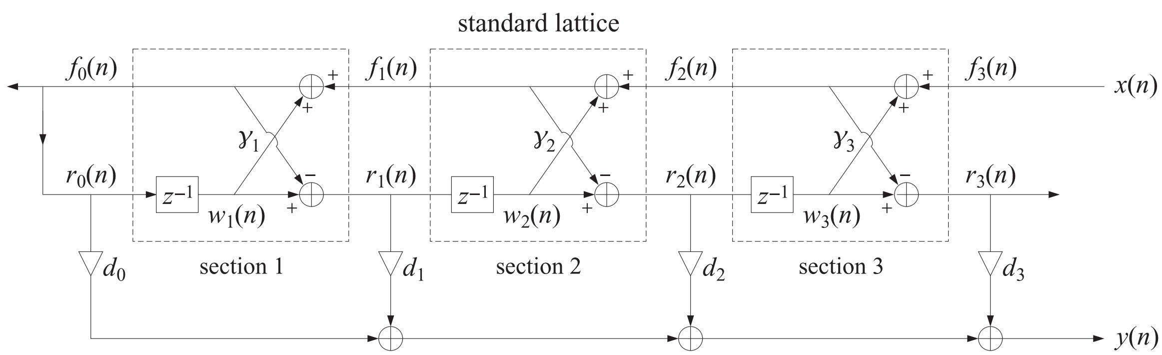
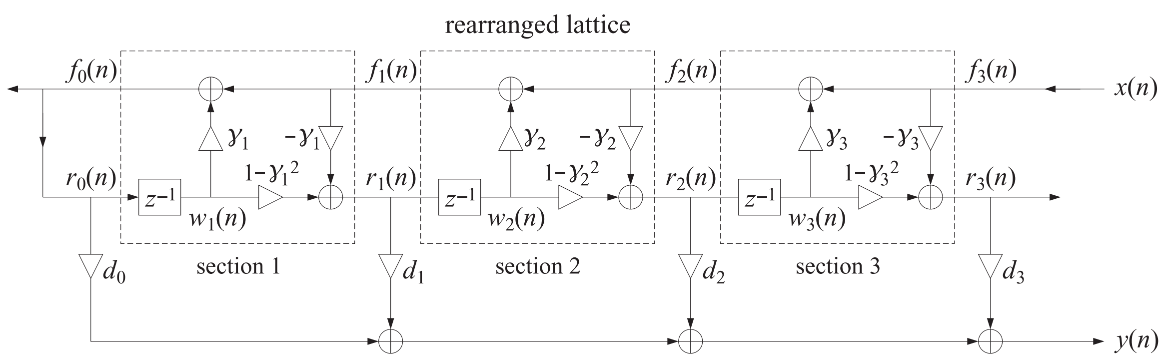
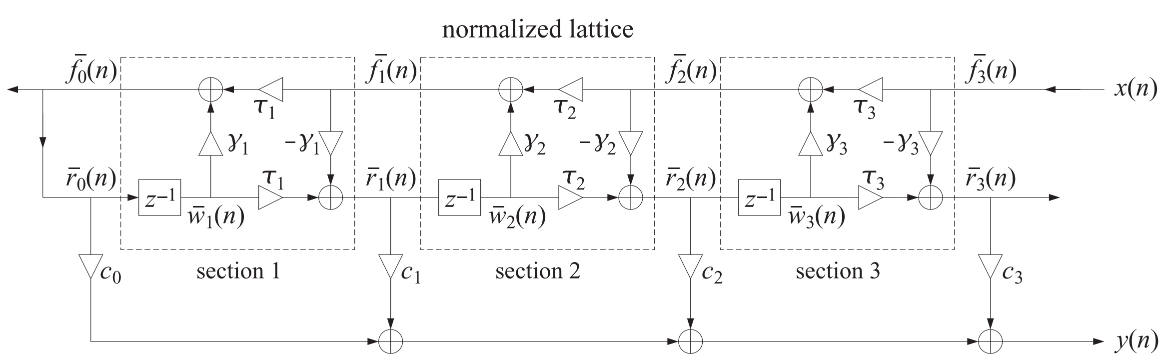
RUTGERS UNIVERSITY
School of Engineering – ECE Department
332:447 – Digital Signal Processing Design
S. J. Orfanidis – Fall 2017

Lattice Realizations – Summary

The standard lattice/ladder realization, and its rearranged un-normalized, and normalized forms are depicted below.







The signals, $f_p(n)$, $r_p(n)$, are the same in the standard and rearranged cases. The signals, $\bar{f}_p(n)$, $\bar{r}_p(n)$, of the normalized case are scaled versions of the standard ones. For the general order-M case, we have for $p = 0, 1, \dots, M$,

$$\boxed{\begin{aligned} f_p(n) &= \sigma_p \bar{f}_p(n) \\ r_p(n) &= \sigma_p \bar{r}_p(n) \\ d_p &= \frac{\sigma_M}{\sigma_p}\, c_p \end{aligned}} \qquad \text{where} \qquad \begin{aligned} &\sigma_0 = 1 \\ &\sigma_p = \tau_p \sigma_{p-1} = \tau_1 \tau_2 \cdots \tau_p\,, \quad p = 1, 2, \dots, M \\ &\tau_p = (1 - \gamma_p^2)^{1/2}\,, \quad p = 1, 2, \dots, M \end{aligned} \tag{1}$$

The difference equations and sample processing algorithm for the standard lattice are,

$$
\boxed{\begin{array}{l}
\text{for each time instant } n\text{, do,}\\
\quad f_M(n) = x(n)\\
\quad f_{M-1}(n) = f_M(n) + \gamma_M w_M(n)\\
\quad r_M(n) = w_M(n) - \gamma_M f_{M-1}(n)\\
\quad y(n) = d_M r_M(n)\\
\quad \text{for } p = M-1, M-2, \dots, 1\text{, do,}\\
\quad\quad f_{p-1}(n) = f_p(n) + \gamma_p w_p(n)\\
\quad\quad r_p(n) = w_p(n) - \gamma_p f_{p-1}(n)\\
\quad\quad w_{p+1}(n+1) = r_p(n)\\
\quad\quad y(n) = y(n) + d_p r_p(n)\\
\quad \text{end}\\
\quad r_0(n) = f_0(n)\\
\quad w_1(n+1) = r_0(n)\\
\quad y(n) = y(n) + d_0 r_0(n)\\
\text{end}
\end{array}}
\quad\Rightarrow\quad
\boxed{\begin{array}{l}
\text{for each input sample } x\text{, do,}\\
\quad f = x\\
\quad f = f + \gamma_M w_M\\
\quad r = w_M - \gamma_M f\\
\quad y = d_M r\\
\quad \text{for } p = M-1, M-2, \dots, 1\text{, do,}\\
\quad\quad f = f + \gamma_p w_p\\
\quad\quad r = w_p - \gamma_p f\\
\quad\quad w_{p+1} = r\\
\quad\quad y = y + d_p r\\
\quad \text{end}\\
\quad r = f\\
\quad w_1 = r\\
\quad y = y + d_0 r\\
\text{end}
\end{array}}
\tag{2}
$$

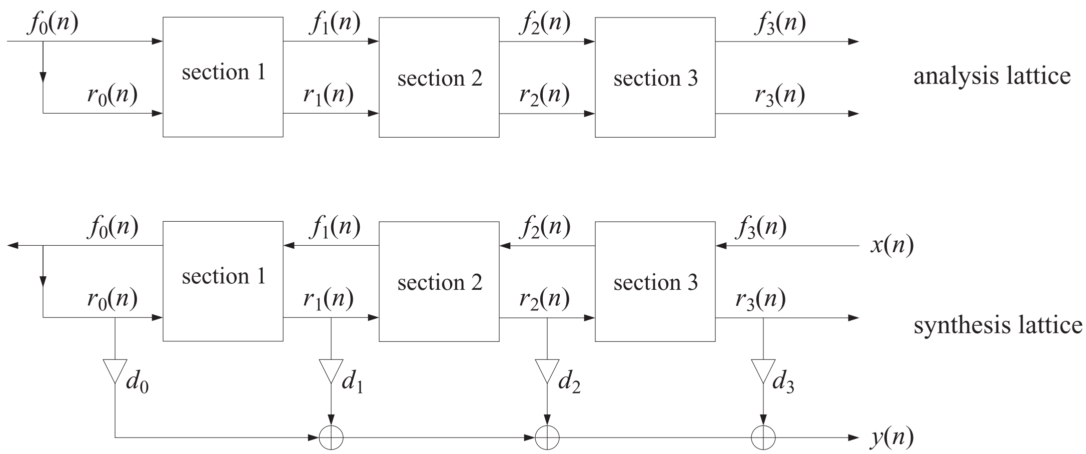
and for the rearranged case,

$$
\boxed{\begin{array}{l}
\text{for each time instant } n\text{, do,}\\
\quad f_M(n) = x(n)\\
\quad r_M(n) = (1-\gamma_M^2) w_M(n) - \gamma_M f_M(n)\\
\quad f_{M-1}(n) = f_M(n) + \gamma_M w_M(n)\\
\quad y(n) = d_M r_M(n)\\
\quad \text{for } p = M-1, M-2, \dots, 1\text{, do,}\\
\quad\quad r_p(n) = (1-\gamma_p^2) w_p(n) - \gamma_p f_p(n)\\
\quad\quad f_{p-1}(n) = f_p(n) + \gamma_p w_p(n)\\
\quad\quad w_{p+1}(n+1) = r_p(n)\\
\quad\quad y(n) = y(n) + d_p r_p(n)\\
\quad \text{end}\\
\quad r_0(n) = f_0(n)\\
\quad w_1(n+1) = r_0(n)\\
\quad y(n) = y(n) + d_0 r_0(n)\\
\text{end}
\end{array}}
\quad\Rightarrow\quad
\boxed{\begin{array}{l}
\text{for each input sample } x\text{, do,}\\
\quad f = x\\
\quad r = (1-\gamma_M^2) w_M - \gamma_M f\\
\quad f = f + \gamma_M w_M\\
\quad y = d_M r\\
\quad \text{for } p = M-1, M-2, \dots, 1\text{, do,}\\
\quad\quad r = (1-\gamma_p^2) w_p - \gamma_p f\\
\quad\quad f = f + \gamma_p w_p\\
\quad\quad w_{p+1} = r\\
\quad\quad y = y + d_p r\\
\quad \text{end}\\
\quad r = f\\
\quad w_1 = r\\
\quad y = y + d_0 r\\
\text{end}
\end{array}}
\tag{3}
$$

In the sample processing algorithms, the variables f, r are recycled from one lattice section to the next, and in the rearranged case, r must be updated before f to avoid overwriting the correct value of f. The sample processing algorithms can easily be converted into a MATLAB function.

A similar set of difference equations and sample processing algorithm can easily be written for the normalized form.

The theory behind these realizations is explained fully in class. Here, we summarize the essential results. The analysis and synthesis versions of the lattice structures are depicted below.



In the analysis structure, the transfer functions from the overall input $F_0(z)$ to the outputs, $F_p(z), R_p(z)$ of the pth lattice section are given in terms of the order-p forward and reversed linear prediction polynomials, $A_p(z), A_p^R(z)$, satisfying the forward Levinson recursions, starting with $A_0(z) = A_0^R(z) = 1$,

$$\begin{aligned} F_p(z) &= A_p(z)F_0(z) \\ R_p(z) &= A_p^R(z)F_0(z) \end{aligned} \tag{4}$$

for $p = 1, 2, \dots, M$, where,

$$\begin{bmatrix} A_p(z) \\ A_p^R(z) \end{bmatrix} = \begin{bmatrix} 1 & -\gamma_p z^{-1} \\ -\gamma_p & z^{-1} \end{bmatrix} \begin{bmatrix} A_{p-1}(z) \\ A_{p-1}^R(z) \end{bmatrix} \qquad \text{(forward Levinson)} \tag{5}$$

In the time-domain, this reads as follows, building an order-p filter from and order-$(p-1)$ one,

$$\mathbf{a}_p = \begin{bmatrix} \mathbf{a}_{p-1} \\ 0 \end{bmatrix} - \gamma_p \begin{bmatrix} 0 \\ \mathbf{a}_{p-1}^R \end{bmatrix}, \quad p = 1, 2, \dots, M \qquad \text{(forward Levinson)} \tag{6}$$

where $\mathbf{a}_p = [1, a_{p1}, a_{p2}, \dots, a_{pp}]^T$ is the column vector of the coefficients of $A_p(z)$, starting with $\mathbf{a}_0 = [1]$. Explicitly, we have,

$$\begin{bmatrix} 1 \\ a_{p,1} \\ a_{p,2} \\ \vdots \\ a_{p,p-1} \\ a_{p,p} \end{bmatrix} = \begin{bmatrix} 1 \\ a_{p-1,1} \\ a_{p-1,2} \\ \vdots \\ a_{p-1,p-1} \\ 0 \end{bmatrix} - \gamma_p \begin{bmatrix} 0 \\ a_{p-1,p-1} \\ a_{p-1,p-2} \\ \vdots \\ a_{p-1,1} \\ 1 \end{bmatrix}$$

The inverse of this recursion is the backward Levinson recursion which starts with the final order-M filter $\mathbf{a}_M = [1, a_{M1}, a_{M2}, \dots, a_{MM}]^T$ and proceeds downwards to order-0, extracting the reflection coefficients γ_p in the process. We have for, $p = M, M-1, \dots, 1$,

$$\gamma_p = -a_{p,p}, \quad \mathbf{a}_{p-1} = \frac{\mathbf{a}_p + \gamma_p \mathbf{a}_p^R}{1 - \gamma_p^2} \qquad \text{(backward Levinson)} \tag{7}$$

or, explicitly,

$$\begin{bmatrix} 1 \\ a_{p-1,1} \\ a_{p-1,2} \\ \vdots \\ a_{p-1,p-1} \\ 0 \end{bmatrix} = \frac{1}{1-\gamma_p^2} \left(\begin{bmatrix} 1 \\ a_{p,1} \\ a_{p,2} \\ \vdots \\ a_{p,p-1} \\ a_{p,p} \end{bmatrix} + \gamma_p \begin{bmatrix} a_{p,p} \\ a_{p,p-1} \\ \vdots \\ a_{p,2} \\ a_{p,1} \\ 1 \end{bmatrix} \right)$$

In the synthesis structure, starting with the right-most input $X(z)=F_M(z)=A_M(z)F_0(z)$, the transfer function from $X(z)$ to the backward output $R_p(z)$ of the pth lattice section is,

$$R_p(z)=A_p^R(z)F_0(z)=\frac{A_p^R(z)}{A_M(z)}A_M(z)F_0(z)=\frac{A_p^R(z)}{A_M(z)}F_M(z)=\frac{A_p^R(z)}{A_M(z)}X(z) \tag{8}$$

for $p=0,1,\dots,M$. Thus, the output $Y(z)$, being the sum of $R_p(z)$ with the ladder coefficients, is

$$Y(z)=\sum_{p=0}^{M}d_pR_p(z)=\frac{\sum_{p=0}^{M}d_pA_p^R(z)}{A_M(z)}X(z) \tag{9}$$

and the overall transfer function from $X(z)$ to $Y(z)$,

$$H(z)=\frac{Y(z)}{X(z)}=\frac{\sum_{p=0}^{M}d_pA_p^R(z)}{A_M(z)}=\frac{B_M(z)}{A_M(z)} \tag{10}$$

where the numerator polynomial $B_M(z)$ is expressible as a linear combination of the reversed $A_p^R(z)$,

$$B_M(z)=\sum_{p=0}^{M}d_pA_p^R(z) \tag{11}$$

This can be written in the time domain in the matrix form,

$$\mathbf{b}_M=\sum_{p=0}^{M}d_p\mathbf{a}_p^R=[\mathbf{a}_0^R,\mathbf{a}_1^R,\cdots,\mathbf{a}_M^R]\begin{bmatrix}d_0\\d_1\\\vdots\\d_M\end{bmatrix}=A\mathbf{d} \tag{12}$$

where A is a unit upper-tridiagonal matrix consisting of the reversed prediction polynomials. For example, for $M=3$, we have,

$$A=[\mathbf{a}_0^R,\mathbf{a}_1^R,\mathbf{a}_2^R,\mathbf{a}_3^R]=\begin{bmatrix}1&a_{11}&a_{22}&a_{33}\\0&1&a_{21}&a_{32}\\0&0&1&a_{31}\\0&0&0&1\end{bmatrix}$$

and Eq. (12) becomes,

$$\mathbf{b}=\begin{bmatrix}b_0\\b_1\\b_2\\b_3\end{bmatrix}=\begin{bmatrix}1&a_{11}&a_{22}&a_{33}\\0&1&a_{21}&a_{32}\\0&0&1&a_{31}\\0&0&0&1\end{bmatrix}\begin{bmatrix}d_0\\d_1\\d_2\\d_3\end{bmatrix}=A\mathbf{d} \tag{13}$$

Solving for the ladder weights in terms of **b** is efficiently done with MATLAB's backslash operator,

$$\mathbf{d}=A^{-1}\mathbf{b}=A\backslash\mathbf{b} \tag{14}$$

Finally, we note that in the normalized form, because of the scale factors of Eq. (1), the overall output and resulting transfer function are as follows, with input $X(z)=\bar{F}_M(z)=\sigma_M^{-1}F_M(z)=\sigma_M^{-1}A_M(z)F_0(z)$,

$$Y(z)=\sum_{p=0}^{M}c_p\bar{R}_p(z)=\sum_{p=0}^{M}c_p\sigma_p^{-1}R_p(z)=\sum_{p=0}^{M}c_p\sigma_p^{-1}A_p^R(z)F_0(z)$$

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{p=0}^{M} c_p \sigma_p^{-1} A_p^R(z) F_0(z)}{\sigma_M^{-1} A_M(z) F_0(z)}, \quad \text{or,}$$

$$H(z) = \frac{\sum_{p=0}^{M} c_p \sigma_p^{-1} \sigma_M A_p^R(z)}{A_M(z)} = \frac{B_M(z)}{A_M(z)} \tag{15}$$

which becomes identical to the transfer function of the standard lattice provided that the d_p and c_p coefficients are related as in Eq. (1),

$$d_p = c_p \sigma_p^{-1} \sigma_M, \quad p = 0, 1, \ldots, M \tag{16}$$

Going back and forth from the direct-form coefficients to the lattice coefficients of the normalized or standard forms can be done with the following functions, which use the recursions (6) and (7), also successively building the matrix A that is needed to generate the ladder coefficients,

```
[g,d] = dir2lat(b,a,type);   % type = 'n', 's', for normalized or standard forms
[b,a] = lat2dir(g,d,type);   % default type = 'n'
```

The essential code in these functions is given below:

```
lat2dir.m     gamma,d --> b,a
---------------------------------------
M = length(g);

d = d(:); g = g(:);

a = 1;
A = eye(M+1);

for p = 1:M,
   a = [a; 0] - g(p)*[0; flip(a)];
   A(1:p+1,p+1) = flip(a);
end

if type == 'n'
   t = sqrt(1-g.^2);
   s = [1; cumprod(t)] / prod(t);
   d = d./s;
end

b = A*d;
```

```
dir2lat.m     b,a --> gamma,d
---------------------------------------
g = zeros(M,1);

A(1:M+1,M+1) = flip(a);

for p = M:-1:1
   g(p) = -a(end);
   a = (a + g(p)*flip(a))/(1-g(p)^2);
   a(end) = [];
   A(1:p,p) = flip(a);
end

d = A\b;

if type == 'n'
   t = sqrt(1-g.^2);
   s = [1; cumprod(t)] / prod(t);
   d = s.*d;
end
```

The built-in MATLAB functions, **tf2latc** and **latc2tf**, perform similar functions, but only for the standard lattice. By their convention, their reflection coefficients k_p are the negatives of ours, that is, $k_p = -\gamma_p$. The ladder coefficients d_p are the same.

The **lat2dir** function can be modified in order to compute the frequency response of the lattice. The essential code is given below.

```
lat2freq.m     gamma,d --> frequency response
---------------------------------------------
M = length(g);       % g,d assumed columns

if type == 'n'
   t = sqrt(1-g.^2);
   s = [1; cumprod(t)] / prod(t);
   d = d./s;
end

a = 1;
B = d(1);            % numerator DTFT

for p = 1:M,
   a = [a; 0] - g(p)*[0; flip(a)];
   B = B + d(p+1) * freqz(flip(a),1,w);
end

A = freqz(a,1,w);    % denominator DTFT

H = B./A;            % frequency response H(w)
```

Example

Consider the following order-3 transfer function,

$$H(z) = \frac{B_3(z)}{A_3(z)} = \frac{7 + 1.14z^{-1} - 3.9z^{-2} + z^{-3}}{1 + 0.1z^{-1} - 0.26z^{-2} + 0.6z^{-3}}$$

with filter coefficient vectors,

$$\mathbf{a}_3 = \begin{bmatrix} 1 \\ 0.1 \\ -0.26 \\ 0.6 \end{bmatrix}, \quad \mathbf{b}_3 = \begin{bmatrix} 7 \\ 1.14 \\ -3.9 \\ 1 \end{bmatrix}$$

We will construct the lower order polynomials $A_2(z), A_1(z), A_0(z)$ from the backward Levinson recursion, and extract the reflection coefficients $\gamma_1, \gamma_2, \gamma_3$ in the process, and also express $B_3(z)$ as a linear combination of the reversed polynomials in the form,

$$B_3(z) = d_0 A_0^R(z) + d_1 A_1^R(z) + d_2 A_2^R(z) + d_3 A_3^R(z) \tag{17}$$

The steps are as follows. First we extract γ_3 and calculate $\mathbf{a}_2$,

$$\gamma_3 = -a_{33} = -0.6 \quad \Rightarrow \quad \mathbf{a}_2 = \frac{\mathbf{a}_3 + \gamma_3 \mathbf{a}_3^R}{1 - \gamma_3^2} = \frac{\begin{bmatrix} 1 \\ 0.1 \\ -0.26 \\ 0.6 \end{bmatrix} + (-0.6)\begin{bmatrix} 0.6 \\ -0.26 \\ 0.1 \\ 1 \end{bmatrix}}{1 - (-0.6)^2} = \begin{bmatrix} 1 \\ 0.4 \\ -0.5 \\ 0 \end{bmatrix}$$

and after deleting the bottom zero in $\mathbf{a}_2$, we repeat the recursion to get γ_2 and $\mathbf{a}_1$,

$$\gamma_2 = -a_{22} = 0.5 \quad \Rightarrow \quad \mathbf{a}_1 = \frac{\mathbf{a}_2 + \gamma_2 \mathbf{a}_2^R}{1 - \gamma_2^2} = \frac{\begin{bmatrix} 1 \\ 0.4 \\ -0.5 \end{bmatrix} + 0.5\begin{bmatrix} -0.5 \\ 0.4 \\ 1 \end{bmatrix}}{1 - 0.5^2} = \begin{bmatrix} 1 \\ 0.8 \\ 0 \end{bmatrix}$$

and after deleting the bottom zero of $\mathbf{a}_1$, we have,

$$\mathbf{a}_1 = \begin{bmatrix} 1 \\ 0.8 \end{bmatrix} \quad \Rightarrow \quad \gamma_1 = -a_{11} = -0.8$$

thus, $[\gamma_1, \gamma_2, \gamma_3] = [-0.8,\ 0.5,\ -0.6]$. The feed-forward numerator coefficients d_p are calculated by solving the equation (13), which reads here,

$$\begin{bmatrix} 1 & 0.8 & -0.5 & 0.6 \\ 0 & 1 & 0.4 & -0.26 \\ 0 & 0 & 1 & 0.1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} 7 \\ 1.14 \\ -3.9 \\ 1 \end{bmatrix} \quad \Rightarrow \quad \begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} 2 \\ 3 \\ -4 \\ 1 \end{bmatrix}$$

with the solution obtained easily with the backward substitution. Finally, we verify Eq. (17)

$$\begin{aligned} B_3(z) &= d_0 A_0^R(z) + d_1 A_1^R(z) + d_2 A_2^R(z) + d_3 A_3^R(z) \\ &= 2 + 3\left(0.8 + z^{-1}\right) + (-4)\left(-0.5 + 0.4z^{-1} + z^{-2}\right) + \left(0.6 - 0.26z^{-1} + 0.1z^{-2} + z^{-3}\right) \\ &= 7 + 1.14z^{-1} - 3.9z^{-2} + z^{-3} \end{aligned}$$

Finally, we note that the normalized lattice feed-forward coefficients c_p are related to the d_p coefficients as follows in the order-3 case,

$$
\begin{aligned}
\sigma_0 &= 1 \\
\sigma_1 &= \tau_1 \\
\sigma_2 &= \tau_1\tau_2 \\
\sigma_3 &= \tau_1\tau_2\tau_3
\end{aligned}
\quad \Rightarrow \quad d_p = \frac{\sigma_3}{\sigma_p}\, c_p \quad \Rightarrow \quad
\begin{aligned}
d_0 &= \tau_1\tau_2\tau_3\, c_0 \\
d_1 &= \tau_2\tau_3\, c_1 \\
d_2 &= \tau_3\, c_2 \\
d_3 &= c_3
\end{aligned}
$$

where

$$
\begin{aligned}
\tau_1 &= \sqrt{1-\gamma_1^2} = \sqrt{1-(-0.8)^2} = 0.6 \\
\tau_2 &= \sqrt{1-\gamma_2^2} = \sqrt{1-(0.5)^2} = 0.8660 \\
\tau_3 &= \sqrt{1-\gamma_3^2} = \sqrt{1-(-0.6)^2} = 0.8
\end{aligned}
$$

References

[1] A. H. Gray and J. D. Markel, “A Normalized Digital Filter Structure,” *IEEE Trans. Acoust., Speech, Signal Process.*, **ASSP-23**, 268 (1975).

[2] A. H. Gray and J. D. Markel, “Roundoff Noise Characteristics of a Class of Orthogonal Polynomial Structures,” *IEEE Trans. Acoust., Speech, Signal Process.*, **ASSP-23**, 473 (1975).