,0,20},{k,0,0.999}
, AxesLabel -> {"u", "k", "sn"}
, PlotLabel -> "Jacobi sine"
, PlotPoints -> 40
];
```



```
Plot3D[cn[u,k]
, {u,0,20},{k,0,0.999}
, AxesLabel -> {"u", "k", "cn"}
, PlotLabel -> "Jacobi cosine"
, PlotPoints -> 40
];
```




```
Plot3D[cd[u,k]
, {u,0,20},{k,0,0.999}
, AxesLabel -> {"u", "k", "cd"}
, PlotLabel -> "Jacobi shifted sine"
, PlotPoints -> 40
];
```



A.11 Elliptic Rational Function

Miroslav D. Lutovac, Dejan V. Tosic and Brian L. Evans
lutovac@iritel.bg.ac.yu tosic@galeb.etf.bg.ac.yu bevans@ece.utexas.edu

■ A.11.1 References

1. M. D. Lutovac, D.M.Rabrenovic,
"A simplified design of some Cauer filters without jacobian elliptic functions,"
IEEE Trans. Circ. Systems: Part II, vol. 39, no. 9, pp. 666–671, Sept. 1992.

2. M. D. Lutovac, D.M.Rabrenovic,
"Algebraic design of some lower-order elliptic filters,"
Electronics Letters, vol. 29, no. 2, pp. 192–193, Jan. 1993.

3. D.M.Rabrenovic, M. D. Lutovac,
"Minimum stopband attenuation of the Cauer filters without elliptic functions and integrals,"
IEEE Trans. Circ. Systems: Part I, vol. 40, no. 9, pp. 618–621, Sept. 1993.

4. M. D. Lutovac, D.M.Rabrenovic,
"Exact determination of the natural modes of some Cauer filters by means of a standard analytical procedure,"
IEE Proc. Circuits Devices Syst., vol. 143, no. 3, pp. 134–138, June 1996.

5. M. D. Lutovac, D. V. Tosic and B. L. Evans,
"Advanced Filter Design for Signal Processing using MATLAB and Mathematica,"
http://iritel.iritel.bg.ac.yu/˜lutovac/www/afdhome.htm
http://galeb.etf.bg.ac.yu/˜tosic/afdhome.htm
http://www.ece.utexas.edu/˜bevans/

■ A.11.2 Initialization

```
SetDirectory[HomeDirectory[]];
<<afd\math\m\clearall.m
<<afd\math\m\jeldf.m
<<graphics‘graphics‘
```

```
Discrimination Factor L(n,1/k)
n = 1,2,3,4,6,8,9,12,16,18
```

■ A.11.3 Notation

cd[u,k] – Jacobi sine shifted
invcd[v,k] – inverse of cd
k – modulus
K[k] – complete elliptic integral of the first kind
Kp[k] – complementary complete elliptic integral of the first kind
L[n,1/k] – discrimination factor
n – order, n>0
R[n,k,p,x] – function from which we generate elliptic rational function

A.11.4 Definitions

```
cd[u_,k_] := JacobiCD[u,k^2];
invcd[v_,k_] := InverseJacobiCD[v,k^2];
K[k_] := EllipticK[k^2];
Kp[k_] := K[Sqrt[1-k^2]];
R[n_,k_,p_,x_] := cd[n*K[p]*invcd[x,k]/K[k],p]
x[k_,u_,v_] := cd[u+I*v,k];
Ruv[n_Integer,k_,u_,v_] := cd[n*(u+I*v),1/L[n,1/k]];
```

A.11.5 Example

```
{R[2,0.8,0.1,-1.0], R[2,0.8,0.1,0], R[2,0.8,0.1,1.0]}
```

```
{1., -1., 1.}
```

Remark

We know that invcd(0,k)=K(k), so, we add a rule

```
{invcd[0,0.9], K[0.9]}
invcd[0.0,0.9]
```

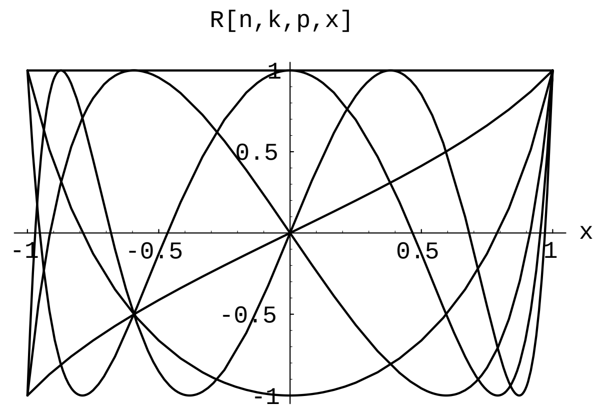
```
{2.28055, 2.28055}
                                 1
Power::infy: Infinite expression -- encountered.
                                 0.
Infinity::indet: Indeterminate expression 0. ComplexInfinity encountered.
2.28054913842277 - 2 EllipticLog[{Indeterminate, ComplexInfinity}, {3.62, 0.0361}] -
 2 (1.654616667522527 I Round[0.302185
       Im[0. - 2 EllipticLog[{Indeterminate, ComplexInfinity}, {3.62, 0.0361}]]] +
    2.28054913842277 Round[0.219245 Re[0. -
         2 EllipticLog[{Indeterminate, ComplexInfinity}, {3.62, 0.0361}]]]])
```

```
invcd[0.0,k_] := K[k];
{invcd[0,0.9], invcd[0.0,0.9]}
```

```
{2.28054913842277, 2.28055}
```

A.11.6 Plots

```
Plot[{R[0, 0.8, 0.1, x],
      R[1, 0.8, 0.1, x],
      R[2, 0.8, 0.1, x],
      R[3, 0.8, 0.1, x],
      R[4, 0.8, 0.1, x],
      R[5, 0.8, 0.1, x]}
, {x,-1,1}
, AxesLabel -> {"x", "R[n,k,p,x]"}];
```



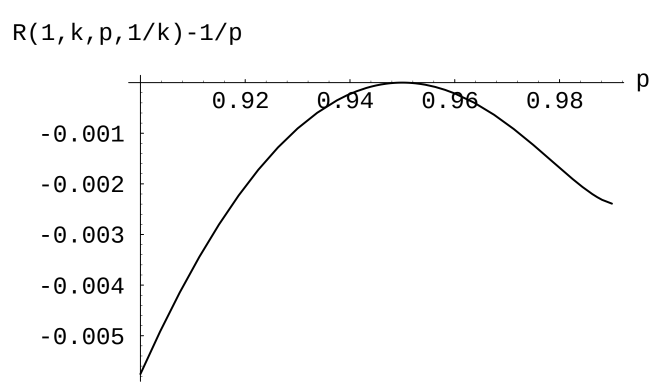
■ A.11.7 Can we Construct a Rational Function from R(n,k,p,x) by Varying k and p?

For n=1
R(1,k,p,x) = cd(K(p)*invcd(x,k)/K(k),p)
and simplyfies to R(1,k,p,x)=x for k=p.
R(1,k,p,0) = 0
R(1,k,p,1) = 1
R(1,k,p,1/k) = 1/k = 1/p

■ A.11.8 Find p Which Satisfies R(n,k,p,1/k)=1/p for n=1

```
n1 = 1
k1 = N[95/100,24]
Plot[R[n1,k1,p,1/k1] - 1/p
, {p,0.9,0.99}
, AxesLabel -> {"p", "R(1,k,p,1/k)-1/p"}];
```

```
1
0.95
```



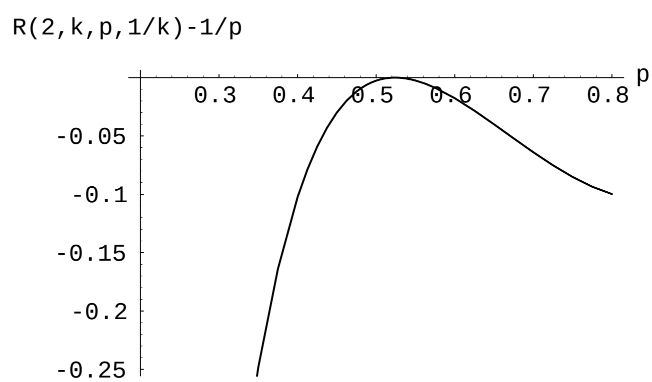
```
sol = FindRoot[
 R[n1,k1,p,1/k1] == 1/p
, {p,0.949,0.951}
, AccuracyGoal -> 16
, MaxIterations -> 64
];
p1 = N[p/.sol,16]
p1exact = k1
```

```
                                         -17
0.950000006790337 + 5.259681766043407 10    I
0.95
```

A.11.9 Find p Which Satisfies R(n,k,p,1/k)=1/p for n=2

```
n2 = 2
k2 = N[95/100,24]
Plot[R[n2,k2,p,1/k2] - 1/p
, {p,0.2,0.8}
, AxesLabel -> {"p", "R(2,k,p,1/k)-1/p"}];
```

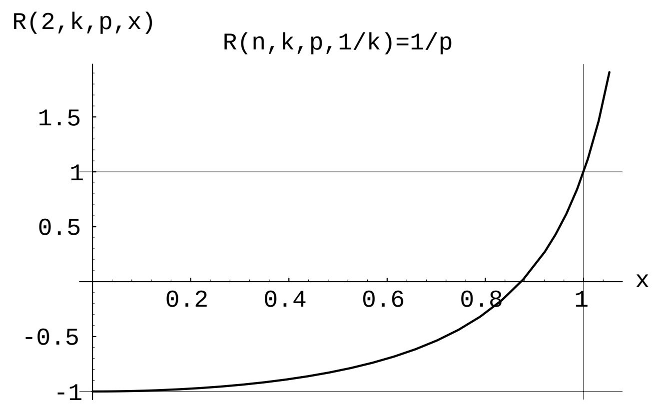
```
2
0.95
```




```
sol = FindRoot[
 R[n2,k2,p,1/k2] == 1/p
, {p,0.52,0.53}
, AccuracyGoal -> 16
, MaxIterations -> 64
];
p2 = N[p/.sol,16]
p2exact = 1/(1/k2+Sqrt[1/k2^2-1])^2
```

```
0.5240999634235638
0.52409994477580075281475
```

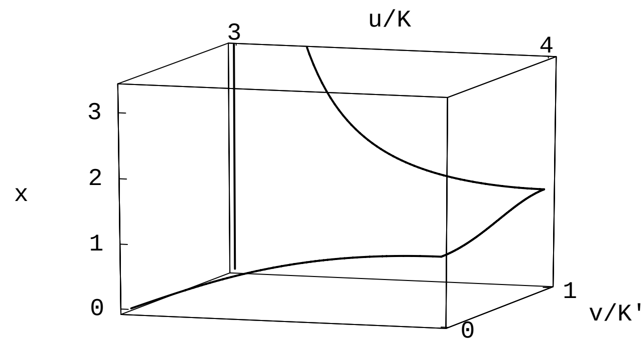
```
Plot[R[n2,k2,p2,x]
, {x,0,1/k2}
, AxesLabel -> {"x", "R(2,k,p,x)"}
, PlotLabel -> "R(n,k,p,1/k)=1/p"
, GridLines -> {{1},{-1,1}}
];
```



From now on we assume R(n,k,p,1/k)=1/p
and calculate p = 1/L(n,1/k)
(L can be exactly computed for n=1,2,3,4,6,8,9,12,16,18)

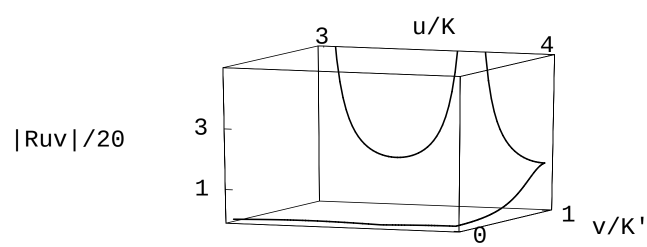
■ A.11.10 Parametric Plot of x in Terms of Complex Parametric Variable w=u+I*v

```
n1 = 3;
k1 = 0.7;
K1 = K[k1];
ParametricPlot3D[
     {{t, 0,      x[k1, t*K1, 0]},
      {4, t-3,    x[k1, 4*K1, (t-3)*K1]},
      {t, 1,   Re[x[k1, t*K1, K1]]}
     }
, {t,3,4}
, Ticks -> {{3,4},{0,1},{0,1,2,3,4}}
, AspectRatio -> 0.6
, ViewPoint -> {10,-30,20}
, AxesLabel -> {"u/K", "v/K'", "x"}
];
```



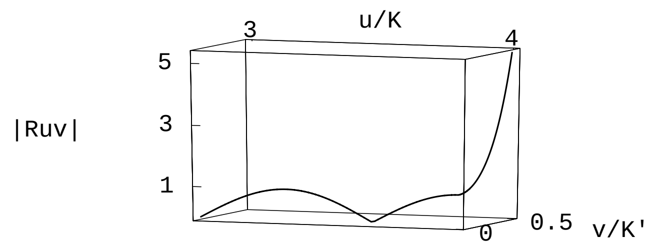
■ A.11.11 Parametric Plot of R in Terms of Complex Parametric Variable w=u+I*v

```
n1 = 3;
k1 = 0.7;
Kn1 = K[1/L[n1,1/k1]];
r0 = 1/20;
ParametricPlot3D[
     {{t, 0,   r0*Abs[Ruv[n1, k1, t*Kn1, 0]]},
      {4, t-3, r0*Abs[Ruv[n1, k1, 4*Kn1, (t-3)*Kn1]]},
      {t, 1,   r0*Abs[Ruv[n1, k1, t*Kn1, Kn1]]}
     }
, {t,3,4}
, Ticks -> {{3,4},{0,1},{1,3,5}}
, AspectRatio -> 0.5
, ViewPoint -> {10,-25,20}
, AxesLabel -> {"u/K", "v/K'", "|Ruv|/20"}
];
```

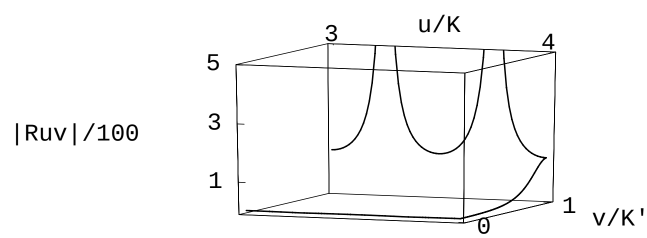



```
n1 = 3;
k1 = 0.7;
Kn1 = K[1/L[n1,1/k1]];
ParametricPlot3D[
     {{t, 0,       Abs[Ruv[n1, k1, t*Kn1, 0]]},
      {4, (t-3)/2, Abs[Ruv[n1, k1, 4*Kn1, (t-3)*Kn1/2]]}
     }
, {t,3,4}
, Ticks -> {{3,4},{0,0.5},{1,3,5}}
, AspectRatio -> 0.5
```

```
, ViewPoint -> {10,-25,20}
, AxesLabel -> {"u/K", "v/K'", "|Ruv|"}
];
```

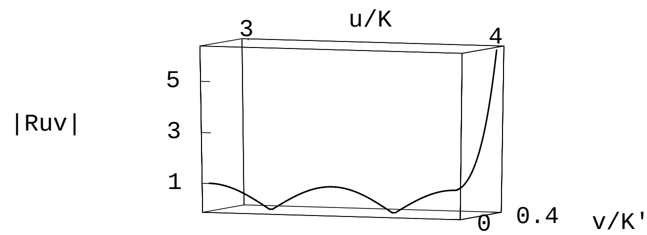



```
n1 = 4;
k1 = 0.7;
Kn1 = K[1/L[n1,1/k1]];
r0 = 1/100;
ParametricPlot3D[
     {{t, 0,   r0*Abs[Ruv[n1, k1, t*Kn1, 0]]},
      {4, t-3, r0*Abs[Ruv[n1, k1, 4*Kn1, (t-3)*Kn1]]},
      {t, 1,   r0*Abs[Ruv[n1, k1, t*Kn1, Kn1]]}
     }
, {t,3,4}
, Ticks -> {{3,4},{0,1},{1,3,5}}
, AspectRatio -> 0.5
, ViewPoint -> {10,-25,20}
, AxesLabel -> {"u/K", "v/K'", "|Ruv|/100"}
];
```




```
n1 = 4;
k1 = 0.7;
Kn1 = K[1/L[n1,1/k1]];
ParametricPlot3D[
     {{t, 0,         Abs[Ruv[n1, k1, t*Kn1, 0]]},
      {4, (t-3)/2.5, Abs[Ruv[n1, k1, 4*Kn1, (t-3)*Kn1/2.5]]}
     }
, {t,3,4}
, Ticks -> {{3,4},{0,0.4},{1,3,5}}
, AspectRatio -> 0.5
```

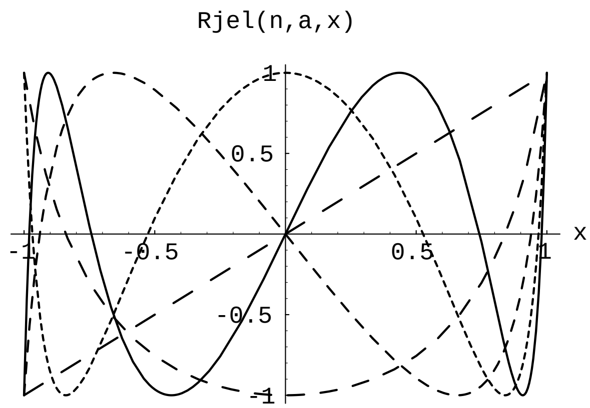
```
, ViewPoint -> {10,-25,20}
, AxesLabel -> {"u/K", "v/K'", "|Ruv|"}
];
```



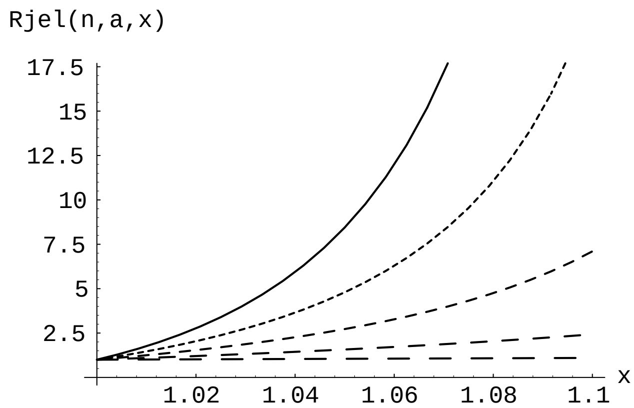
■ A.11.12 Elliptic Rational Function in Terms of Jacobi Elliptic Functions: a=1/k

```
Rjel[0, a_, x_] := 1
Rjel[1, a_, x_] := x
Rjel[2, a_, x_] := (( 1+Sqrt[1-1/a^2])*x^2-1)/
                   ((-1+Sqrt[1-1/a^2])*x^2+1) ;
Rjel[3,a_,x_] := Block[
{k=1/a,z1,z2,b1,b2,b3},
 z1 = JacobiDN[2*EllipticK[k^2]/3,k^2];
 b3 = (1 + z1)^2;
 b1 = -1 - 2*z1;
 b2 = -1 + z1^2;
(b3*x^3 + b1*x)/(b2*x^2 + 1)
];
Rjel[4, a_, x_] := Block[{t},
 t = Sqrt[1 - 1/a^2];
((1+t)*(1+Sqrt[t])^2*x^4-2(1+t)*(1+Sqrt[t])*x^2+1)/
((1+t)*(1-Sqrt[t])^2*x^4-2(1+t)*(1-Sqrt[t])*x^2+1)];
Rjel[5,a_,x_] := Block[
{k=1/a,z1,z2,b1,b2,b3,b4,b5},
 z1 = JacobiDN[2*EllipticK[k^2]/5,k^2];
 z2 = JacobiDN[4*EllipticK[k^2]/5,k^2];
 b5 = (1+z1)^2*(1+z2)^2;
 b3 = -( 1-z1^2*z2^2+(1+z1)^2*(1+z2)^2+2*(z1+z2));
 b1 = ( 1+2*(z1+z2));
 b4 = (1-z1^2)*(1-z2^2);
 b2 = -(2-z1^2-z2^2);
(b5*x^5 + b3*x^3 + b1*x)/(b4*x^4 + b2*x^2 + 1)
];
a1 = 1.1;
Plot[{Rjel[1, a1, x],
      Rjel[2, a1, x],
      Rjel[3, a1, x],
      Rjel[4, a1, x],
      Rjel[5, a1, x]}
```

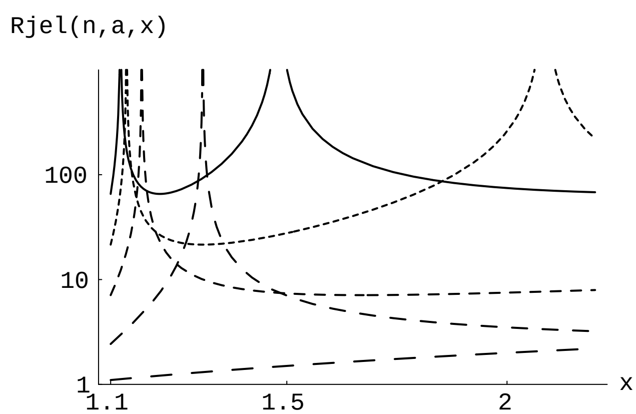
```
, {x,-1,1}
, AxesLabel -> {"x", "Rjel(n,a,x)"}
, PlotStyle -> {Dashing[{0.04}],
               Dashing[{0.03}],
               Dashing[{0.02}],
               Dashing[{0.01}],
               Dashing[{}]}
];
```




```
a1 = 1.1;
Plot[{Rjel[1, a1, x],
      Rjel[2, a1, x],
      Rjel[3, a1, x],
      Rjel[4, a1, x],
      Rjel[5, a1, x]}
, {x,1,a1}
, AxesLabel -> {"x", "Rjel(n,a,x)"}
, PlotStyle -> {Dashing[{0.04}],
               Dashing[{0.03}],
               Dashing[{0.02}],
               Dashing[{0.01}],
               Dashing[{}]}
];
```

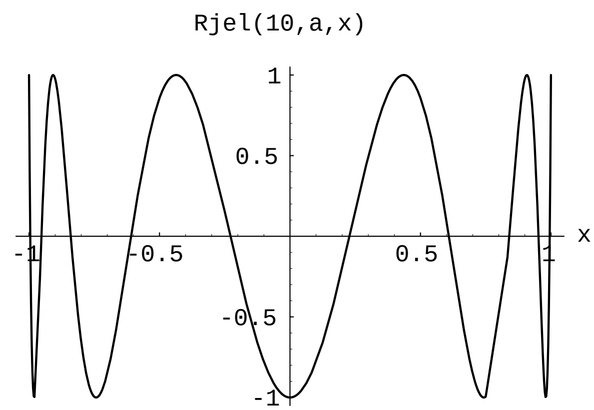



```
a1 = 1.1;
LogPlot[{Abs[Rjel[1, a1, x]],
         Abs[Rjel[2, a1, x]],
         Abs[Rjel[3, a1, x]],
         Abs[Rjel[4, a1, x]],
         Abs[Rjel[5, a1, x]]}
, {x,a1,2*a1}
, AxesLabel -> {"x", "Rjel(n,a,x)"}
, PlotRange -> {1,1000}
, Ticks -> {{a1,1.5,2},{1,10,100}}
, PlotStyle -> {Dashing[{0.04}],
                Dashing[{0.03}],
                Dashing[{0.02}],
                Dashing[{0.01}],
                Dashing[{}]}
];
```

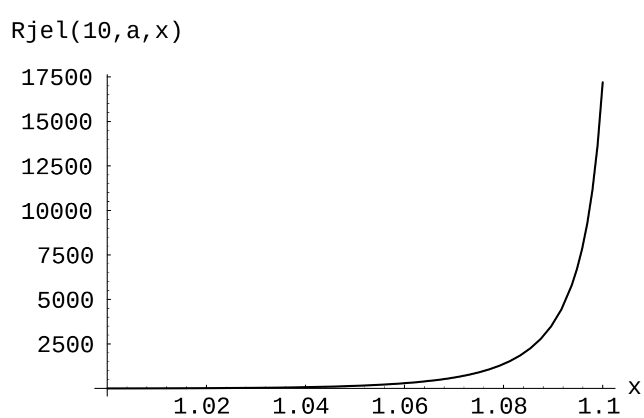


■ A.11.13 Nesting Property

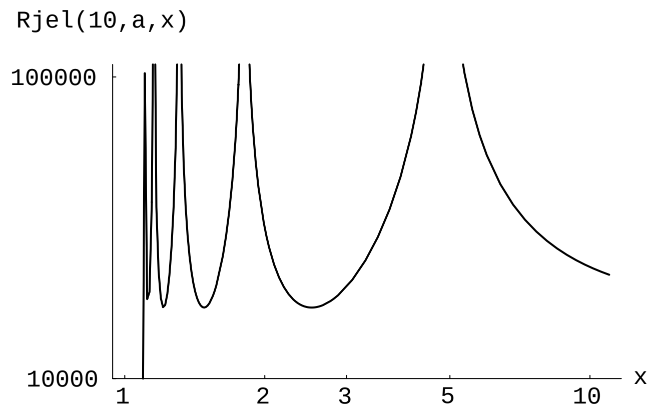
```
Rjel[10, a_, x_] := Rjel[2, Rjel[5,a,a], Rjel[5,a,x]];
a1 = 1.1;
{Rjel[10,a1,-1.0], Rjel[10,a1,0.0], Rjel[10,a1,1.0]}
{1., -1., 1.}
a1 = 1.1;
Plot[Evaluate[Rjel[10, a1, x]]
, {x,-1,1}
, AxesLabel -> {"x", "Rjel(10,a,x)"}
];
```




```
a1 = 1.1;
Plot[Evaluate[Rjel[10, a1, x]]
, {x,1,a1}
, AxesLabel -> {"x", "Rjel(10,a,x)"}
];
```



```
a1 = 1.1;
LogLogPlot[Evaluate[Abs[Rjel[10, a1, x]]]
, {x,1,10*a1}
, AxesLabel -> {"x", "Rjel(10,a,x)"}
, PlotRange -> {10^4,1.1*10^5}
, Ticks -> {{1,2,3,5,10}, {10^4, 10^5}}
];
```




```
Rjel[10,a1,x] //Together
```

```
                   2             4            6           8            10
211.208 - 5361.41 x  + 28064.4 x  - 57774.2 x  + 52056. x  - 17198.7 x
-----------------------------------------------------------------------
                     2             4            6             8        10
 -211.208 + 528.325 x  - 484.596 x  + 194.543 x  - 30.7152 x  + 1. x
```

A.12 Transfer Function with Minimal Q-Factors

Miroslav D. Lutovac, Dejan V. Tosic and Brian L. Evans
lutovac@iritel.bg.ac.yu tosic@galeb.etf.bg.ac.yu bevans@ece.utexas.edu

A.12.1 References

1. D.M.Rabrenovic, M. D. Lutovac,
"Elliptic filters with minimal Q-factors,"
Electronics Letters, vol. 30, no. 3, pp. 206–207, Feb. 1994.

2. M. D. Lutovac, D. V. Tosic and B. L. Evans,
"Advanced Filter Design for Signal Processing using MATLAB and Mathematica,"
http://iritel.iritel.bg.ac.yu/˜lutovac/www/afdhome.htm
http://galeb.etf.bg.ac.yu/˜tosic/afdhome.htm
http://www.ece.utexas.edu/˜bevans/

A.12.2 Initialization

```
SetDirectory[HomeDirectory[]];
<<afd\math\m\clearall.m
<<afd\math\m\erfdf.m
<<afd\math\m\erfez.m
<<afd\math\m\erfzh.m
<<afd\math\m\erf235.m
<<graphics`graphics`
```

```
Discrimination Factor L(n,a)
n = 1,2,3,4,6,8,9,12,16,18
Exact Formulas for Zeros: n=2^i 3^j
X(n,a)={list-of-zeros}, X(n,a,i)
n = 1,2,3,4,6,8,9,12,16,18
Zeta Function Z(n,a,e)
Elliptic rational functions: n=2^i 3^j
Relliptic(n,a,x)
n = 1,2,3,4,5,6,8,9,10,12,15,16,18
```

A.12.3 Notation

a – selectivity factor
e – ripple factor
n – order
L(n,a) – discrimination factor
H - normalized lowpass transfer function
HminQ - minQ normalized lowpass transfer function
Q(s) – Q-factor of a pole s
QminQ(n,a) – list of pole Q-factors of minQ transfer function
QminQ(n,a,i) – pole Q-factor of minQ transfer function
Relliptic(n,a,x) – elliptic rational function
s - normalized compex frequency
S(n,a,e) – list of transfer function poles

S(n,a,e,i) – transfer function pole
SminQ(n,a,e) – list of minQ transfer function poles
SminQ(n,a,e,i) – minQ transfer function pole
X(n,a) – list of zeros of elliptic rational function
X(n,a,i) – zero of elliptic rational function
w - normalized angular frequency
Z(n,a,e) – zeta function

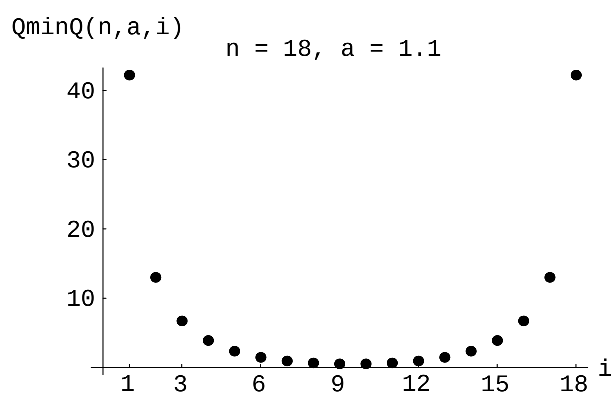
■ A.12.4 Definition and Procedures

```
Q[s_] := - Abs[s]/(2*Re[s]);

QminQ[n_Integer,a_,i_Integer] :=
 (a + X[n,a,i]^2)/
 (2 Sqrt[1- X[n,a,i]^2]*Sqrt[a^2 - X[n,a,i]^2]);

QminQ[n_Integer,a_] :=
 (a + X[n,a]^2)/
 (2 Sqrt[1- X[n,a]^2]*Sqrt[a^2 - X[n,a]^2]);

SminQ[n_Integer,a_,i_Integer] :=  Block[
{den,num,numim,numre,x,z},
 numre = -Sqrt[1- X[n,a,i]^2]*Sqrt[a^2- X[n,a,i]^2];
 numim = X[n,a,i]*(a + 1);
 num = numre + I*numim;
 den = a + X[n,a,i]^2;
 Sqrt[a]*num/den
];

SminQ[n_Integer,a_] :=  Block[
{den,num,numim,numre,x,z},
 numre = -Sqrt[1- X[n,a]^2]*Sqrt[a^2- X[n,a]^2];
 numim = X[n,a]*(a + 1);
 num = numre + I*numim;
 den = a + X[n,a]^2;
 Sqrt[a]*num/den
];

S[n_Integer, a_, e_, i_Integer] := Block[
{den,num,numim,numre,x,z},
 x = X[n,a,i];
 z = Z[n,a,e];
 numre = -z*Sqrt[1 - z^2]*Sqrt[1 - x^2]*Sqrt[1 - x^2/a^2];
 numim = x*Sqrt[1 - (1-1/a^2)*z^2];
 num = numre + I*numim;
 den = 1 - (1 - x^2/a^2)*z^2;
 num/den
];

S[n_Integer, a_, e_] :=  S[n,a,e,#]& /@ Range[n]
```

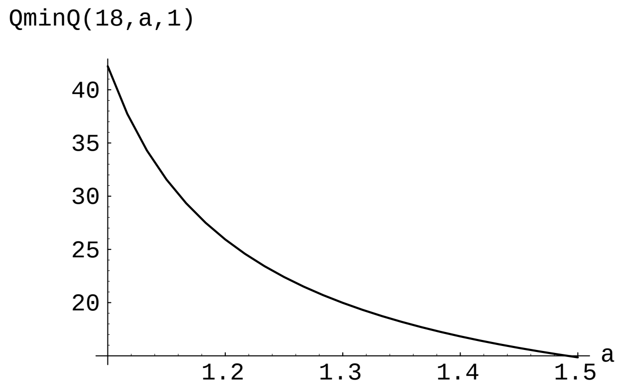
A.12.5 Example

```
n1 = 18;
a1 = 1.1;
e1 = 0.1;
i1 = 1;
S[n1,a1,e1,i1]
SminQ[n1,a1,i1]
Q[S[n1,a1,e1,i1]]
Q[SminQ[n1,a1,i1]]
QminQ[n1,a1,i1]
QminQ[n1,a1]
ListPlot[%
,PlotStyle->{AbsolutePointSize[4]}
,AxesLabel -> {"i", "QminQ(n,a,i)"}
,Ticks->{{1,3,6,9,12,15,18},{10,20,30,40}}
,PlotLabel -> "n = 18, a = 1.1"
];
```

```
-0.0053858 - 1.00386 I
-0.0124234 - 1.04874 I
93.1967
42.2111
42.2111
{42.2111, 13.0034, 6.71209, 3.88702, 2.34521, 1.45084, 0.930164, 0.642843, 0.515239,
 0.515239, 0.642843, 0.930164, 1.45084, 2.34521, 3.88702, 6.71209, 13.0034, 42.2111}
```




```
Plot[QminQ[18,a,1]
,{a,1.1,1.5}
,AxesLabel -> {"a", "QminQ(18,a,1)"}
];
```

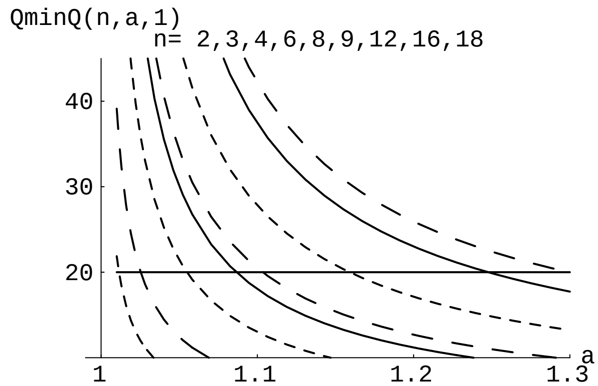


■ A.12.6 Find Selectivity Factor for Given Maximal Q-Factor

```
FindRoot[QminQ[18,a,1]==20, {a,1.1,1.5}]
```

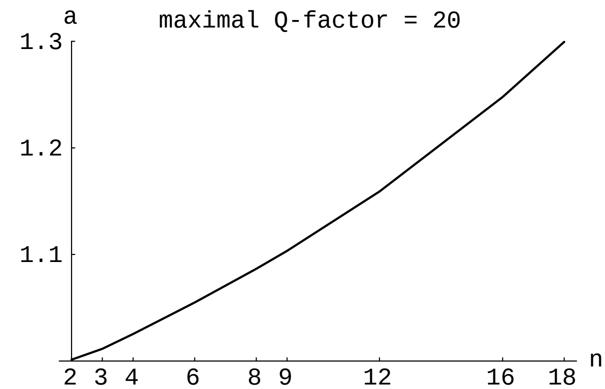
```
{a -> 1.29948}
```

```
Plot[{QminQ[18,a,1],
      QminQ[16,a,1],
      QminQ[12,a,1],
      QminQ[9, a,1],
      QminQ[8, a,1],
      QminQ[6, a,1],
      QminQ[4, a,1],
      20,
      QminQ[3,a,1],
      QminQ[2,a,1]}
,{a,1.01,1.3}
,PlotRange -> {{0.99,1.3},{10,45}}
,PlotStyle -> {Dashing[{.04}],
               Dashing[{}],
               Dashing[{.02}]}
,AxesLabel -> {"a", "QminQ(n,a,1)"}
,Ticks->{{1.0,1.1,1.2,1.3},{10,20,30,40}}
,PlotLabel -> "n= 2,3,4,6,8,9,12,16,18"
];
```



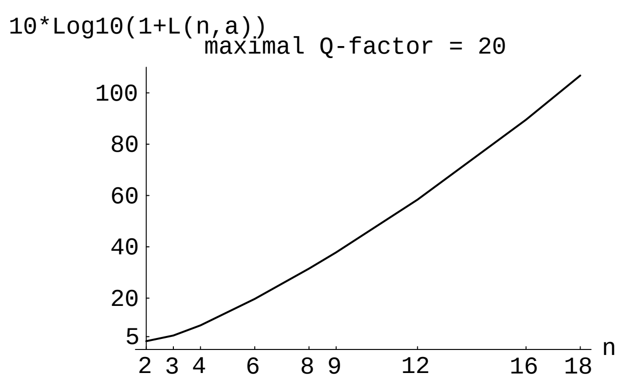

```
nminQ = {18,16,12,9,8,6,4,3,2};
aminQ = a /.
{FindRoot[QminQ[18,a,1]==20, {a,1.1,  1.3}],
 FindRoot[QminQ[16,a,1]==20, {a,1.1,  1.3}],
 FindRoot[QminQ[12,a,1]==20, {a,1.1,  1.3}],
 FindRoot[QminQ[9, a,1]==20, {a,1.1,  1.3}],
 FindRoot[QminQ[8, a,1]==20, {a,1.05, 1.1}],
 FindRoot[QminQ[6, a,1]==20, {a,1.04, 1.1}],
 FindRoot[QminQ[4, a,1]==20, {a,1.03, 1.1}],
 FindRoot[QminQ[3, a,1]==20, {a,1.02, 1.1}],
 FindRoot[QminQ[2, a,1]==20, {a,1.001,1.1}]
};
t = Transpose[{nminQ,aminQ}]
ListPlot[t
, PlotJoined -> True
, PlotRange -> {1,1.3}
, AxesLabel -> {"n", "a"}
, Ticks->{{2,3,4,6,8,9,12,16,18},{1,1.1,1.2,1.3}}
, PlotLabel -> "maximal Q-factor = 20"
];
```

```
{{18, 1.29948}, {16, 1.24774}, {12, 1.15906}, {9, 1.10338}, {8, 1.08654}, {6, 1.05495},
 {4, 1.02538}, {3, 1.01146}, {2, 1.00125}}
```




```
tL = Table[{nminQ[[i]],
            10*Log[10,1+L[nminQ[[i]],aminQ[[i]]]]}
     ,{i,1,9}
     ]
ListPlot[tL
, PlotJoined -> True
, PlotRange -> {0,110}
, AxesLabel -> {"n", "10*Log10(1+L(n,a))"}
, Ticks->{{2,3,4,6,8,9,12,16,18},{0,5,20,40,60,80,100}}
, PlotLabel -> "maximal Q-factor = 20"
];
```

```
{{18, 106.763}, {16, 89.5348}, {12, 58.386}, {9, 37.8317}, {8, 31.5114}, {6, 19.7043},
 {4, 9.39632}, {3, 5.4385}, {2, 3.23299}}
```

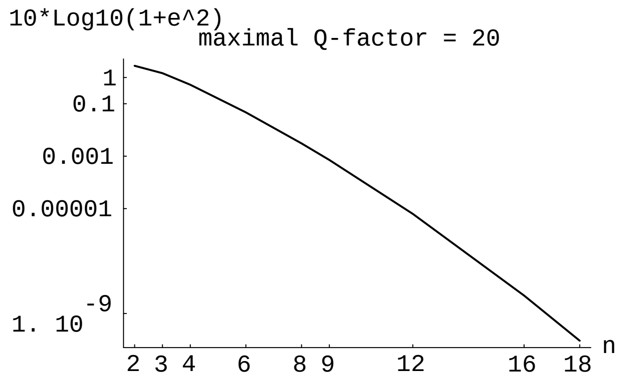


A.12.7 Ripple Factor e_minQ = 1 / Sqrt[L(n, a)]

```
te = Table[{nminQ[[i]],
            10*Log[10,1+1/L[nminQ[[i]],aminQ[[i]]]]}
     ,{i,1,9}
     ]
LogListPlot[te
```

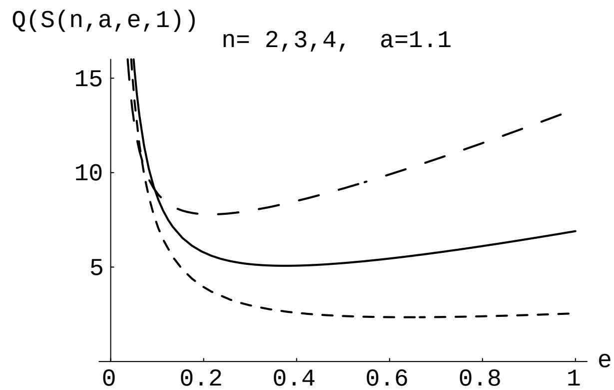
```
, PlotJoined -> True
, AxesLabel -> {"n", "10*Log10(1+e^2)"}
, Ticks->{{2,3,4,6,8,9,12,16,18},{1,.1,.001,.00001,.000000001}}
, PlotLabel -> "maximal Q-factor = 20"
];
```

```
                -11                    -9                   -6
{{18, 9.15089 10   }, {16, 4.83395 10  }, {12, 6.29779 10  }, {9, 0.00071557},
 {8, 0.0030676}, {6, 0.0467403}, {4, 0.530139}, {3, 1.46215}, {2, 2.79847}}
```



■ A.12.8 Pole Q-Factor in Terms of Ripple Factor

```
a1 = 1.1;
Plot[{Q[S[4,a1,e,1]],
      Q[S[3,a1,e,1]],
      Q[S[2,a1,e,1]]}
,{e,0.01,1}
,PlotRange -> {0,16}
,PlotStyle -> {Dashing[{.04}],
               Dashing[{}],
               Dashing[{.02}]}
,AxesLabel -> {"e", "Q(S(n,a,e,1))"}
,Ticks->{{0,0.2,0.4,0.6,0.8,1},{0,5,10,15}}
,PlotLabel -> "n= 2,3,4,  a=1.1"
];
```



■ A.12.9 Minimal Q-Factor Transfer Function

```
HminQ[n_,a_,s_] := Block[{e,i,K,t},
 e = 1/Sqrt[L[n,a]];
 If[EvenQ[n],
   K = Product[X[n,a,i]^2,{i,1,n/2}]/(Sqrt[1+e^2]*Sqrt[a^n]);
   t = K*Product[(s^2+(a^2/X[n,a,i]^2))/(s^2+(Sqrt[a]/QminQ[n,a,i])*s+a)
         ,{i,1,n/2}];,
   K = Product[X[n,a,i]^2,{i,1,(n-1)/2}]/((s+Sqrt[a])*Sqrt[a^(n-2)]);
   t = K*Product[(s^2+(a^2/X[n,a,i]^2))/(s^2+(Sqrt[a]/QminQ[n,a,i])*s+a)
         ,{i,1,(n-1)/2}];
 ];
 t
];
a1 = 1.1;
H1 = HminQ[1,a1,s]
H2 = HminQ[2,a1,s]
H3 = HminQ[3,a1,s]
H4 = HminQ[4,a1,s]
H6 = HminQ[6,a1,s]
H8 = HminQ[8,a1,s];
H9 = HminQ[9,a1,s];
```

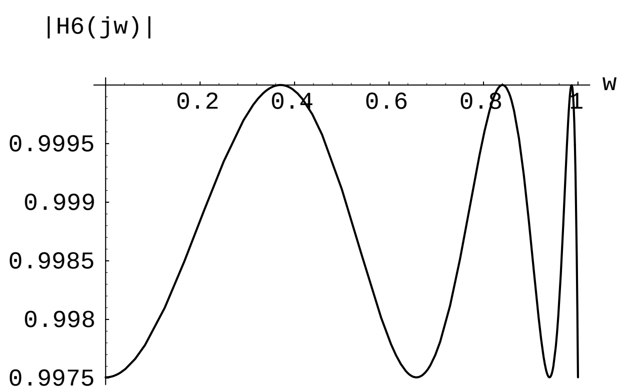
```
  1.04881
-----------
1.04881 + s
                    2
0.540094 (1.71408 + s )
-----------------------
                    2
 1.1 + 0.447214 s + s
```

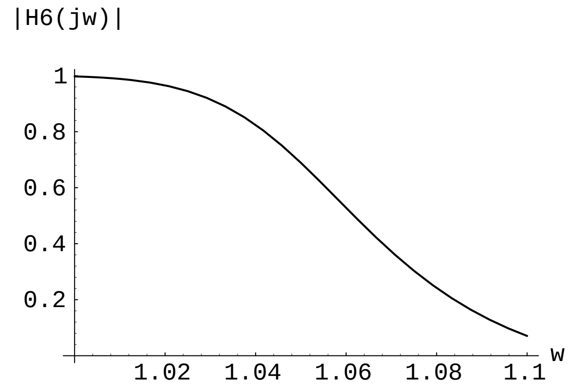
```
                                  2
       0.841917 (1.37031 + s )
-------------------------------------
                                         2
(1.04881 + s) (1.1 + 0.206892 s + s )
                         2              2
    0.210643 (1.29093 + s ) (4.34993 + s )
----------------------------------------------
                     2                       2
(1.1 + 0.134615 s + s ) (1.1 + 1.24828 s + s )
                              2              2              2
        0.0705794 (1.24336 + s ) (1.71408 + s ) (8.82646 + s )
-----------------------------------------------------------------------
                      2                        2                      2
(1.1 + 0.0806565 s + s ) (1.1 + 0.447214 s + s ) (1.1 + 1.63152 s + s )
```

```
H6w = H6 /. s->I*w;
```

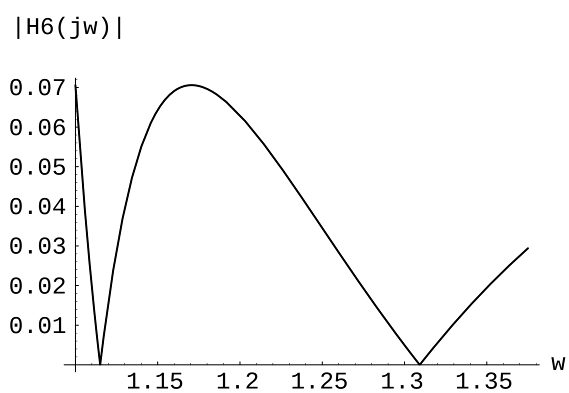
■ A.12.10 Magnitude Response in Terms of Angular Frequency

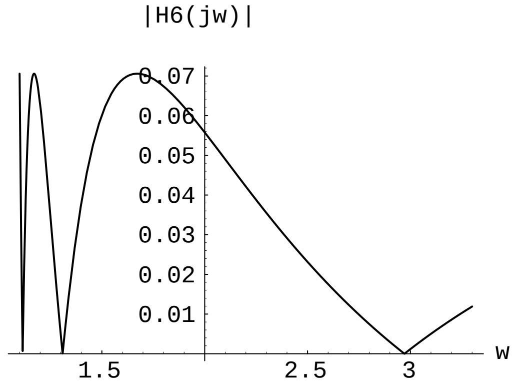
```
Plot[Abs[H6w]
,{w,0,1}
,AxesLabel -> {"w", "|H6(jw)|"}
];
Plot[Abs[H6w]
,{w,1,a1}
,AxesLabel -> {"w", "|H6(jw)|"}
];
```



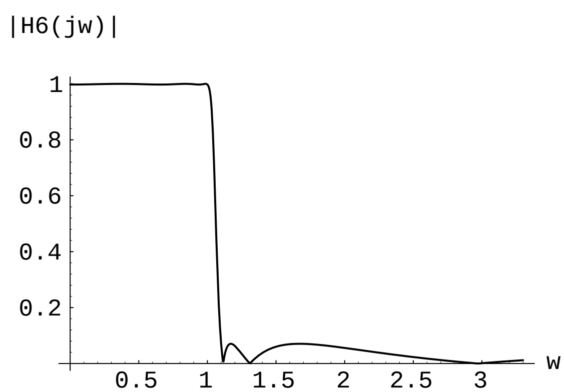


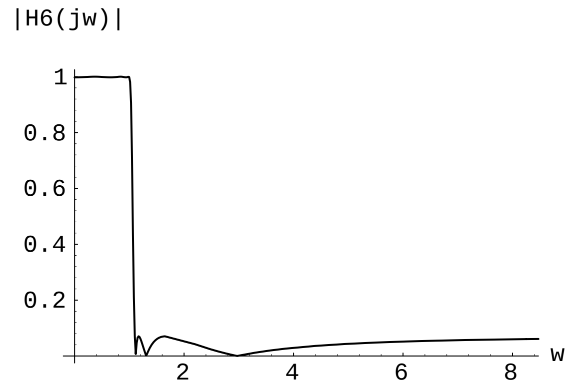

```
Plot[Abs[H6w]
,{w,a1,5*a1/4}
,AxesLabel -> {"w", "|H6(jw)|"}
];
Plot[Abs[H6w]
,{w,a1,3*a1}
,AxesLabel -> {"w", "|H6(jw)|"}
,PlotPoints->75
];
```





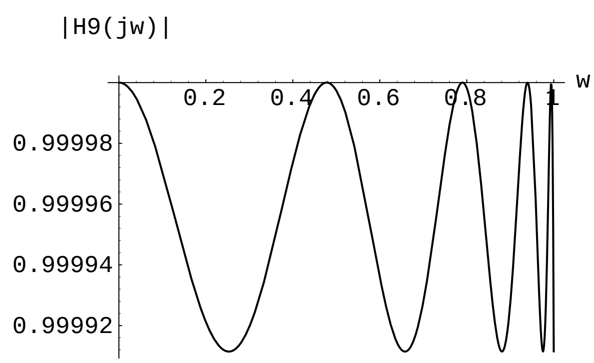

```
Plot[Abs[H6w]
,{w,0,3*a1}
,AxesLabel -> {"w", "|H6(jw)|"}
];
Plot[Abs[H6w]
,{w,0,12*a1}
,AxesLabel -> {"w", "|H6(jw)|"}
];
```



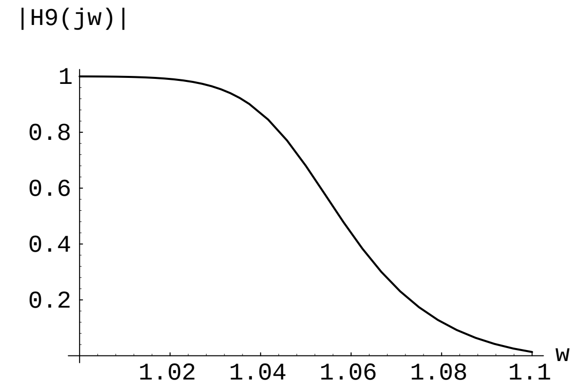



```
H9w = H9 /. s->I*w;
```

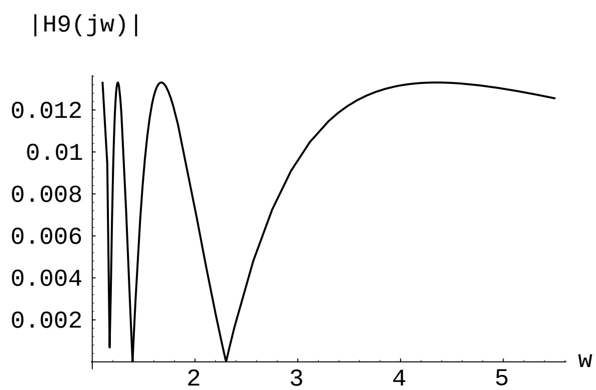
```
Plot[Abs[H9w]
,{w,0,1}
,AxesLabel -> {"w", "|H9(jw)|"}
];
```



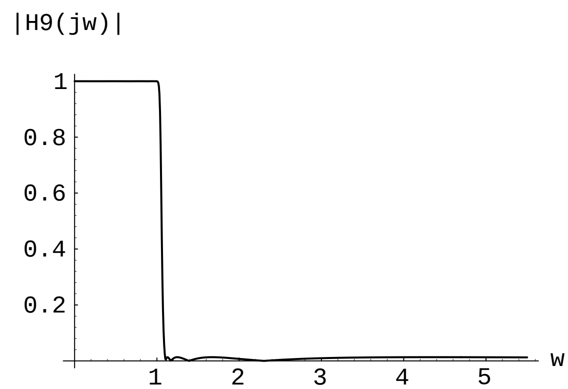

```
Plot[Abs[H9w]
,{w,1,a1}
,AxesLabel -> {"w", "|H9(jw)|"}
];
```




```
Plot[Abs[H9w]
,{w,a1,5*a1}
,AxesLabel -> {"w", "|H9(jw)|"}
];
```




```
Plot[Abs[H9w]
,{w,0,5*a1}
,AxesLabel -> {"w", "|H9(jw)|"}
];
```


|H9(jw)|
1
0.8
0.6
0.4
0.2
1
2
3
4
5
w