Direct Form Filters

Young Won Lim
12/14/23

Please send corrections (or suggestions) to youngwlim@hotmail.com.
This document was produced by using LibreOffice.

Young Won Lim
12/14/23

Based on

Introduction to Signal Processing

S. J. Ofranidis

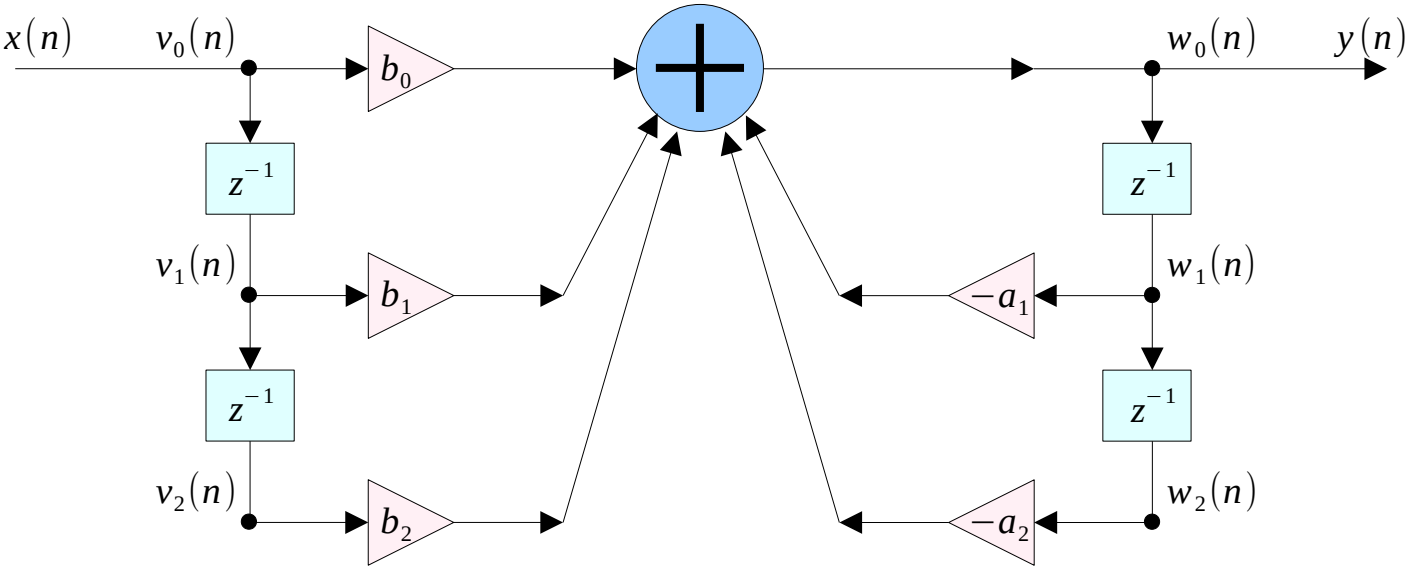
The necessities in DSP C Programming

FIR Filter (A.pdf) 20191114

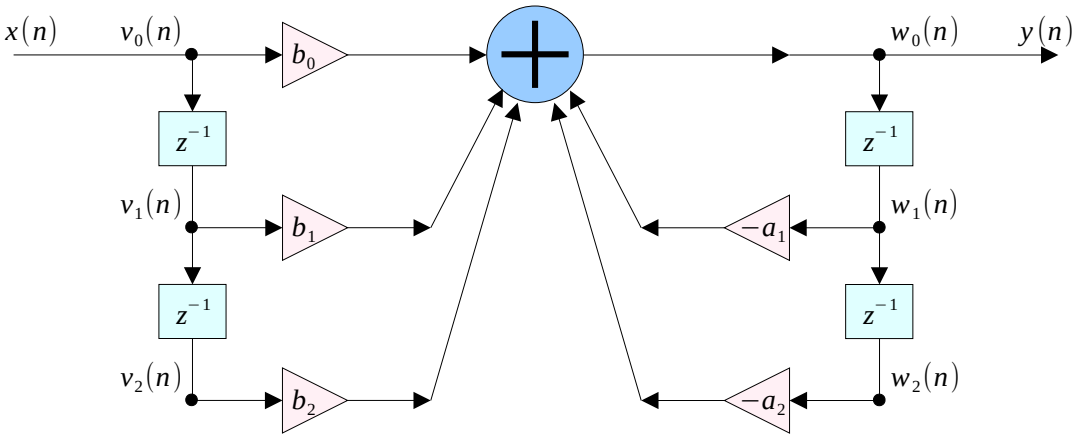
Direct Form

$$H(z) = \frac{N(Z)}{D(z)} = \frac{b_0 + b_1 z^{-1} + b_2^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

$$y_n = -a_1 y_{n-1} - a_2 y_{n-2} + b_0 x_n + b_1 x_{n-1} + b_2 x_{n-2}$$



Direct Form



$$v_0(n) = x(n)$$
$$v_1(n) = x(n-1) = v_0(n-1)$$
$$v_2(n) = x(n-2) = v_1(n-1)$$

$$w_0(n) = y(n)$$
$$w_1(n) = y(n-1) = w_0(n-1)$$
$$w_2(n) = y(n-2) = w_1(n-1)$$

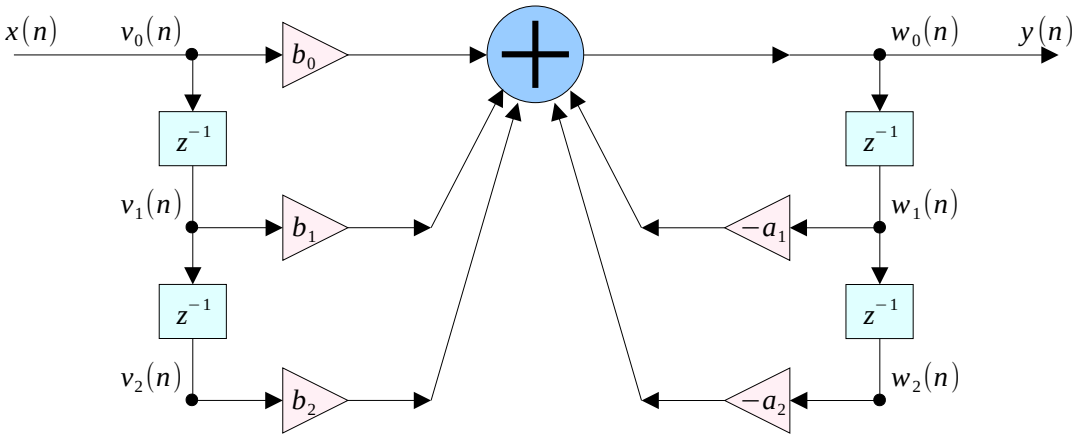
$$v_1(n+1) = v_0(n)$$
$$v_2(n+1) = v_1(n)$$

$$w_1(n+1) = w_0(n)$$
$$w_2(n+1) = w_1(n)$$

Direct Form

$$v_1(n+1) = v_0(n)$$

$$v_2(n+1) = v_1(n)$$



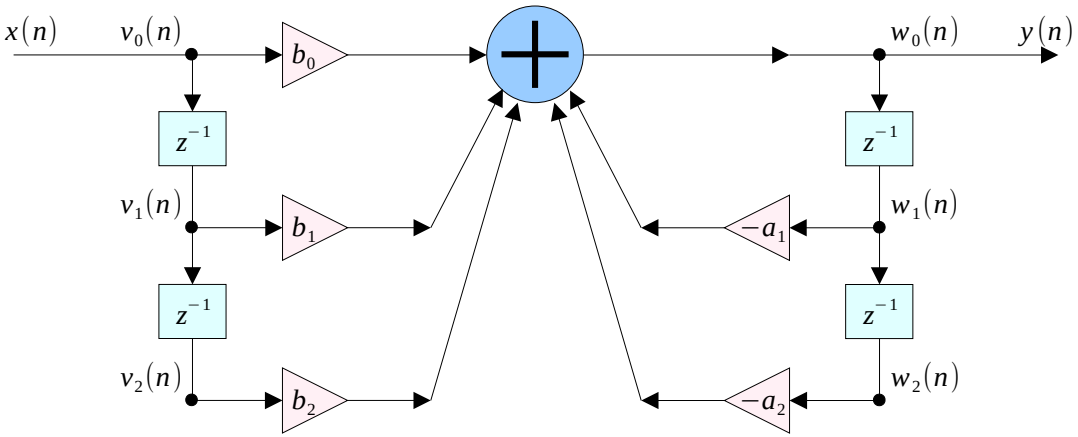
$$w_1(n+1) = w_0(n)$$

$$w_2(n+1) = w_1(n)$$

$$y_n = -a_1 y_{n-1} - a_2 y_{n-2} + b_0 x_n + b_1 x_{n-1} + b_2 x_{n-2}$$

$$y_n = -a_1 y_{n-1} - a_2 y_{n-2} + b_0 v_0(n) + b_1 v_1(n) + b_2 v_2(n)$$

Direct Form



$$v_0(n) = x(n)$$

$$w_0(n) = b_0 v_0(n) + b_1 v_1(n) + b_2 v_2(n) - a_1 w_1(n) - a_2 w_2(n)$$

$$y(n) = w_0(n)$$

$$v_2(n+1) = v_1(n) \qquad w_2(n+1) = w_1(n)$$

$$v_1(n+1) = v_0(n) \qquad w_1(n+1) = w_0(n)$$

for each input sample x do:

$$v_0 = x$$

$$w_0 = b_0 v_0 + b_1 v_1 + b_2 v_2 - a_1 w_1 - a_2 w_2$$

$$y = w_0$$

$$v_2 = v_1 \qquad w_2 = w_1$$

$$v_1 = v_0 \qquad w_1 = w_0$$

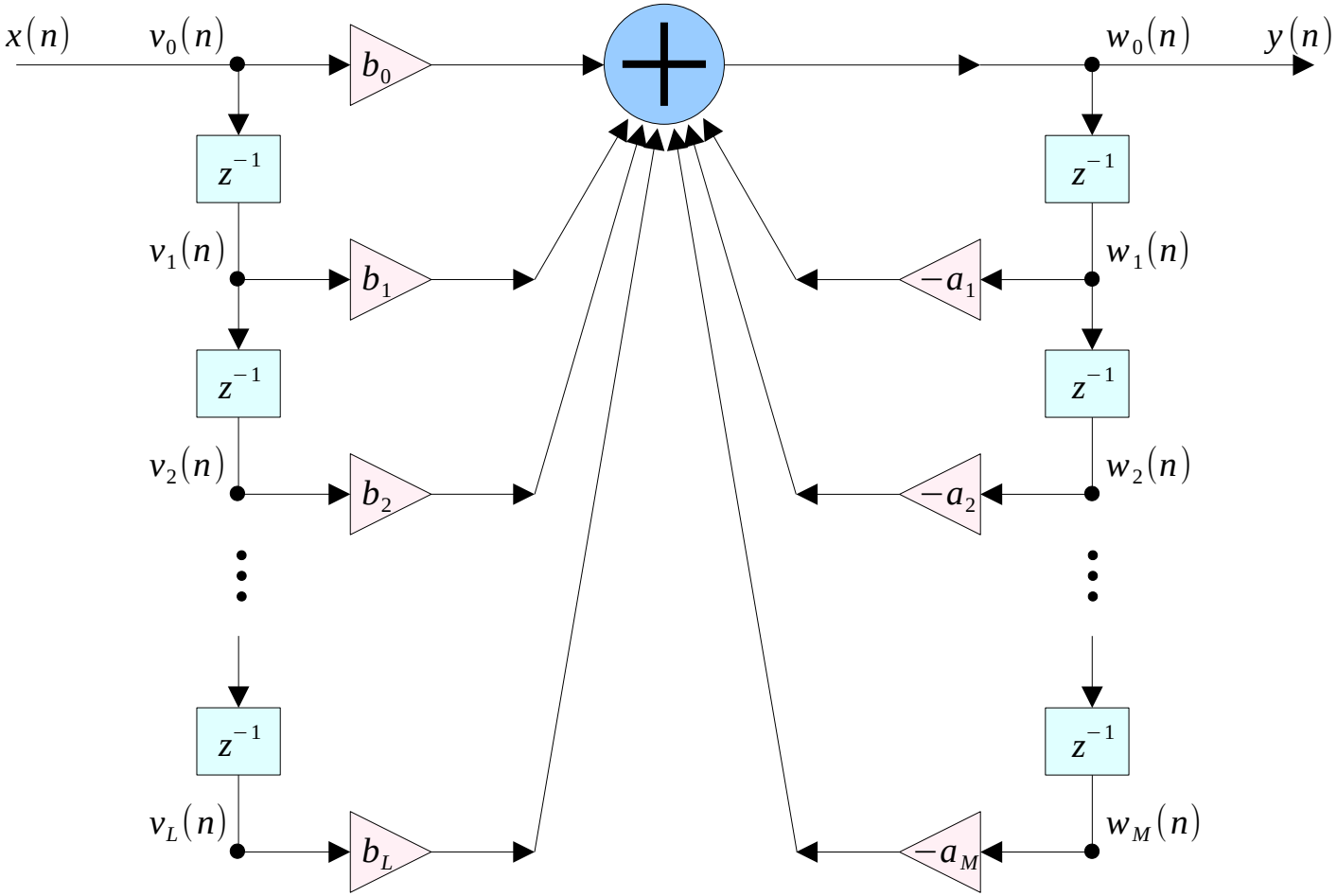
Direct Form

$$H(z) = \frac{N(Z)}{D(z)} = \frac{b_0 + b_1 z^{-1} + b_2^{-2} + \cdots + b_L z^{-L}}{1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_M z^{-M}}$$

$$y_n = -a_1 y_{n-1} - a_2 y_{n-2} - \cdots - a_M y_{n-M} + b_0 x_n + b_1 x_{n-1} + b_2 x_{n-2} + \cdots + b_L x_{n-L}$$

$$y_n = b_0 x_n + b_1 x_{n-1} + b_2 x_{n-2} + \cdots + b_L x_{n-L}$$
$$- a_1 y_{n-1} - a_2 y_{n-2} - \cdots - a_M y_{n-M}$$

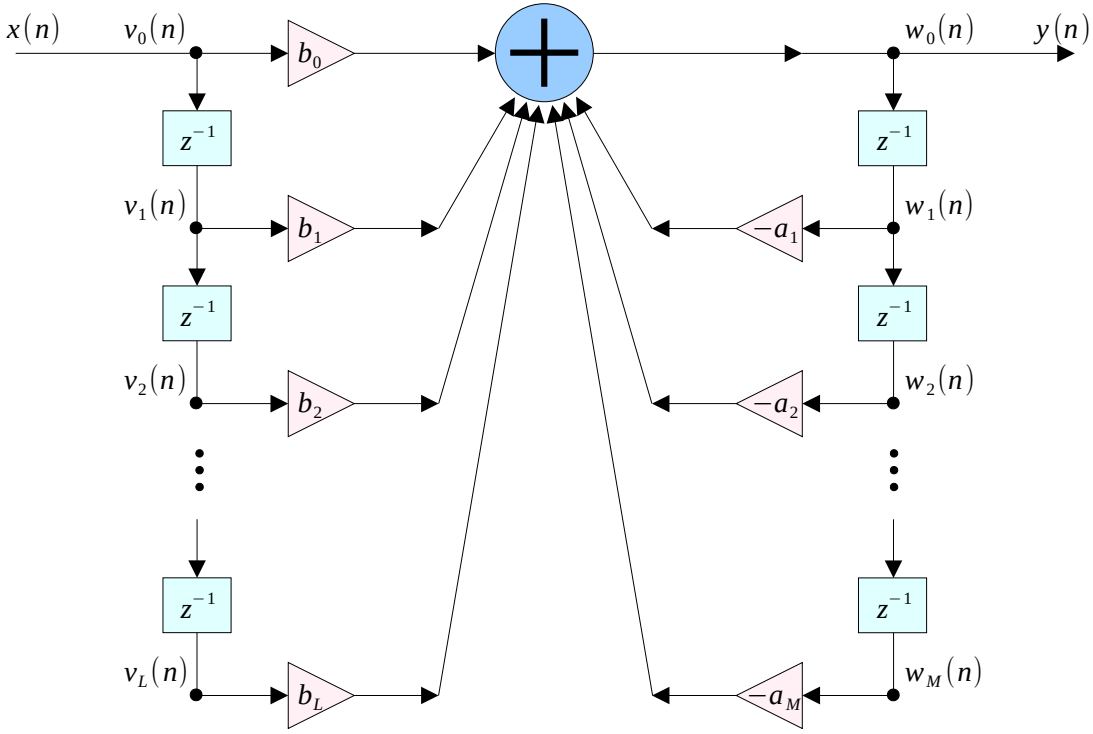
Direct Form



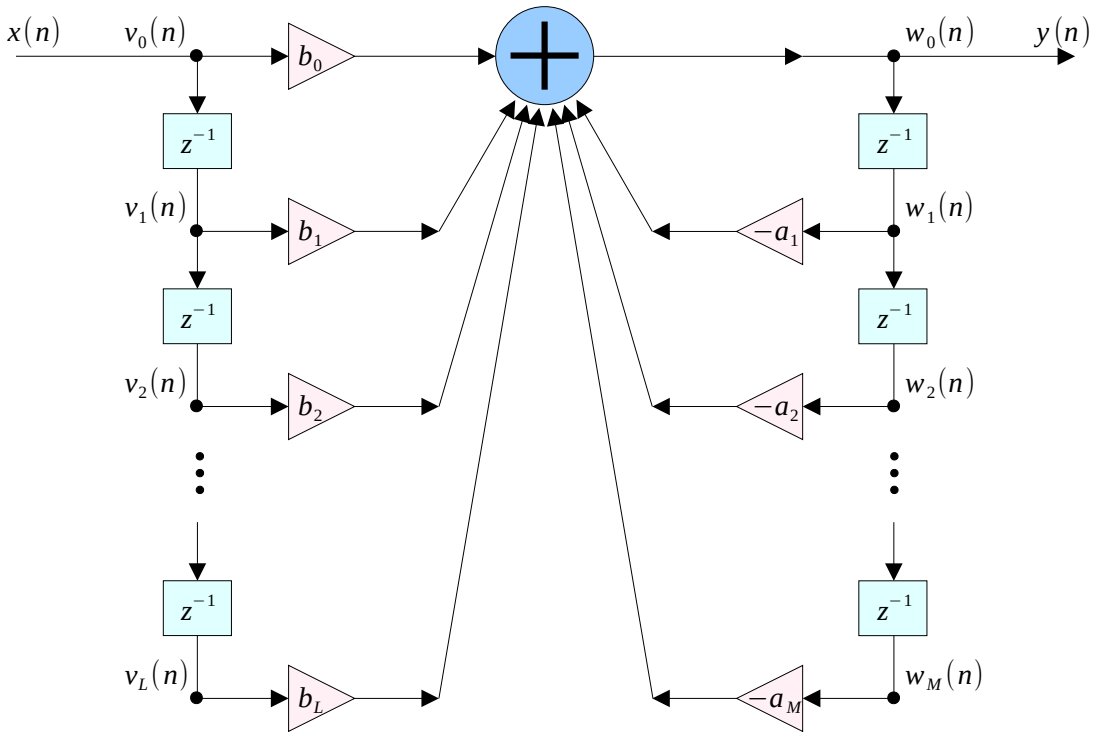
Direct Form

$$H(z) = \frac{N(Z)}{D(z)} = \frac{b_0 + b_1 z^{-1} + b_2^{-2} + \cdots + b_L z^{-L}}{1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_M z^{-M}}$$

$$y_n = b_0 x_n + b_1 x_{n-1} + b_2 x_{n-2} + \cdots + b_L x_{n-L} - a_1 y_{n-1} - a_2 y_{n-2} - \cdots - a_M y_{n-M}$$



Direct Form

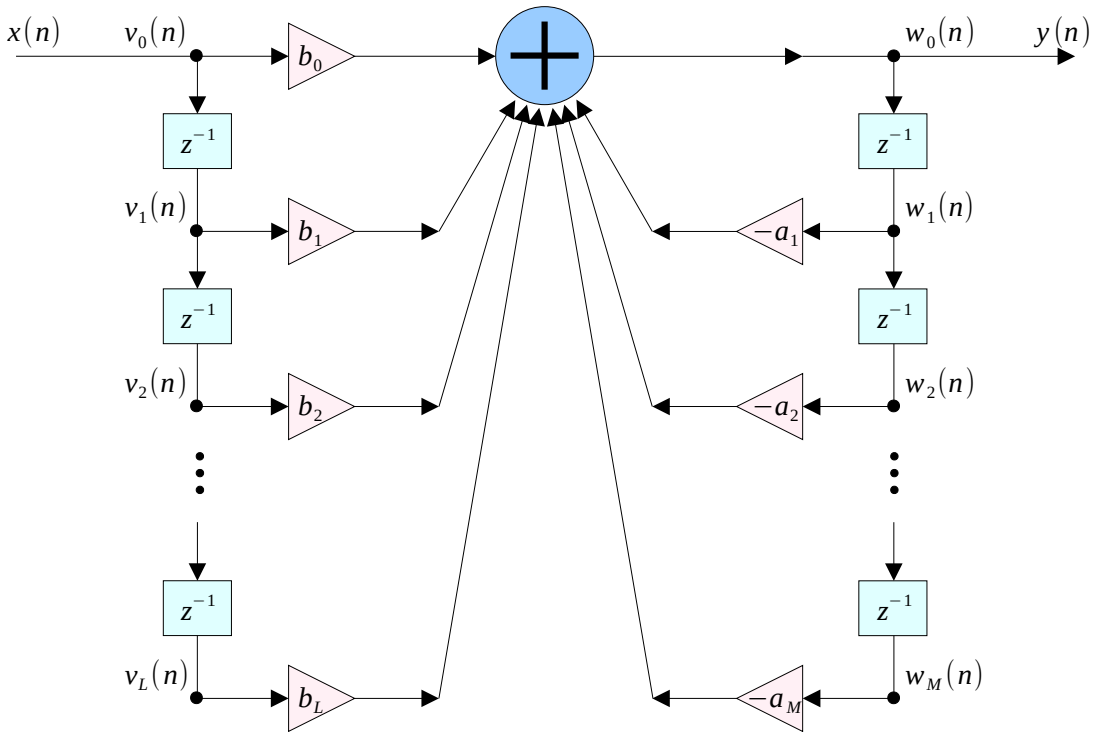


$$v_i(n) = x(n-i),\ \ i=0,1,\ldots,L$$
$$w_i(n) = y(n-i),\ \ i=0,1,\ldots,M$$

$$v_i(n+1)=v_{i-1}(n),\ \ i=0,1,\ldots,L$$
$$w_i(n+1)=w_{i-1}(n),\ \ i=1,2,\ldots,M$$

$$\begin{aligned} w_i(n+1) &= y((n+1)-i) \\ &= y(n-(i-1)) \\ &= w_{i-1}(n) \end{aligned}$$

Direct Form



$$v_i(n) = x(n-i),\ \ i=0,1,\ldots,L$$
$$w_i(n) = y(n-i),\ \ i=0,1,\ldots,M$$

$$v_i(n+1)=v_{i-1}(n),\ \ i=0,1,\ldots,L$$
$$w_i(n+1)=w_{i-1}(n),\ \ i=1,2,\ldots,M$$

$$y_n = b_0 x_n + b_1 x_{n-1} + b_2 x_{n-2} + \cdots + b_L x_{n-L} - a_1 y_{n-1} - a_2 y_{n-2} - \cdots - a_M y_{n-M}$$

$$w_0(n) = b_0 v_0(n) + b_1 v_1(n) + \cdots + b_L v_L(n) - a_1 w_1(n) - a_2 w_2(n) - \cdots - a_M w_M(n)$$

Direct Form

$$v_0(n) = x(n)$$

$$w_0(n) = b_0 v_0(n) + b_1 v_1(n) + b_2 v_2(n) + \cdots + b_L v_L(n) - a_1 w_1(n) - a_2 w_2(n) - \cdots - a_M w_M(n)$$

$$y(n) = w_0(n)$$

$$v_i(n+1) = v_{i-1}(n) \qquad i = L, L-1, \cdots 1$$

$$w_i(n+1) = w_{i-1}(n) \qquad i = M, M-1, \cdots, 1$$

for each input sample x do:

$$v_0 = x$$

$$w_0 = b_0 v_0 + b_1 v_1 + b_2 v_2 + \cdots + b_L v_L - a_1 w_1 - a_2 w_2 - \cdots - a_M w_M$$

$$y = w_0$$

$$v_i = v_{i-1} \qquad i = L, L-1, \cdots 1$$

$$w_i = w_{i-1} \qquad i = M, M-1, \cdots, 1$$

Direct Form

```
/* dir.c - IIR filtering in direct form */

double dir(  int M, double *a,              usage: y = dir(M, a, L, b, w, v, x);
             int L, double *b,              M, L denominator and numerator orders
             double *w, double *v,          v, w are internal states
             double x )                     current input sample x
{
    int i;

    v[0] = x;
    W[0] = 0;

    for (i=0; i<=L; i++)
        v[0] += b[i] * v[i];                numerator part v[0]

    for (i=1; i<=M; i++)
        w[0] -= a[i] * w[i];                denominator part w[0]

    for (i=L; i>=1; i--)
        v[i] = v[i-1];                      reverse-order updating of v

    for (i=M; i>=1; i--)
        w[i] = w[i-1];                      reverse-order updating of w

    return w[0];                            current output sample
}
```

Direct Form

```
/* dir.c - IIR filtering in direct form */

double dir(  int M, double *a,
             int L, double *b,
             double *w, double *v,
             double x )
{
    int i;

    v[0] = x;
    W[0] = 0;

    for (i=0; i<=L; i++)
        v[0] += b[i] * v[i];

    for (i=1; i<=M; i++)
        w[0] -= a[i] * w[i];

    for (i=L; i>=1; i--)
        v[i] = v[i-1];

    for (i=M; i>=1; i--)
        w[i] = w[i-1];

    return w[0];
}
```

for each input sample x do:

$$v_0 = x$$

$$w_0 = b_0 v_0 + b_1 v_1 + b_2 v_2 + \cdots + b_L v_L - a_1 w_1 - a_2 w_2 - \cdots - a_M w_M$$

$$y = w_0$$

$$v_i = v_{i-1} \qquad i = L, L-1, \cdots 1$$

$$w_i = w_{i-1} \qquad i = M, M-1, \cdots, 1$$

Direct Form

```
double *a, *b, *w, *v;

a = (double *) calloc(M+1, sizeof(double));
b = (double *) calloc(L+1, sizeof(double));

A[0] = 1;

w = (double *) calloc(M+1, sizeof(double));
v = (double *) calloc(L+1, sizeof(double));
```

(M+1)–dimensional
(L+1)–dimensional

always so

(M+1)–dimensional
(L+1)–dimensional

```
for (n = 0; n < Ntot; n++)
   y[n] = dir(M, a, L, b, w, v, x[n]);
```

for each input sample x do:

$$v_0 = x$$

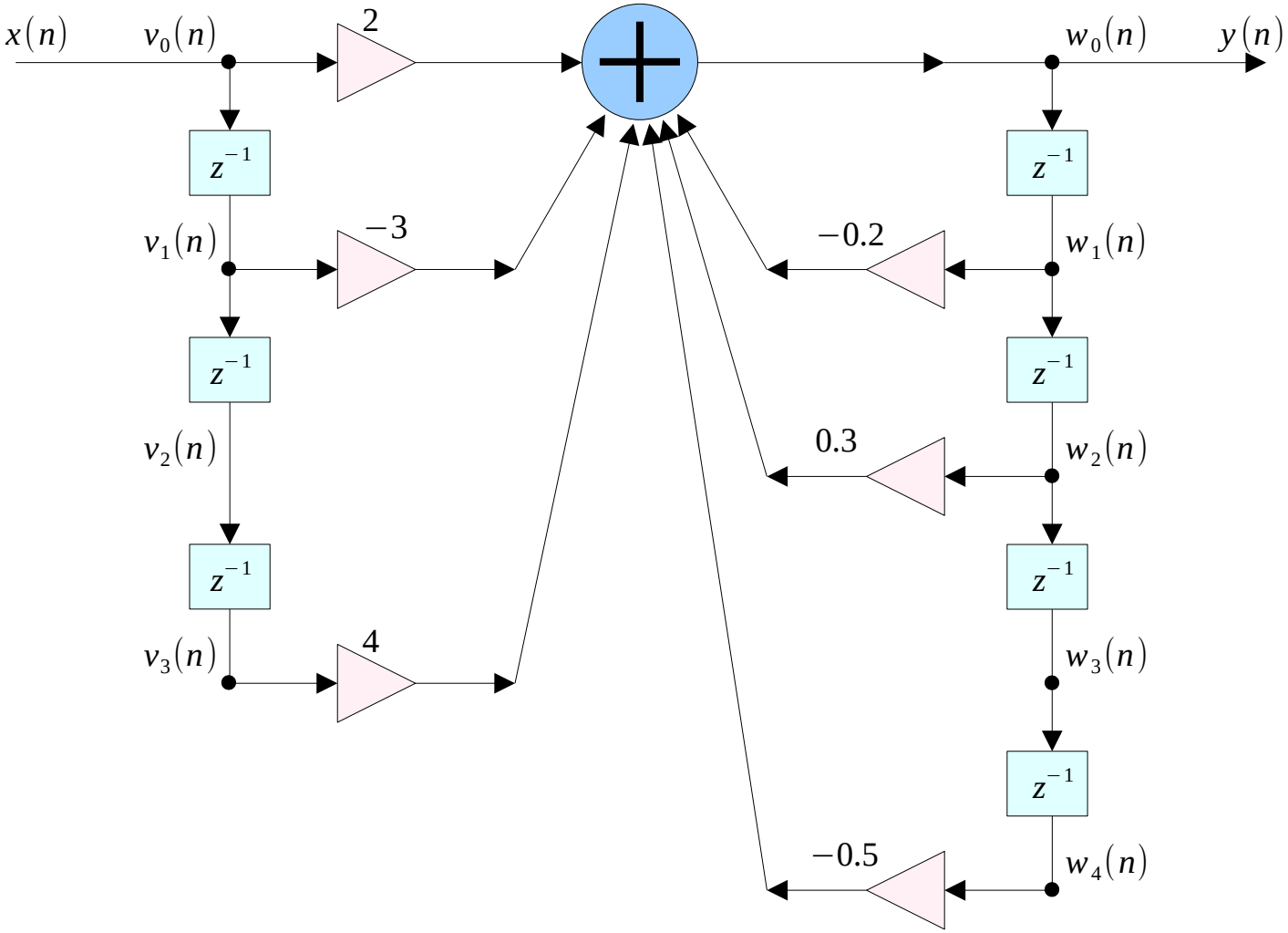
$$w_0 = b_0 v_0 + b_1 v_1 + b_2 v_2 + \cdots + b_L v_L - a_1 w_1 - a_2 w_2 - \cdots - a_M w_M$$

$$y = w_0$$

$$v_i = v_{i-1} \qquad i = L, L-1, \cdots 1$$

$$w_i = w_{i-1} \qquad i = M, M-1, \cdots, 1$$

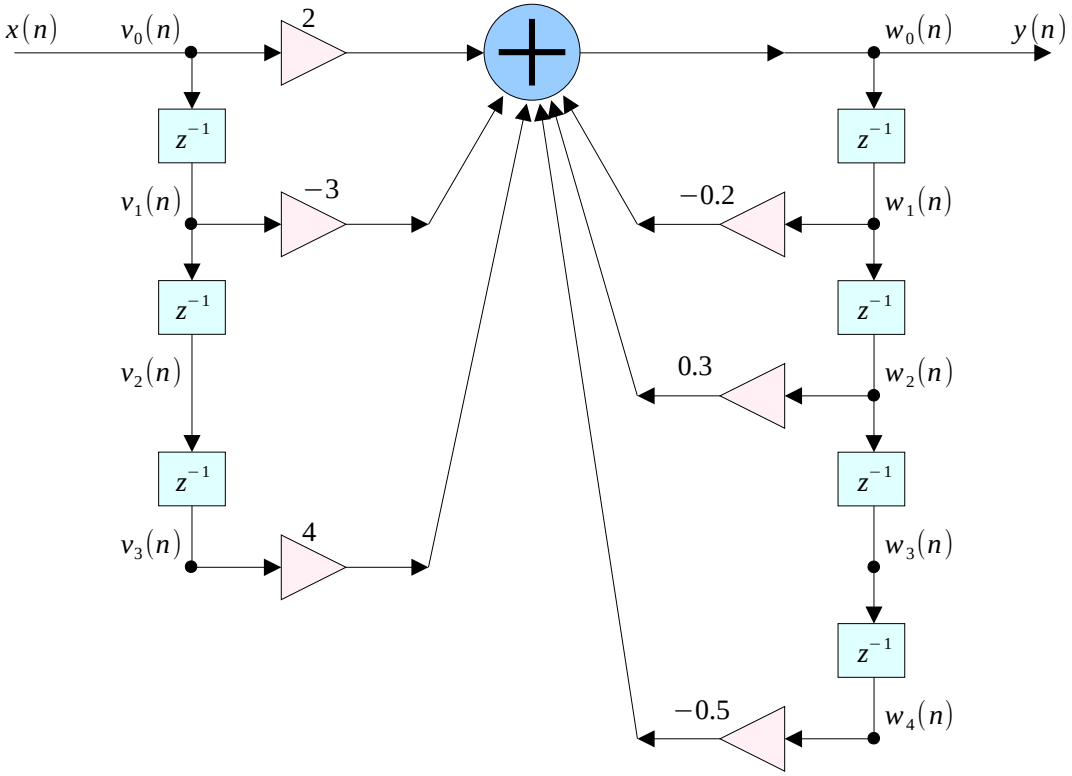
Direct Form



Direct Form

$$H(z) = \frac{N(Z)}{D(z)} = \frac{2-3z^{-1}+4z^{-3}}{1+0.2z^{-1}-0.3z^{-2}+0.5z^{-4}}$$

$$y_n = -0.2\,y_{n-1}+0.3\,y_{n-2}-0.5\,y_{n-4}+2\,x_n-3\,x_{n-1}+4\,x_{n-3}$$



for each input sample x do:

$$v_0 = x$$

$$w_0 = 2v_0 - 3v_1 + 4v_3 - 0.2\,y_{n-1} + 0.3\,y_{n-2} - 0.5\,y_{n-4}$$

$$y = w_0$$

$$w_4 = w_3$$

$$w_3 = w_2 \qquad v_3 = v_2$$

$$w_2 = w_1 \qquad v_2 = v_1$$

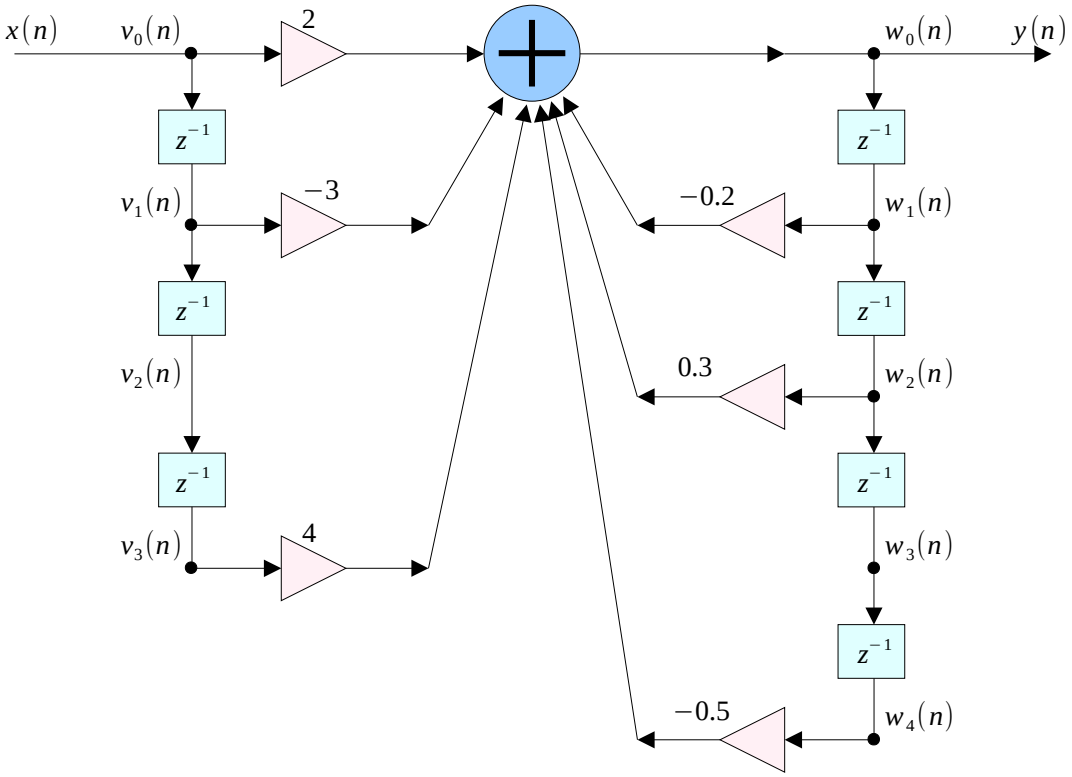
$$w_1 = w_0 \qquad v_1 = v_0$$

Direct Form

$$\boldsymbol{a} = [a_0, a_1, a_2, a_3, a_4] = [1, 0.2, -0.3, 0.0, 0.5]$$

$$\boldsymbol{b} = [b_0, b_1, b_2, b_3] = [2, -3, 0, 4]$$

$$\boldsymbol{w} = [w_0, w_1, w_2, w_3, w_4], \quad \boldsymbol{v} = [v_0, v_1, v_2, v_3]$$



for each input sample x do:

$$v_0 = x$$

$$w_0 = 2v_0 - 3v_1 + 4v_3 - 0.2y_{n-1} + 0.3y_{n-2} - 0.5y_{n-4}$$

$$y = w_0$$

$$w_4 = w_3$$

$$w_3 = w_2 \qquad v_3 = v_2$$

$$w_2 = w_1 \qquad v_2 = v_1$$

$$w_1 = w_0 \qquad v_1 = v_0$$

Direct Form

$$(L, \dot{\boldsymbol{b}}, \boldsymbol{v}) = \boldsymbol{b}^T \boldsymbol{v} = b_0 v_0 + b_1 v_1 + \cdots + b_L v_L$$
$$(M, \dot{\boldsymbol{a}}, \boldsymbol{w}) = \boldsymbol{a}^T \boldsymbol{w} = a_0 w_0 + a_1 w_1 + \cdots + a_M w_M$$

$$w_0 = 0 \quad \rightarrow \quad (M, \dot{\boldsymbol{a}}, \boldsymbol{w}) = \boldsymbol{a}^T \boldsymbol{w} = a_1 w_1 + \cdots + a_M w_M$$

for each input sample x do:

$$v_0 = x$$
$$w_0 = b_0 v_0 + b_1 v_1 + b_2 v_2 + \cdots + b_L v_L - a_1 w_1 - a_2 w_2 - \cdots - a_M w_M$$
$$y = w_0$$

$$v_i = v_{i-1} \qquad i = L, L-1, \cdots 1$$
$$w_i = w_{i-1} \qquad i = M, M-1, \cdots, 1$$

for each input sample x do:

$$v_0 = x$$
$$w_0 = 0$$
$$w_0 = (L, \dot{\boldsymbol{b}}, \boldsymbol{v}) - (M, \dot{\boldsymbol{a}}, \boldsymbol{w})$$
$$y = w_0$$
$$delay(L, \boldsymbol{v})$$
$$delay(M, \boldsymbol{w})$$

Direct Form

```
/* dir2.c - IIR filtering in direct form */

double dot();
void delay();

double dir2 (int M, double *a,
             int L, double *b,
             double *w, double *v,
             double x)
{
    v[0] = x;
    w[0] = 0;


    w[0] = dot(L, b, v) - dot(M, a, w);

    delay(L, v);

    delay(M, w);

    return w[0];
}
```

usage: y = dir2(M, a, L, b, w, v, x);

current input sample
needed for dot(M,a,w)

current output

update input delay line

update output delay line

Direct Form

References

[1] S. J. Ofranidis , Introduction to Signal Processing