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

Project 3 – Normalized Lattice Realizations and Quantization Effects in IIR Filters

Assigned 09/28/17 – Due 10/05/17

1. Introduction

The purpose of this project is twofold: first, to construct MATLAB functions for the design and filtering of the normalized lattice realizations, and second, to study the impact of coefficient quantization on the design specifications and stability of IIR filters, and to evaluate the relative robustness of different realizations, such as direct/canonical/transposed, cascade, and normalized lattice.

2. Design Procedures

1. The normalized lattice realization is one of the best, if not the best, realization with regard to robustness and stability under coefficient quantization, roundoff noise accumulation, and protection against internal overflows. These nice properties can be traced to the fact that all internal branches are scaled by multipliers that have magnitude less than unity. Its block diagram is summarized in the Appendix and compared to the standard lattice realization.

   a. Write a MATLAB function, **nlfilt**, that implements IIR filtering in the normalized lattice form, formulated on a sample-by-sample basis with the help of the internal states w_p as shown on the block diagram—no time-arrays should be used internally, except for $x(n)$ and $y(n)$. It should also be able to handle the standard lattice realization, and should have syntax,

```
y = nlfilt(gamma, d, x, type);

% gamma = M-dimensional vector of reflection coefficients
%     d = (M+1)-dimensional vector of ladder coefficients
%     x = input signal vector
%     y = output signal vector, same size as x
%
% type = 'n', 's', for normalized or standard realization
```

   Note that MATLAB has a built-in function, **latcfilt**, for the standard lattice, but not for the normalized one.

   b. In addition to **nlfilt**, also write two MATLAB functions, **dir2lat** and **lat2dir**, that allow you to pass back and forth from the direct-form coefficients to the lattice coefficients of the normalized or standard forms,

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

   The essential code for such functions is given in the Appendix. For the standard lattice, these are equivalent to the built-in functions, **tf2latc** and **latc2tf**, except for the opposite sign convention used for the reflection coefficients.

   Finally, write a MATLAB function, **lat2freq**, that calculates the frequency response $H(\omega)$ of the normalized or standard lattice forms at any vector of digital frequencies ω in rads/sample. It must be structured to evaluate either Eq. (12) or (16) of the Appendix at the unit-circle points $z = e^{j\omega}$, and should have syntax,

```
H = lat2freq(g,d,w,type);   % type = 'n' or 's'
```

The essential code for such a function is included in the Appendix

c. To test your design, consider the following order-3 filter,

$$H(z) = \frac{2.484 + 3.57264\,z^{-1} - 3.1896\,z^{-2} + 0.9\,z^{-3}}{1 - 0.544\,z^{-1} - 0.3344\,z^{-2} + 0.8z^{-3}}$$

Calculate the realization coefficients of the standard and normalized lattice forms, γ_p, τ_p, d_p, c_p. Note, that the computed values are exact with 4-digit precision. Then, filter the signal, $\mathbf{x} = [1,2,3,4,5,6,7,8,9]$, through your function, **nlfilt**, for both normalized and standard types, as well as through, **filter**, and compare the results. To check your answer, the correct output is,

```
%    x       filter    nlfilt
% --------------------------
%  1.0000    2.4840    2.4840
%  2.0000    9.8919    9.8919
%  3.0000   17.6195   17.6195
%  4.0000   26.0804   26.0804
%  5.0000   31.1079   31.1079
%  6.0000   34.2572   34.2572
%  7.0000   34.6499   34.6499
%  8.0000   35.6617   35.6617
%  9.0000   37.5911   37.5911
```

2. Consider the double resonator filter given in cascade and direct forms:

$$\begin{aligned} H(z) &= \frac{1}{1 - 1.8955z^{-1} + 0.9930z^{-2}} \cdot \frac{1}{1 - 1.6065z^{-1} + 0.9859z^{-2}} \\ &= \frac{1}{1 - 3.5020z^{-1} + 5.0240z^{-2} - 3.4640z^{-3} + 0.9790z^{-4}} \end{aligned} \tag{1}$$

using four-digit precision to represent the coefficients.

a. Calculate the zeros of the denominators in polar form and place them on the z-plane with respect to the unit circle. Are they inside the unit circle? Plot the magnitude response of the filter in dB for $0 \le \omega \le \pi/2$. Then, using the function **filter** calculate the filter's impulse response $h(n)$ for $n = 0, 1, \ldots, 599$, and plot it versus n.

b. Consider the cascade form and round the coefficients of the individual second-order sections to two-digit accuracy. Then, repeat all questions of part (a). Discuss the stability of the resulting filter and compare its magnitude response with that of part (a).

c. Consider the fourth-order direct-form denominator polynomial and round its coefficients to two-digit accuracy. Then, again, repeat all questions of part (a). Discuss the stability of the resulting filter and compare its magnitude response with that of part (a), if such comparison makes sense (why or why not?).

3. It is desired to design and implement a digital lowpass Chebyshev type-1 filter having the following specifications: sampling rate of 20 kHz, passband frequency of 1 kHz, stopband frequency of 2 kHz, passband attenuation of 1 dB, and stopband attenuation of 50 dB. The design method described in section 11.6.6 of the I2SP book [1], generates a sixth-order transfer function in cascade and direct forms:

$$\begin{aligned} H(z) &= G \cdot \frac{(1 + z^{-1})^2}{1 + a_{11}z^{-1} + a_{12}z^{-2}} \cdot \frac{(1 + z^{-1})^2}{1 + a_{21}z^{-1} + a_{22}z^{-2}} \cdot \frac{(1 + z^{-1})^2}{1 + a_{31}z^{-1} + a_{32}z^{-2}} \\ &= G \cdot \frac{(1 + z^{-1})^6}{1 + a_1z^{-1} + a_2z^{-2} + a_3z^{-3} + a_4z^{-4} + a_5z^{-5} + a_6z^{-6}} \end{aligned}$$

where the normalization gain factor is $G = 8.07322364 \times 10^{-7}$ and is to remain fixed in the remainder of this problem. The full precision coefficients are given below:

$$\begin{aligned} \mathbf{a}_1 &= [1, a_{11}, a_{12}] = [1\,,\ -1.86711351\,,\ 0.96228613] \\ \mathbf{a}_2 &= [1, a_{21}, a_{22}] = [1\,,\ -1.84679822\,,\ 0.89920764] \\ \mathbf{a}_3 &= [1, a_{31}, a_{32}] = [1\,,\ -1.85182222\,,\ 0.86344488] \end{aligned} \tag{2}$$

where "full precision" means eight-digit precision.

a. Carry out the filter design procedure, and verify that the above 8-digit coefficients, and the gain G, are correct. Then, calculate the direct-form denominator coefficient vector:

$$\mathbf{a} = \mathbf{a}_1 * \mathbf{a}_2 * \mathbf{a}_3 = [1, a_1, a_2, a_3, a_4, a_5, a_6]$$

as well as the normalized lattice coefficients, γ_p, c_p, using your function **dir2lat**. Replace all these coefficients by their 8-digit approximations, and verify that the resulting values are given correctly by the following table,

p	a_p	γ_p	Gc_p
0	1.00000000		0.25668361
1	−5.56573395	0.98144049	0.15971532
2	13.05062482	−0.98561779	0.03518539
3	−16.49540451	0.98529921	0.00411257
4	11.84993647	−0.98329869	0.00028542
5	−4.58649943	0.96911938	0.00001405
6	0.74713457	−0.74713457	0.00000081

b. Calculate the six zeros of $\mathbf{a}$ and their magnitudes, and display them on the z-plane.

Calculate and plot in dB the magnitude response of this filter over the interval $0 \le f \le 1.2$ kHz, and verify that it meets the prescribed 1 dB passband specification.

Using the normalized lattice form implemented by your function, **nlfilt**, calculate and plot the impulse response of this filter, $h(n)$, for $n = 0, 1, \ldots, 200$. Moreover, calculate it also using the function **filter**, and compare the two methods.

c. Determine the quantized version of $\mathbf{a}$, say $\hat{\mathbf{a}}$, rounded to five-digit fractional accuracy and calculate its 6 zeros and their magnitudes. Are all the zeros inside the unit circle? Compare the zeros with the full precision zeros. Using $\hat{\mathbf{a}}$ as the filter denominator vector, calculate in dB the magnitude response.

Then, round the normalized lattice coefficients γ_p, c_p to 5-digit fractional precision and calculate the corresponding magnitude response of the normalized lattice using your function **lat2freq**, and plot it together with the above 5-digit direct form, and with the exact response of part (b).

d. Determine the quantized version of $\mathbf{a}$, say $\hat{\mathbf{a}}$, rounded to four-digit fractional accuracy and calculate its 6 zeros and their magnitudes. Are all the zeros inside the unit circle? You should find that the direct form is now unstable and therefore the calculation of its magnitude response is meaningless (why is that?).

However, the normalized lattice and cascaded forms remain stable. To see this, round the normalized lattice coefficients γ_p, c_p to 4-digit precision, and verify that the resulting γ_p's have magnitude less than one, which guarantees stability by the Schur-Cohn stability criterion. Then, calculate the magnitude response of the 4-digit normalized lattice using your function **lat2freq**.

In addition, round to four-digit fractional precision the *individual* second-order coefficient vectors $\mathbf{a}_i$, $i = 1, 2, 3$, of the cascade form of Eq. (2), and compute their zeros displaying them on the z-plane, and also compute the corresponding magnitude response.

On the same graph, plot the 4-digit normalized lattice and cascade magnitude responses, and the exact response.

e. Repeat all the questions of part (d) using 3-digit fractional precision.
In addition, compute and plot the corresponding impulse responses, $h(n)$, for $n = 0, 1, \ldots, 200$, of the 3-digit normalized lattice and cascade forms, and the exact response.

3. References

These references may be found on Sakai Resources.

[1] S. J. Orfanidis, *Introduction to Signal Processing*, Prentice Hall, Upper Saddle River, NJ, 1996, available online, 2010.

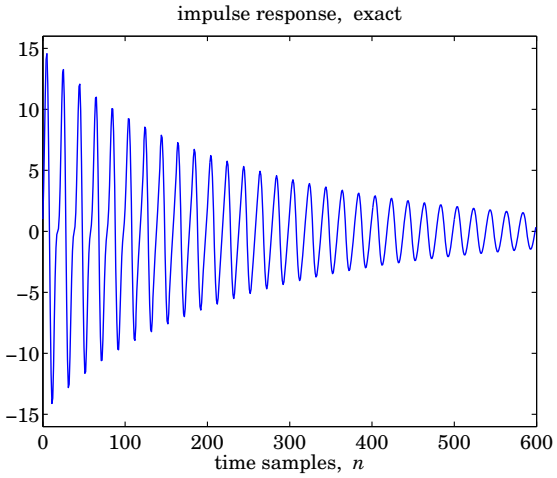
[2] A. H. Gray and J. D. Markel, "A Normalized Digital Filter Structure," *IEEE Trans. Acoust., Speech, Signal Process.*, **ASSP-23**, 268 (1975).

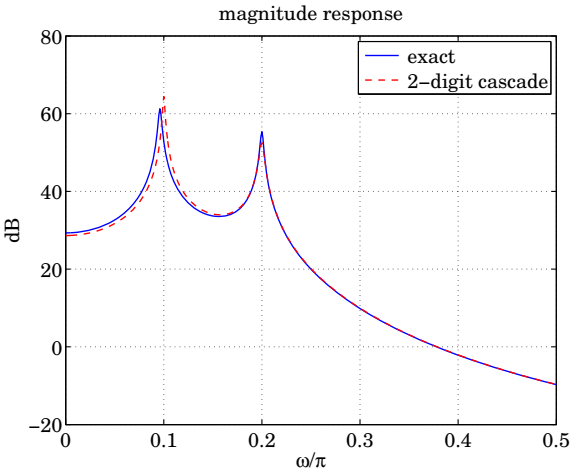
[3] A. H. Gray and J. D. Markel, "Roundoff Noise Characteristics of a Class of Orthogonal Polynomial Structures," *IEEE Trans. Acoust., Speech, Signal Process.*, **ASSP-23**, 473 (1975).

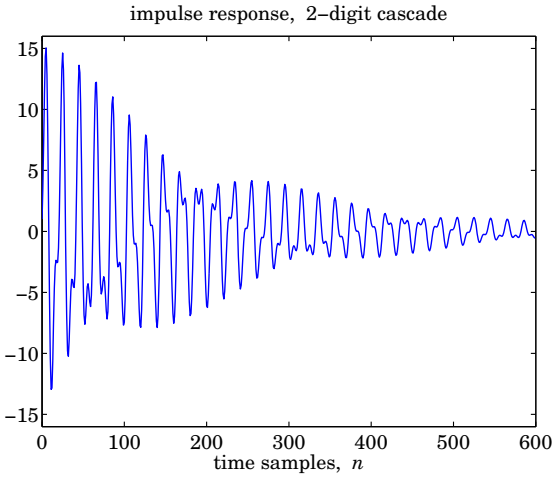
[4] A. H. Gray and J. D. Markel, "Digital Lattice and Ladder Filter Synthesis," *IEEE Trans. Audio Electroacoust.*, **AU-21**, 491 (1973).

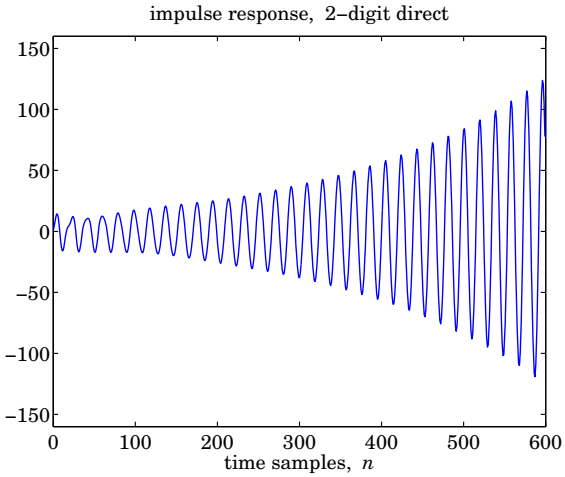
[5] J. A. Moorer, "48-Bit Integer Processing Beats 32-Bit Floating Point for Professional Audio Applications," Presented at the 107th Convention of the AES, New York, September 1999, *AES Preprint 5038*.

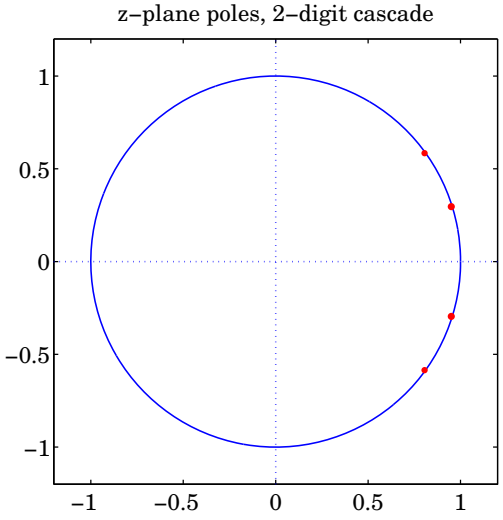
Example Graphs

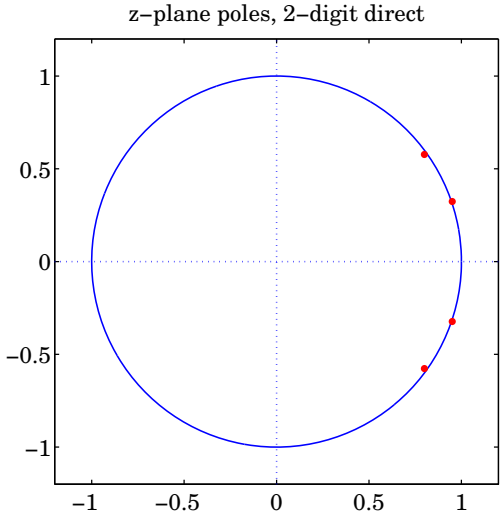


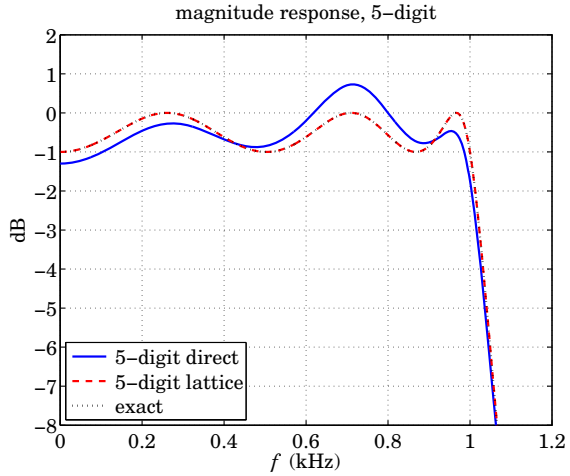


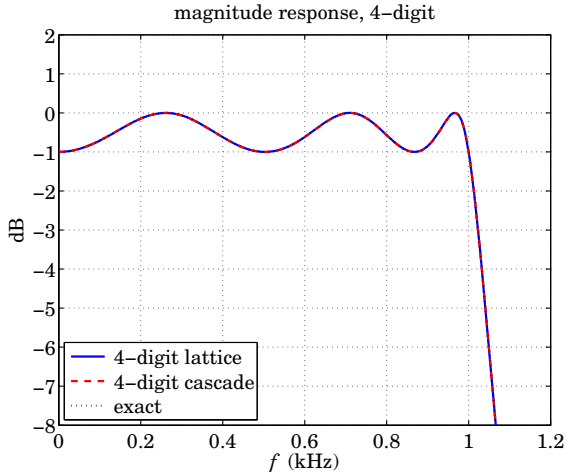


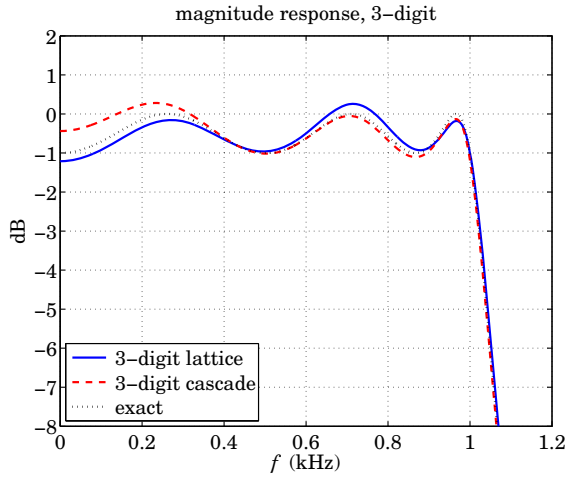


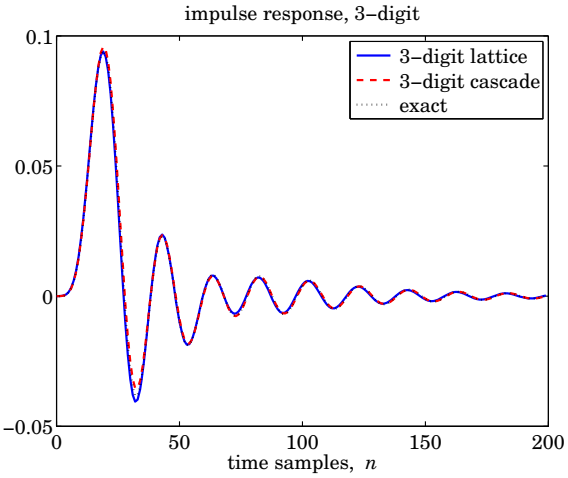


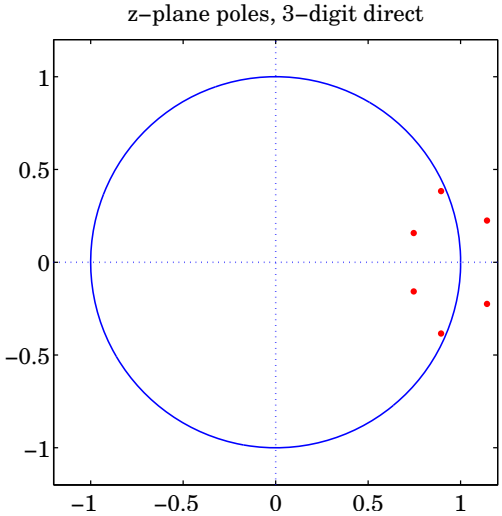


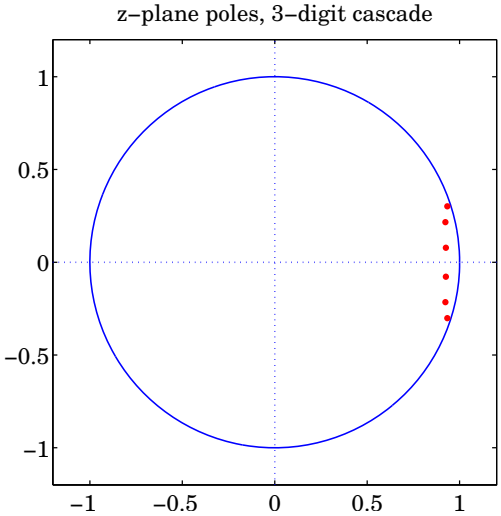

magnitude response, 5–digit
dB
f (kHz)
5–digit direct
5–digit lattice
exact


magnitude response, 4–digit
dB
f (kHz)
4–digit lattice
4–digit cascade
exact


magnitude response, 3–digit
dB
f (kHz)
3–digit lattice
3–digit cascade
exact

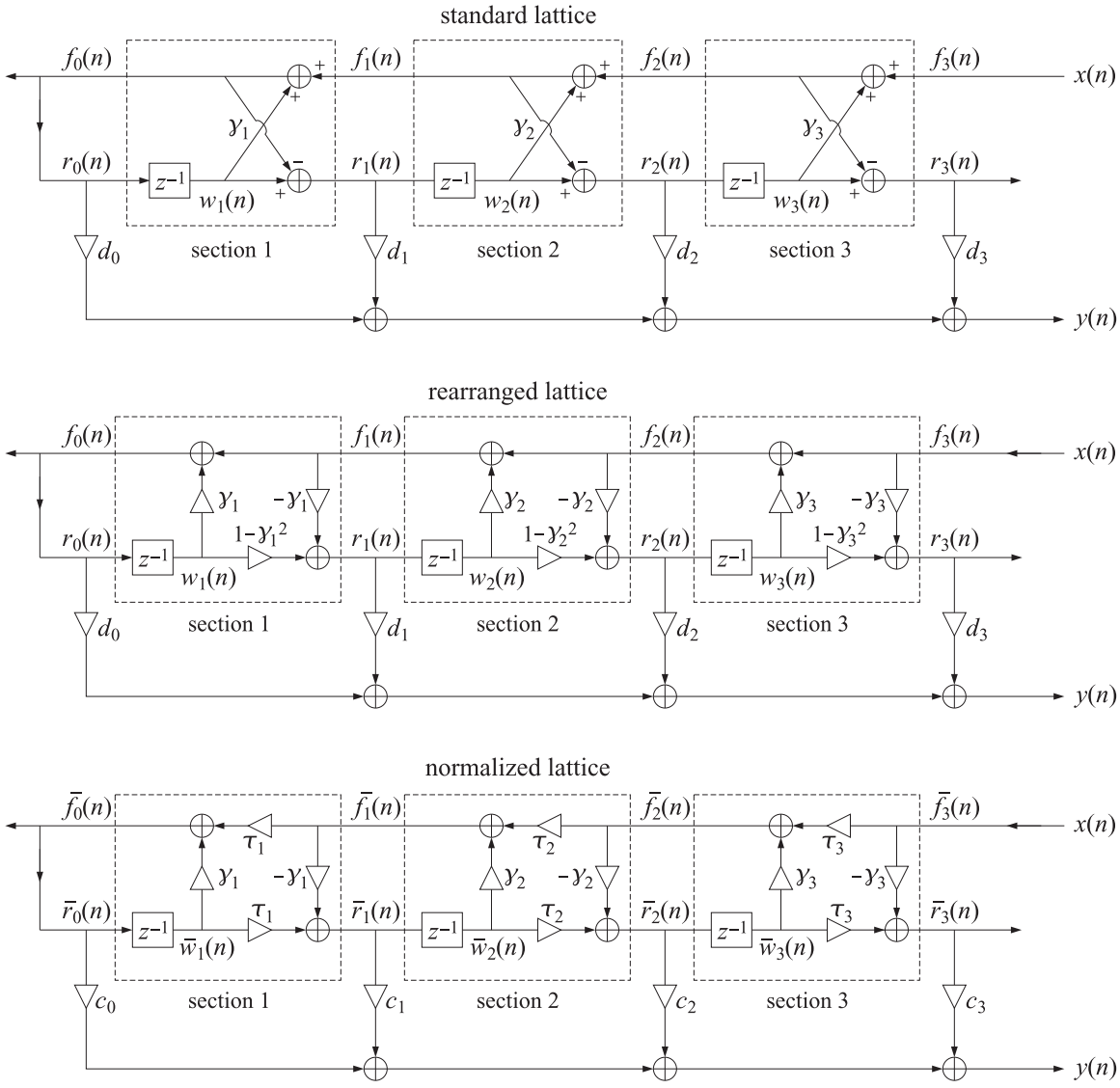

impulse response, 3–digit
time samples, n
3–digit lattice
3–digit cascade
exact


z–plane poles, 3–digit direct


z–plane poles, 3–digit cascade

Appendix
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 $p = 0, 1, \ldots, M$,

$$\boxed{\begin{aligned} f_p(n) &= \sigma_p \bar{f}_p(n) \\ r_p(n) &= \sigma_p \bar{r}_p(n) \\ d_p &= \frac{\sigma_M}{\sigma_p}\, c_p \end{aligned}} \qquad \text{where} \quad \begin{aligned} &\sigma_0 = 1 \\ &\sigma_p = \tau_p \sigma_{p-1} = \tau_1 \tau_2 \cdots \tau_p, \quad p = 1, 2, \ldots, M \\ &\tau_p = (1 - \gamma_p^2)^{1/2}, \quad p = 1, 2, \ldots, M \end{aligned} \tag{3}$$

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

for each time instant n, do,

$f_M(n) = x(n)$
$f_{M-1}(n) = f_M(n) + \gamma_M w_M(n)$
$r_M(n) = w_M(n) - \gamma_M f_{M-1}(n)$
$y(n) = d_M r_M(n)$

for $p = M-1, M-2, \ldots, 1$, do,

$f_{p-1}(n) = f_p(n) + \gamma_p w_p(n)$
$r_p(n) = w_p(n) - \gamma_p f_{p-1}(n)$
$w_{p+1}(n+1) = r_p(n)$
$y(n) = y(n) + d_p r_p(n)$

end

$r_0(n) = f_0(n)$
$w_1(n+1) = r_0(n)$
$y(n) = y(n) + d_0 r_0(n)$

end

$\Rightarrow$

for each input sample x, do,

$f = x$
$f = f + \gamma_M w_M$
$r = w_M - \gamma_M f$
$y = d_M r$

for $p = M-1, M-2, \ldots, 1$, do,

$f = f + \gamma_p w_p$
$r = w_p - \gamma_p f$
$w_{p+1} = r$
$y = y + d_p r$

end

$r = f$
$w_1 = r$
$y = y + d_0 r$

end

(4)

and for the rearranged case,

for each time instant n, do,

$f_M(n) = x(n)$
$r_M(n) = (1 - \gamma_M^2) w_M(n) - \gamma_M f_M(n)$
$f_{M-1}(n) = f_M(n) + \gamma_M w_M(n)$
$y(n) = d_M r_M(n)$

for $p = M-1, M-2, \ldots, 1$, do,

$r_p(n) = (1 - \gamma_p^2) w_p(n) - \gamma_p f_p(n)$
$f_{p-1}(n) = f_p(n) + \gamma_p w_p(n)$
$w_{p+1}(n+1) = r_p(n)$
$y(n) = y(n) + d_p r_p(n)$

end

$r_0(n) = f_0(n)$
$w_1(n+1) = r_0(n)$
$y(n) = y(n) + d_0 r_0(n)$

end

$\Rightarrow$

for each input sample x, do,

$f = x$
$r = (1 - \gamma_M^2) w_M - \gamma_M f$
$f = f + \gamma_M w_M$
$y = d_M r$

for $p = M-1, M-2, \ldots, 1$, do,

$r = (1 - \gamma_p^2) w_p - \gamma_p f$
$f = f + \gamma_p w_p$
$w_{p+1} = r$
$y = y + d_p r$

end

$r = f$
$w_1 = r$
$y = y + d_0 r$

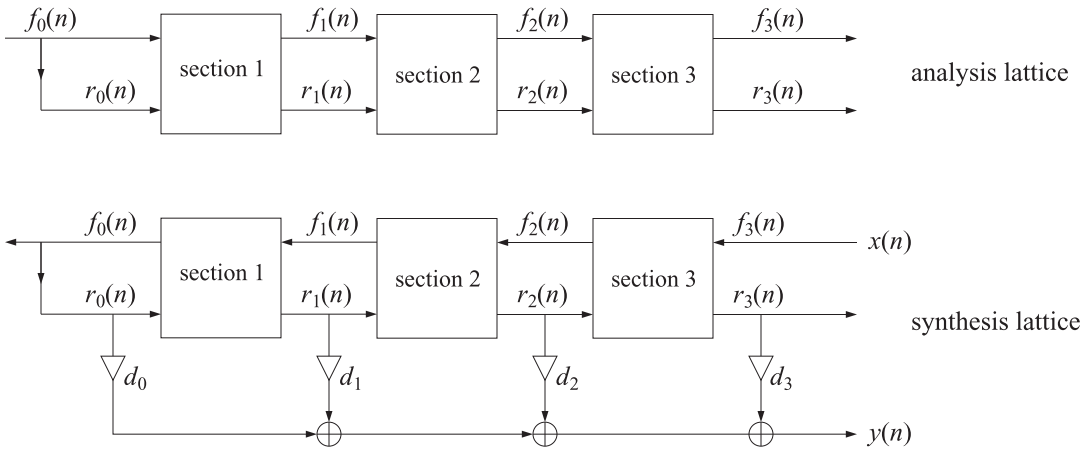
end

(5)

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 $A_0(z) = A_0^R(z) = 1$,

$$\begin{aligned} F_p(z) &= A_p(z) F_0(z) \\ R_p(z) &= A_p^R(z) F_0(z) \end{aligned} \tag{6}$$

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

$$\begin{bmatrix} A_p(z) \\ A_p^R(z) \end{bmatrix} = \begin{bmatrix} 1 & -\gamma_p z^{-1} \\ -\gamma_p & z^{-1} \end{bmatrix} \begin{bmatrix} A_{p-1}(z) \\ A_{p-1}^R(z) \end{bmatrix} \qquad \text{(forward Levinson)} \tag{7}$$

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

$$\mathbf{a}_p = \begin{bmatrix} \mathbf{a}_{p-1} \\ 0 \end{bmatrix} - \gamma_p \begin{bmatrix} 0 \\ \mathbf{a}_{p-1}^R \end{bmatrix}, \quad p = 1, 2, \ldots, M \qquad \text{(forward Levinson)} \tag{8}$$

where $\mathbf{a}_p = [1, a_{p1}, a_{p2}, \ldots, a_{pp}]^T$ 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 $\mathbf{a}_M = [1, a_{M1}, a_{M2}, \ldots, a_{MM}]^T$ and proceeds downwards to order-0, extracting the reflection coefficients γ_p in the process. We have for, $p = M, M-1, \ldots, 1$,

$$\gamma_p = -a_{p,p}, \quad \mathbf{a}_{p-1} = \frac{\mathbf{a}_p + \gamma_p \mathbf{a}_p^R}{1 - \gamma_p^2} \qquad \text{(backward Levinson)} \tag{9}$$

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,

$$R_p(z) = A_p^R(z)F_0(z) = \frac{A_p^R(z)}{A_M(z)}A_M(z)F_0(z) = \frac{A_p^R(z)}{A_M(z)}F_M(z) = \frac{A_p^R(z)}{A_M(z)}X(z) \tag{10}$$

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

$$Y(z) = \sum_{p=0}^{M} d_p R_p(z) = \frac{\sum_{p=0}^{M} d_p A_p^R(z)}{A_M(z)}X(z) \tag{11}$$

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

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{p=0}^{M} d_p A_p^R(z)}{A_M(z)} = \frac{B_M(z)}{A_M(z)} \tag{12}$$

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

$$B_M(z) = \sum_{p=0}^{M} d_p A_p^R(z) \tag{13}$$

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

$$\mathbf{b}_M = \sum_{p=0}^{M} d_p \mathbf{a}_p^R = [\mathbf{a}_0^R, \mathbf{a}_1^R, \cdots, \mathbf{a}_M^R] \begin{bmatrix} d_0 \\ d_1 \\ \vdots \\ d_M \end{bmatrix} = A\mathbf{d} \tag{14}$$

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. (14) becomes,

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

Solving for the ladder weights in terms of $\mathbf{b}$ is efficiently done with MATLAB's backslash operator,

$$\mathbf{d} = A^{-1}\mathbf{b} = A \backslash \mathbf{b} \tag{15}$$

Finally, we note that in the normalized form, because of the scale factors of Eq. (3), the overall output and resulting transfer function are as follows, with input $X(z) = \bar{F}_M(z) = \sigma_M^{-1}F_M(z) = \sigma_M^{-1}A_M(z)F_0(z)$,

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

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

$$H(z) = \frac{\sum_{p=0}^{M} c_p \sigma_p^{-1} \sigma_M A_p^R(z)}{A_M(z)} = \frac{B_M(z)}{A_M(z)} \tag{16}$$

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. (3),

$$d_p = c_p \sigma_p^{-1} \sigma_M, \quad p = 0, 1, \ldots, M \tag{17}$$

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 (8) and (9), 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)
```