# Second-Order Bandpass/Bandreject Filter Design

Yangang Cao

February 15, 2019

1. Definition of Bandpass/Bandreject Filter

2. Second-Order Allpass Filter

3. Second-Order Bandpass/Bandreject Filter
   - second-order bandpass filter
   - second-order bandreject filter

1. Definition of Bandpass/Bandreject Filter

2. Second-Order Allpass Filter

3. Second-Order Bandpass/Bandreject Filter
   - second-order bandpass filter
   - second-order bandreject filter

Definition of Bandpass/Bandreject filter:

- **Bandpass (BP)** filters select frenquencies between a lower cut-off frenquency $f_{cl}$ and a higher cut-off frenquency $f_{ch}$. Frenquencies below $f_{cl}$ and frenquencies higher than $f_{ch}$ are attenuated.
- **Bandreject (BR)** filters attenuate frenquencies between a lower cut-off frenquency $f_{cl}$ and a higher cut-off frenquency $f_{ch}$. Frenquencies below $f_{cl}$ and frenquencies higher than $f_{ch}$ are passed.

A second-order allpass filter is given by the transfer function

$$A(z) = \frac{-c + d(1-c)z^{-1} + z^{-2}}{1 + d(1-c)z^{-1} - cz^{-2}}$$

$$c = \frac{\tan(\pi f_b/f_S) - 1}{\tan(\pi f_b/f_S) + 1}$$

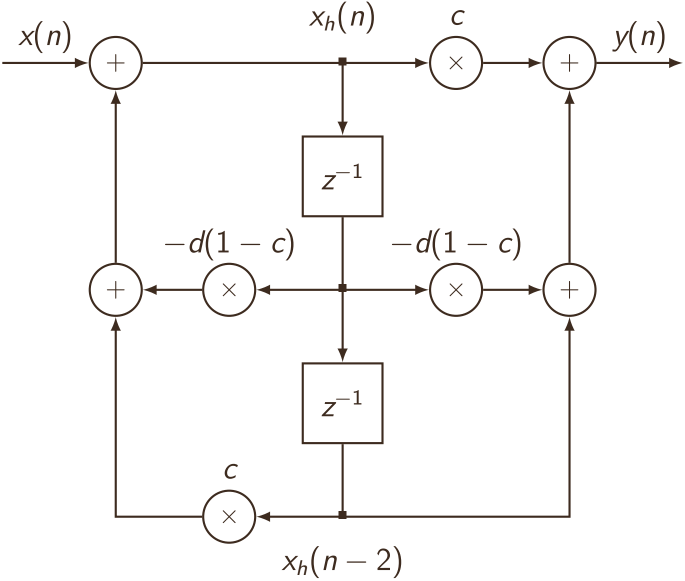
$$d = -\cos(2\pi f_c/f_S),$$

and the corresponding difference equations

$$x_h(n) = x(n) - d(1-c)x_h(n-1) + cx_h(n-2)$$

$$y(n) = -cx_h(n) + d(1-c)x_h(n-1) + x_h(n-2).$$

## Block diagram of second-order allpass filter



Corresponding state and output equations are:

$$\begin{bmatrix} x_h(n) \\ x_h(n-1) \end{bmatrix} = \begin{bmatrix} -d(1-c) & c \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x_h(n-1) \\ x_h(n-2) \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} x(n)$$

$$y(n) = \begin{bmatrix} (1-c^2)d & 1-c^2 \end{bmatrix} \begin{bmatrix} x_h(n-1) \\ x_h(n-2) \end{bmatrix} + (-c)x(n).$$

Matlab code:

```
function y = secondallpass(audio, para)
% y = secondallpass(audio, para)
% Author: Yangang Cao
% Applies a allpass filter to the input signal.
% para(1) is the normalized center frequency ...
    in (0,1), i.e. 2*fc/fS.
% para(2) is the normalized bandwidth in (0,1) ...
    i.e. 2*fb/fS.
c = (tan(pi*para(2)/2)-1) / ...
    (tan(pi*para(2)/2)+1);
d = -cos(pi*para(1));
x = [0; 0];
x_1 = 0;
A = [-d*(1-c), c; 1, 0];
B = [1; 0];
C = [d*(1-c^2), 1-c^2];
D = -c;
for n=1:length(audio)
    x_1 = A * x + B * audio(n);
    y(n) = C * x + D * audio(n);
    x = x_1;
end
```

1. Definition of Bandpass/Bandreject Filter

2. Second-Order Allpass Filter

3. Second-Order Bandpass/Bandreject Filter
   - second-order bandpass filter
   - second-order bandreject filter

Second-order bandpass and bandreject filters can be described by the following transfer function

$$H(z) = \frac{1}{2}[1 \mp A(z)] \quad (BP/BR - /+)$$

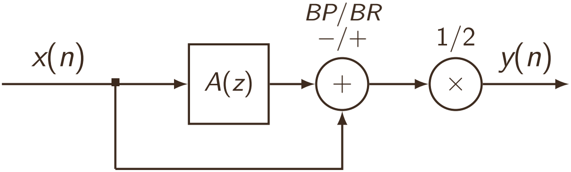
$$A(z) = \frac{-c + d(1-c)z^{-1} + z^{-2}}{1 + d(1-c)z^{-1} - cz^{-2}}$$

$$c = \frac{\tan(\pi f_b/f_S) - 1}{\tan(\pi f_b/f_S) + 1}$$

$$d = -\cos(2\pi f_c/f_S),$$

where a tunable second-order allpass $A(z)$ with tuning parameters $c$ and $d$ is used. The minus sign (-) denotes the bandpass operation and the plus sign (+) the bandreject operation.

Block diagram of second-order bandpass/bandreject filter



The difference equations of second-order bandpass filter are:

$$x_h(n) = x(n) - d(1-c)x_h(n-1) + cx_h(n-2)$$

$$y(n) = \frac{1+c}{2}x_h(n) - \frac{1+c}{2}x_h(n-2),$$

and corresponding state and output equations are:

$$\begin{bmatrix} x_h(n) \\ x_h(n-1) \end{bmatrix} = \begin{bmatrix} -d(1-c) & c \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x_h(n-1) \\ x_h(n-2) \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} x(n)$$

$$y(n) = \begin{bmatrix} \frac{d(c^2-1)}{2} & \frac{c^2-1}{2} \end{bmatrix} \begin{bmatrix} x_h(n-1) \\ x_h(n-2) \end{bmatrix} + \frac{1+c}{2}x(n)$$

## Matlab code:

```matlab
function y = apbandpass(audio, para)
% y = apbandpass(audio, para)
% Author: Yangang Cao
% Applies a bandpass filter to the input signal.
% para(1) is the normalized center frequency ...
    in (0,1), i.e. 2*fc/fS.
% para(2) is the normalized bandwidth in (0,1) ...
    i.e. 2*fb/fS.
c = (tan(pi*para(2)/2)-1) / ...
    (tan(pi*para(2)/2)+1);
d = -cos(pi*para(1));
x = [0; 0];
x_1 = 0;
A = [-d*(1-c), c; 1, 0];
B = [1; 0];
C = [d*(c^2-1)/2, (c^2-1)/2];
D = (1+c)/2;
for n=1:length(audio)
    x_1 = A * x + B * audio(n);
    y(n) = C * x + D * audio(n);
    x = x_1;
end
```

The difference equations of second-order bandreject filter are:

$$x_h(n) = x(n) - d(1-c)x_h(n-1) + cx_h(n-2)$$

$$y(n) = \frac{1-c}{2}x_h(n) + d(1-c)x_h(n-1) + \frac{1-c}{2}x_h(n-2),$$

and corresponding state and output equations are:

$$\begin{bmatrix} x_h(n) \\ x_h(n-1) \end{bmatrix} = \begin{bmatrix} -d(1-c) & c \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x_h(n-1) \\ x_h(n-2) \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} x(n)$$

$$y(n) = \begin{bmatrix} \frac{d(1-c^2)}{2} & \frac{1-c^2}{2} \end{bmatrix} \begin{bmatrix} x_h(n-1) \\ x_h(n-2) \end{bmatrix} + \frac{1-c}{2}x(n)$$

Matlab code:

```
function y = apbandreject(audio, para)
% y = apbandreject(audio, para)
% Author: Yangang Cao
% Applies a bandreject filter to the input ...
    signal.
% para(1) is the normalized center frequency ...
    in (0,1), i.e. 2*fc/fS.
% para(2) is the normalized bandwidth in (0,1) ...
    i.e. 2*fb/fS.
c = (tan(pi*para(2)/2)-1) / ...
    (tan(pi*para(2)/2)+1);
d = -cos(pi*para(1));
x = [0; 0];
x_1 = 0;
A = [-d*(1-c), c; 1, 0];
B = [1; 0];
C = [d*(1-c^2)/2, (1-c^2)/2];
D = (1-c)/2;
for n=1:length(audio)
    x_1 = A * x + B * audio(n);
    y(n) = C * x + D * audio(n);
    x = x_1;
end
```