Digital State-Variable Filters

Helmut Keller

This paper provides an overview over all commonly used first- and second-order analog filter types and describes, step by step, the derivation of corresponding digital state-variable filters. The derived digital state-variable filters are bilinear-transformed versions of the analog filters. They are suitable for modulation of the filter parameters. Possible applications are e.g. wah-wah effects, emulations of voltage controlled filters of analog synthesizers, or dynamic equalizers.

1. Introduction

Analog filters and digital infinite impulse response (IIR) filters are close “cousins”. Filters with arbitrary order can be composed as a series connection of second-order sub-filters. Four different direct forms are commonly used for the digital filters. The digital filters can be derived from analog prototype filters via the bilinear transformation. The relationship between the analog and the digital filter coefficients is, however, rather complicated. If the filter parameters need to change over time, the digital direct forms are not suitable whereas an analog state-variable filter is. The digital state-variable filters derived in this paper are suitable for time-varying filter parameters, as well, while they are still bilinear transformed versions of the original analog filters. However, they have nearly the same filter coefficients as the original analog filters. A disadvantage of the digital state-variable filters is their increased resource consumption compared to the direct forms. Thus, the direct forms are still the first choice for static digital filters.

2. First-order analog filter

The complex transfer function $H(f)$ of a first-order analog filter is defined in equation (1). In this paper, f is the frequency of interest, f_0 is the filter pole-frequency, and j is the square root of minus one.

$$H(f)=\frac{b_1+b_0\frac{\mathrm{j}f}{f_0}}{1+\frac{\mathrm{j}f}{f_0}} \tag{1}$$

Table 1 shows the coefficients of common first-order filter types.

Table 1: Coefficients of common first-order filter types

Type	f_0	b_0	b_1
Flat	> 0 Hz	1	1
Lowpass	f_c	0	1
Highpass	f_c	1	0
Allpass	$\frac{1}{\pi\,\tau_d}$	1	-1
Low-shelf	$\frac{f_s}{A}$	1	A^2
High-shelf	$A f_s$	A^2	1

The parameter f_c is the cutoff frequency of the low- and highpass filters.

The parameter τ_d is the group delay of the allpass filter at frequencies much lower than f_0.

The parameter A of the low-shelf filter is the gain at the center frequency f_s of its slope. The gain at frequencies much lower than f_s is A^2.

The parameter A of the high-shelf filter is the gain at the center frequency f_s of its slope. The gain at frequencies much higher than f_s is A^2.

There are other possible definitions of first-order shelf filters too. The definitions used in this paper seem to be the most reasonable ones and are popular in modern audio applications.

3. Second-order analog filter

The complex transfer function of a second-order analog filter is defined as:

$$H(f)=\frac{b_2+b_1\frac{1}{Q}\frac{\mathrm{j}f}{f_0}-b_0\frac{f^2}{f_0^2}}{1+\frac{1}{Q}\frac{\mathrm{j}f}{f_0}-\frac{f^2}{f_0^2}} \qquad (2)$$

Table 2 shows the coefficients of common second-order filter types.

Table 2: Coefficients of common-second order filter types

Type	f_o	Q	b_0	b_1	b_2
Flat	> 0 Hz	> 0	1	1	1
Lowpass	f_c	Q_c	0	0	1
Highpass	f_c	Q_c	1	0	0
Bandpass	f_c	Q_c	0	1	0
Notch	f_c	Q_c	1	0	1
Allpass	$\frac{1}{\pi Q_c \tau_d}$	Q_c	1	-1	1
Peak equalizer	f_c	$A\,Q_c$	1	A^2	1
High-shelf	$\frac{f_s}{\sqrt{A}}$	$\frac{1}{\sqrt{\left(A+\frac{1}{A}\right)\left(\frac{1}{L}-1\right)+2}}$	A^2	A	1
Low-shelf	$\sqrt{A} f_s$	$\frac{1}{\sqrt{\left(A+\frac{1}{A}\right)\left(\frac{1}{L}-1\right)+2}}$	1	A	A^2
Tone stack	f_c	≤ 0.5	T	M	B
Eliptic lowpass	f_c	Q_c	$\frac{f_c^2}{f_n^2}$	0	1
Eliptic highpass	f_c	Q_c	1	0	$\frac{f_n^2}{f_c^2}$
Lowpass with 20 dB / decade slope	f_c	Q_c	0	Q_c	1
Highpass with 20 dB / decade slope	f_c	Q_c	1	Q_c	0

The parameter f_c is the cut off frequency of the low- or highpass filters and the center frequency of the bandpass, notch, peak equalizer and tone stack filters.

The parameter Q_c is the quality factor of the filters.

The parameter τ_d is the group delay of the allpass filter at frequencies much lower than f_0. The group delay of a second order analog allpass filter is optimal flat with $Q_c = 1/\sqrt{3}$.

The parameter A of the peak equalizer filter is the gain at f_l and f_h. Both frequencies are defined by equations 5 and 6. The gain at f_c is A^2.

The parameter A of the low-shelf filter is the gain at the center frequency f_s of its slope. The gain at frequencies much lower than f_c is is A^2.

The parameter *A* of the high-shelf filter is the gain at the center frequency f_s of its slope. The gain at frequencies much higher than f_s is A^2.

The parameter *L* controls the steepness of the shelf filters slopes. The filters degrade to first-order shelf filters with $L = 0.5$.

The parameters *B*, *M* and *T* of the tone stack filter are the gains at frequencies much lower than, in the range of and much higher than f_c.

The parameter f_n of the eliptic filters is their notch frequency.

The low- and highpass filters with a 20 dB / decade slope degrade to first-order low- and highpass filters with $Q_c = 0.5$.

There are other possible definitions of second-order shelf and peak equalizer filters too. The definitions used in this paper seem to be the most reasonable ones and are quite popular in modern audio applications.

4. Relations between bandwidth and quality factor

The bandwidth *bw* of a band pass, notch or peak equalizer filter is defined as:

$$bw = \frac{f_c}{Q_c} = f_h - f_l \tag{3}$$

with

$$f_c = \sqrt{f_h f_l} \tag{4}$$

with

$$f_l = \frac{\sqrt{1+4Q_c^2} - 1}{2Q_c} f_c \tag{5}$$

and

$$f_h = \frac{\sqrt{1+4Q_c^2} + 1}{2Q_c} f_c \tag{6}$$

The logarithmic bandwidth *lbw* in fractions of tenths of a decade is defined as:

$$lbw = \frac{10}{\ln(10)} \ln\left(\frac{f_h}{f_l}\right) \tag{7}$$

With a given *lbw* we obtain:

$$Q_c = \frac{1}{2\sinh\left(\frac{\ln(10)}{20} lbw\right)} \tag{8}$$

5. Combination of two first-order analog filters

The series connection of a first first-order analog filter with coefficients f_{10}, b_{10} and b_{11} and a second first-order analog filter with coefficients f_{20}, b_{20} and b_{21} results in a second-order filter with the coefficients of table 3.

Table 3: *Coefficients for two first-order filters*

f_0	Q	b_0	b_1	b_2
$\sqrt{f_{10} f_{20}}$	$\frac{f_0}{f_{10}+f_{20}}$	$b_{10} b_{20}$	$\frac{b_{11} f_{20}+b_{21} f_{10}}{f_{10}+f_{20}}$	$b_{11} b_{21}$

6. Analog filters of arbitrary order

Every analog filter of even order N can be composed as a series connection of N / 2 second-order filters. Every analog filter of odd order N can be composed as a series connection of (N + 1) / 2 second-order filters if an additional flat first-order filter is inserted into the filter chain. Setting f_{20} of the additional flat filter to f_{10} of the necessary first-order filter, we obtain the coefficients of table 4.

Table 4: *Coefficients for a single first-order filter*

f_0	Q	b_0	b_1	b_2
f_{10}	0.5	b_{10}	$\frac{b_{11}+1}{2}$	b_{11}

Thus we can compose every analog filter as a series connection of second-order analog filters.

7. Bilinear transformation and frequency warping

We obtain a bilinear-transformed digital IIR filter with the sample period T from the analog filter with the following substitution:

$$\mathrm{j} f \Rightarrow \frac{1}{\pi T} \frac{z-1}{z+1} \tag{9}$$

with

$$z = \mathrm{e}^{\mathrm{j} 2\pi f T} \tag{10}$$

The frequency f_a of the analog filter relates to the frequency f_d of the bilinear-transformed digital filter as shown in equation 11.

$$f_\mathrm{a} = \frac{\tan\left(\pi T f_\mathrm{d}\right)}{\pi T} \tag{11}$$

The magnitude and the phase of the bilinear-transformed digital filter are identical to the original analog filter except of the nonlinear warping of the frequency axis.

In order to project a certain calculated frequency f_x of the analog filter to the same frequency in the digital filter, the analog filter must be designed with the prewarped calculated frequency f_{wx}. This approach requires $f_x < 0.5 / T$:

$$f_{wx} = \frac{\tan(\pi T f_x)}{\pi T} \tag{12}$$

8. Preserving bandwidth

The bandwidth of a second-order bandpass, notch, or peak equalizer filter is approximately preserved in the bilinear-transformed digital filter by substituting the calculated quality Q_c of the analog filter with the prewarped calculated quality Q_{wc}:

$$Q_{wc} = \frac{1}{2 \sinh\left(\frac{\pi T f_c}{\sin(2 \pi T f_c)} \ln\left(\frac{\sqrt{1+4 Q_c^2}+1}{\sqrt{1+4 Q_c^2}-1} \right) \right)} \tag{13}$$

Alternatively, the band confines f_l and f_h – and thus also the bandwidth – are exactly preserved if f_l and f_h are substituted with f_{wl} and f_{wh}. In this case the center frequency f_c is preserved approximately only. This approach requires $f_h < 0.5 / T$:

$$f_{wl} = \frac{\tan(\pi T f_l)}{\pi T} \tag{14}$$

$$f_{wh} = \frac{\tan(\pi T f_h)}{\pi T} \tag{15}$$

$$f_{wc} = \sqrt{f_{wl} f_{wh}} \tag{16}$$

$$Q_{wc} = \frac{f_{wc}}{f_{wh} - f_{wl}} \tag{17}$$

9. Digital optimally flat group-delay filters

The phase and group delay of a bilinear-transformed filter are transformed differently compared to magnitude and phase. Therefore a bilinear-transformed analog optimal flat group-delay filter does not necessarily result in a digital optimally flat group-delay filter. Digital filters with orders higher than two must be optimized for an optimally flat group-delay in the digital frequency domain directly (Thiran allpass filters). First-order filters do not need any optimization because they have always optimally flat group delay. Digital second-order optimally flat group-delay filters are obtained by the bilinear transformation of an analog second-order filter with the quality factor Q_c from equation 19.

$$Q_c = \sqrt{\frac{1 - \left(\frac{T}{\tau_d} \right)^2}{3}} \tag{19}$$

The group delay of analog optimally flat group-delay allpass filters is τ_d at frequencies much lower than f_0, and decreases for frequencies much higher than f_0.

The group delay of digital optimally flat group-delay allpass filters of order N behaves similar if $\tau_d < NT$. The group delay is, however, constant for all frequencies if $\tau_d = NT$. The group delay even increases at high frequencies if $\tau_d < NT$. The group delay at low frequencies must satisfy $\tau_d > (N-1)T$.

10. Analog second-order state-variable filter

The analog second-order state-variable filter uses two integrators with the following complex transfer function:

$$H(f) = \frac{f_0}{\mathrm{j}f} \tag{20}$$

The integrators have unity gain at f_0. Fig. 1 shows a possible realization of the analog second-order state-variable filter.

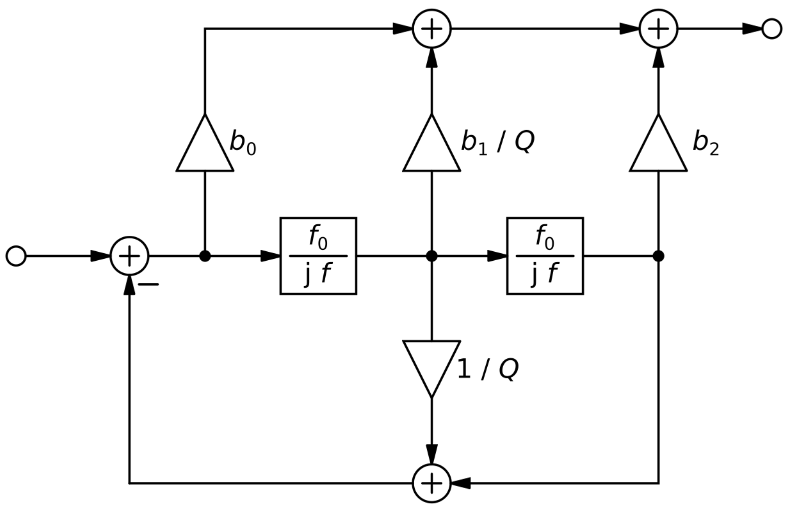

Fig. 1. Analog second-order state-variable filter

11. Digital second-order state-variable filter

A substitution of the analog integrators in the analog state-variable filter by digital integrators as defined in equation 21 would result in the bilinear-transformed digital state-variable filter because equation 21 is just an rearrangement of equation 9.

$$\frac{f_0}{\mathrm{j} f} \Rightarrow k \frac{z+1}{z-1} \qquad (21)$$

with

$$k = \pi f_0 T \qquad (22)$$

However, this substitution would lead to delay free feedback loops and thus is not realizable. Using the transposed direct form II of the digital integrators it is possible to avoid the delay free feedback loops. For this the feedback signals are tapped at the sample delay outputs instead of at the integrator outputs. This modification requires however some additional modification in digital filter. Fig. 2 shows a possible realization of the bilinear-transformed digital second-order state-variable filter.

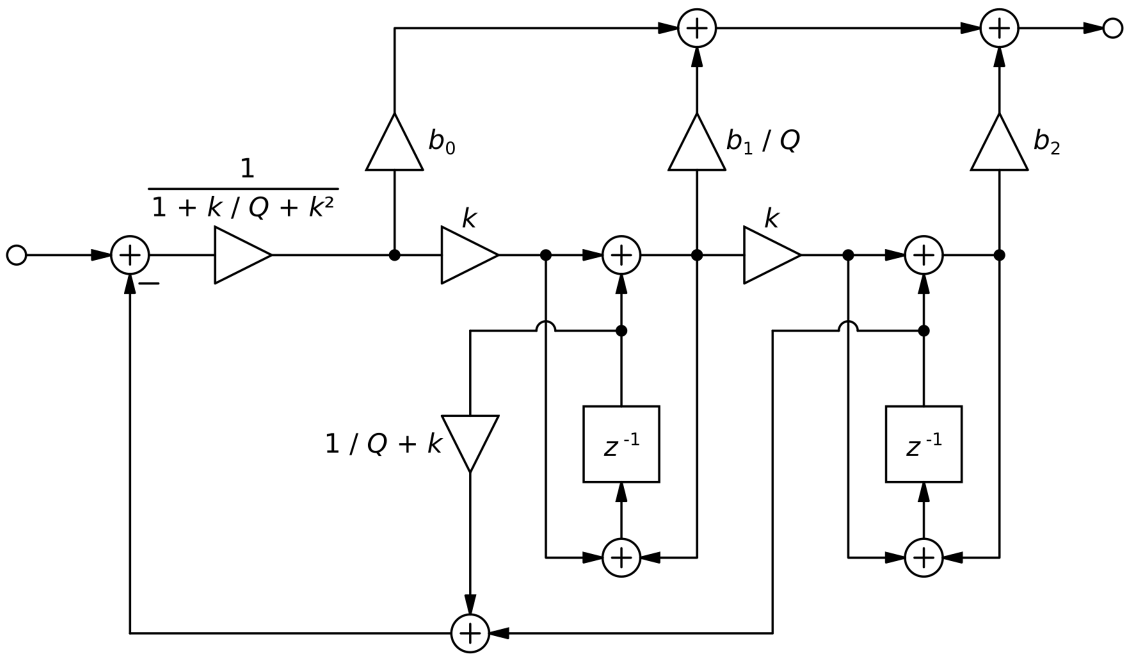

Fig. 2. Digital second-order state-variable filter

Prewarping can be applied if desired by using the f_0 or Q values of the prewarped analog filter.

The following pseudo code describes the calculation of the bilinear-transformed digital second-order state-variable filter step by step:

- Initialize the two filter states s1 and s2 with zeros
- Initialize the filter parameters f0, Q, b0, b1 and b2 with default values
- Initialize T = T / s according to the sample rate
- Initialize pit = 4 * atan(1) * T
- Read the actual filter parameters in real time:

  f0 = f_0 / Hz
  Q = Q
  b0 = b_0
  b1 = b_1
  b2 = b_2

- Calculate internal filter parameters at initialization and if f_0, Q or b_1 have changed:

  k = pit * f0
  kq = 1 / Q
  kdiv = 1 / (1+ k * (kq + k))
  kf = kq + k
  kb1= kq * b1

- Calculate the output sample y for each input sample x:

  hp = kdiv * (x – kf * s1 – s2)
  aux = k * hp
  bp = aux + s1
  s1 = aux + bp
  aux = k * bp
  lp = aux + s2
  s2 = aux + lp
  y = b0 * hp + kb1 * bp + b2 * lp

12. Digital first-order state-variable filter

A first-order digital state-variable filter is not necessary because it was shown that it can be realized by a second-order digital state-variable filter. However, in order to reduce memory and performance requirements, a first-order digital state-variable filter makes sense, after all. The following figure shows a possible realization of the bilinear-transformed digital first-order state-variable filter:

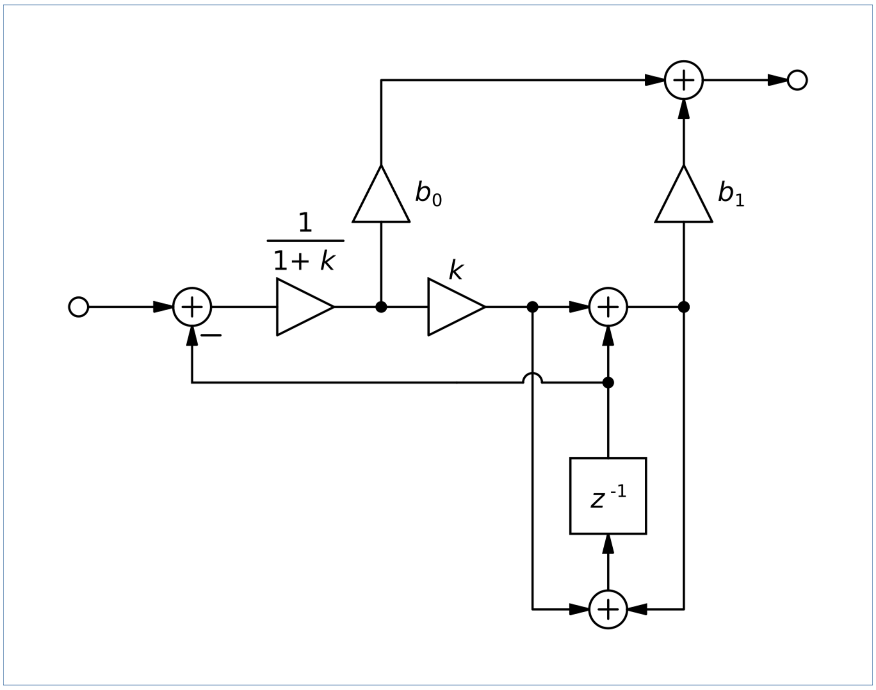

Fig. 3. *Digital first-order state-variable filter*

Prewarping can be applied if desired by using f_0 of the prewarped analog filter.

The following pseudo code describes the calculation of the bilinear-transformed digital first-order variable filter step by step:

- Initialize the filter state s1 with zeros
- Initialize the filter parameters f0, b0 and b1 with default values
- Initialize T = *T* / s according to the sample rate
- Initialize pit = 4 * atan(1) * T
- Read the actual filter parameters in real time:

  f0 = f_0 / Hz
  b0 = b_0
  b1 = b_1

- Calculate internal filter parameters at initialization and if f_0 has changed:

  k = pit * f0
  kdiv = 1 / (1+ k)

- Calculate the output sample y for each input sample x:

  hp = kdiv * (x – s1)
  aux = k * hp
  lp = aux + s1
  s1 = aux + lp
  y = b0 * hp + b1 * lp

13. Filter parameter smoothing

So far we know how to calculate the filter even if the filter parameters would change with the audio sample rate. In many applications filter parameters do not change as fast. The control data are typically received through a lower-priority thread at much lower rates. A typical example is the control of the resonance frequency of a second-order lowpass filter for a wah-wah-like effect by a MIDI foot controller. As a consequence the control data could lead to big leaps in the parameter values, possibly resulting in annoying audible transients. Here, the ideal solution would be to apply a first-order lowpass filter with a time constant τ_{smooth} of some 10 ms between the control data and the filter parameter calculation with the audio sample rate. However, there is a more efficient approach to achieve similar results:

The filter parameters are only updated if the control data has changed. A first-order lowpass filter is inserted between all slowly updated filter parameters p_{xu} and the filter parameters p_x the latter being actually used in the filter and updated with the audio sample rate.

The following pseudo code describes the calculation of the desired smoothing low pass filters step by step. The selected low pass structure requires minimal resources:

- Initialize all px and pxu with the default values of p_x
- Initialize all smoothing filter states sx with px
- Initialize T = T / s according to the sample rate
- Initialize tau = τ_{smooth} / s according to the desired smoothing time constant
- Initialize ksmooth = T / tau
- Calculate pxu if new control data is available
- Calculate all px with the audio sample rate:

```
px = ksmooth * (pxu - sx) + sx
sx = px
```

For the digital second-order state-variable filter the pseudo code filter parameters b0, kb1, b1, k, kdiv and kf need to be smoothed. For the digital first-order state-variable filter the pseudo code filter parameters b0, b1, k and kdiv need to be smoothed. Note that this effective smoothing approach could lead in theory to temporarily unstable filters. In practice this is, however, not an issue. This effective smoothing approach could be applied on the coefficients of the direct forms of the digital filters, as well. However, in this case temporarily unstable filters are likely to be an issue, and the calculation of the filter-coefficient updates would require much more computational resources. The direct forms of digital filters are therefore not suitable for time varying filters.

14. Summary

Digital second-order state-variable filters can be easily derived from the analog filter models. The substitution of the analog integrators by the digital integrators results by design in the bilinear-transformed digital filter. Three of the analog filter parameters, namely b_0, b_1 and b_2, remain identical in both the digital and the analog filter. The analog filter parameters f_0 and Q may require some prewarping due to the frequency warping of the bilinear transformation. Some additional simple corrections in the feedback loop of the digital filter are also necessary. The digital filter parameters derived from f_0 and Q are thus very close but not identical to the analog filter parameters. They could be calculated with moderate computational cost even at the audio sample rate. Since filter-parameter smoothing is required in most applications anyway, the filter parameters need to be updated even much less often. An effective pragmatic approach for the filter parameter smoothing has been proposed.

The embedding of first-order filters in second-order filters has been described. Alternatively, the newly introduced digital first-order state-variable filter could be used.

The bilinear transformation is not suitable for optimally flat group-delay filters of orders higher than one. However, for second-order filters an appropriate prewarping of the quality factor has been introduced.

15. Literature

[1] A. Wishnick, “Time-Varying Filters For Musical Applications”, Proc. of the 17th Int. Conference on Digital Audio Effects (DAFx-14), Erlangen, Germany, September 1-5, 2014

[2] V. Lazzarini, J.Timoney, “Improving the Chamberlin Digital State-variable Filter”, Computer Science, ArXiv, November 2021