Lattice Filters

Table of Contents

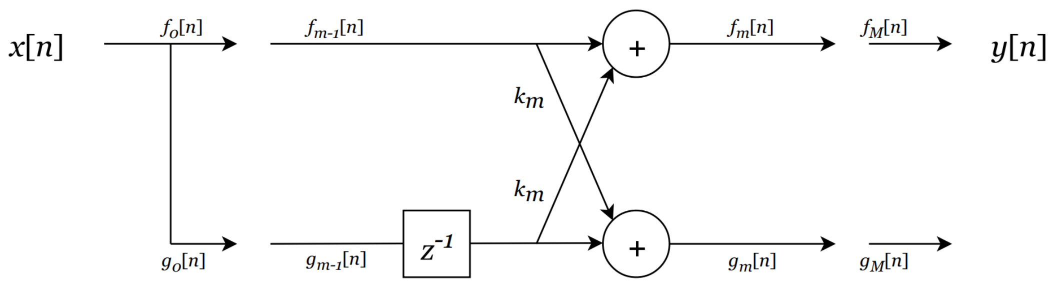
Lattice Structures 1
All-zero lattice structure 1
Single-stage filter 3
Two stage filter 3
Three stage filter 4
How to find reflection coefficients from the impulse response? 4
All-pole lattice structure 5
What are the advantages of lattice structures? 7
[✓] ADSI Problem 2.1: Implementation of All-zero lattice filters 7
[✓] ADSI Problem 2.2: Find impulse response of an all-zero lattice filter 8
[✓] ADSI Problem 2.3: Find the reflection coefficients for an all-zero lattice filter 10
[✓] ADSI Problem 2.5: All-zero lattice filter, find coefficients from system function (division by zero problem) 13
[✓] ADSI Problem 2.7: Find system function an all-pole filter from reflection coefficients 15
[✓] ADSI Problem 2.10: Linear prediction and lattice filters 16
1) Compute autocorrelation from a signal 16
2) Compute the coefficients for the optimum 2nd-order linear predictor 19
3) Compare the linear predictor with a two-stage all-zero lattice filter 20
4) Compute reflection coefficients from the linear predictor 21
5) Compute prediction error 22
Exam 2016 Problem 3: All-pole Lattice Filter 23
1) Explain why a system is a digital notch filter 23
2) Find reflection coefficients for the all-pole lattice filter 24
3) Determine the group delay in the digital notch filter 25
Exam 2017 Problem 2: Lattice Structures 27
1) Determine the two equations relating outputs to inputs 27
2) When is the first non-zero sample observed at the output? 28
3) Calculate the transfer function for a one-stage lattice filter 29
Exam 2014 Problem 1 30
Functions 30

Lattice Structures

A lattice filter is an example of an all-pass filter typically used the analysis and synthesis of speech signals.

All-zero lattice structure

An ***all-zero*** lattice models an FIR system.



Each section has two inputs ($f_{m-1}[n]$ and $g_{m-1}[n-1]$) and two outputs $(f_m[n], g_m[n])$.

The *m*'th section/stage can be computed as follows:

$$f_m[n] = f_{m-1}[n] + k_m g_{m-1}[n-1], \quad m = 1, 2, \ldots, M \qquad (9.55a)$$

$$g_m[n] = k_m f_{m-1}[n] + g_{m-1}[n-1]. \quad m = 1, 2, \ldots, M \qquad (9.55b)$$

The overall system input and output are given by:

$$x[n] = f_0[n] = g_0[n], \qquad (9.56a)$$

$$y[n] = f_M[n]. \qquad (9.56b)$$

In general, the outputs of the *m*th section correspond to two FIR filters with the same coefficients but in reverse order:

$$f_m[n] = \sum_{i=0}^{m} a_i^{(m)} x[n-i], \quad m = 1, 2, \ldots, M \qquad (9.64a)$$

$$g_m[n] = \sum_{i=0}^{m} a_{m-i}^{(m)} x[n-i]. \quad m = 1, 2, \ldots, M \qquad (9.64b)$$

The system functions of these all-zero FIR filters are given by:

$$A_m(z) \triangleq \frac{F_m(z)}{F_0(z)} = \sum_{i=0}^{m} a_i^{(m)} z^{-i}, \quad a_0^{(0)} = 1 \qquad (9.65a)$$

$$B_m(z) \triangleq \frac{G_m(z)}{G_0(z)} = \sum_{i=0}^{m} a_{m-i}^{(m)} z^{-i} \triangleq \sum_{i=0}^{m} b_i^{(m)} z^{-i}. \qquad (9.65b)$$

Suppose we have a FIR system of the form:

$$H(z) = \sum_{k=0}^{M} h[k] z^{-1}$$

We can start taking the z-transforms.

Since $z[n] = f_0[n] = g_0[n]$, taking the z-transform is straightforward:

$$X(z) = F_0(z) = G_0(z)$$

The difference equations of the *m*th section is:

$$f_m[n] = f_{m-1}[n] + k_m g_{m-1}[n-1]$$
$$g_m[n] = k_m f_{m-1}[n] + g_{m-1}[n-1]$$

Taking the z-transform of the mth section gives us:

$$F_m(z) = F_{m-1}(z) + k_m G_{m-1}(z) z^{-1}$$
$$G_m(z) = k_m F_{m-1}(z) + G_{m-1}(z) z^{-1}$$

We can normalise these two z-transforms as follows:

$$A_m(z) = \frac{F_m(z)}{F_0(z)} = A_{m-1}(z) + k_m B_{m-1}(z) z^{-1}$$

$$B_m(z) = \frac{G_m(z)}{G_0(z)} = k_m A_{m-1}(z) + B_{m-1}(z) z^{-1}$$

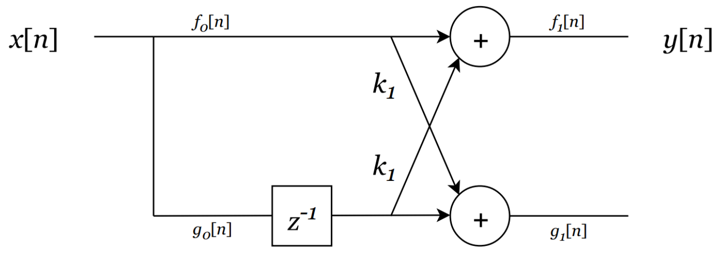
By the way we normalise, we notice that $A_0(z) = B_0(z) = 1$ because

$$A_0 = \frac{F_0(z)}{F_0(z)} = 1 \text{ and } B_0 = \frac{G_0(z)}{G_0(z)} = 1$$

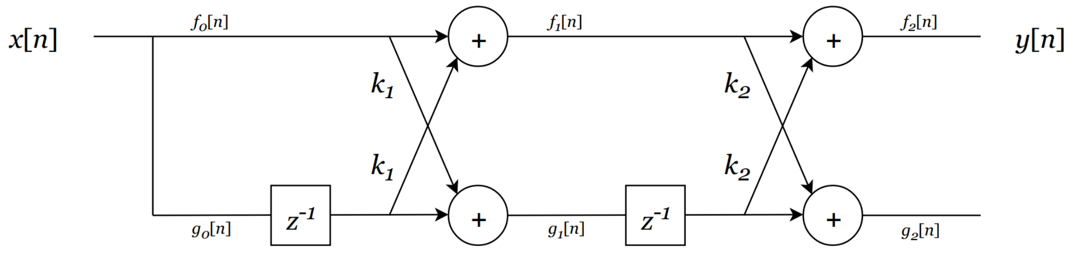
Because of the normalisation, we can write:

$$H(z) = h[0] A_M(z)$$

Single-stage filter



Two stage filter



Compute the difference equations for the two FIR systems:

$$f_2[n] = f_1[n] + k_2 g_1[n-1]$$
$$g_2[n] = k_2 f_1[n] + g_1[n-1]$$

$f_1[n] = f_0[n] + k_1 g_0[n-1]$
$g_1[n] = k_1 f_0[n] + g_0[n-1]$

If we substitute we get:

$f_2[n] = f_1[n] + k_2 g_1[n-1]$
$f_2[n] = f_0[n] + k_1 g_0[n-1] + k_2(k_1 f_0[n] + g_0[n-1])$

Since $x[n] = f_0[n] = g_0[n]$ and $y[n] = f_2[n]$ then we can rewrite the difference equation for two stage all-zero lattice filter as:

$$y[n] = x[n] + k_1(1+k_2)x[n-1] + k_2 x[n-2]$$

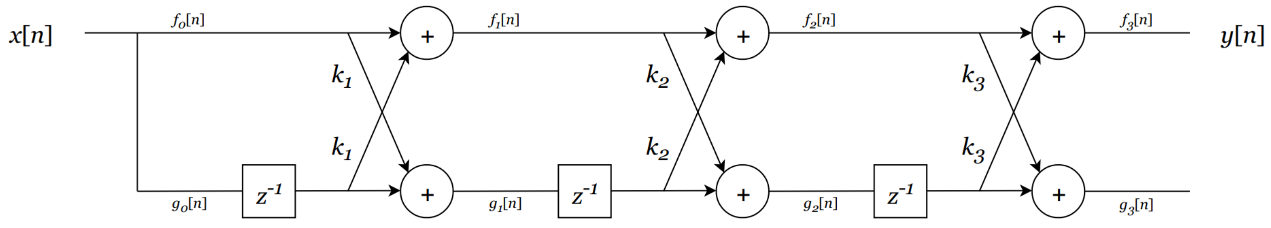
Taking the z-transform, we have:

$$Y(z) = X(z) + k_1(1+k_2)z^{-1}X(z) + k_2 z^{-2}X(z)$$

We can compute the transfer function:

$$H(z) = \frac{Y(z)}{X(z)} = 1 + k_1(1+k_2)z^{-1} + k_2 z^{-2}$$

Three stage filter



How to find reflection coefficients from the impulse response?

Suppose we want to determine the reflection coefficients $k_m, m = 1, 2, \cdots, M$ from the following an Mth-order normalised FIR filter:

$$H(z) = \sum_{k=0}^{M} h[k]z^{-1}$$

First, we can define the filter coefficients as follows:

$$a_k = \frac{h[k]}{h[0]} \text{ where } k = 0, 1, \cdots, M$$

The recursive algorithm works as follows:

1. Compute $A_M(z) = \dfrac{H(z)}{h[0]}$
2. Compute $k_M = a_M$ where a_M is the last coefficients of $A_M(z)$. For example, if $A_M(z) = 1 + 0.06z^{-1} - 0.42z^{-2} + 0.5z^{-3}$ then $k_M = 0.5$
3. Compute $B_M(z)$ by flipping the coefficients of $A_M(z)$

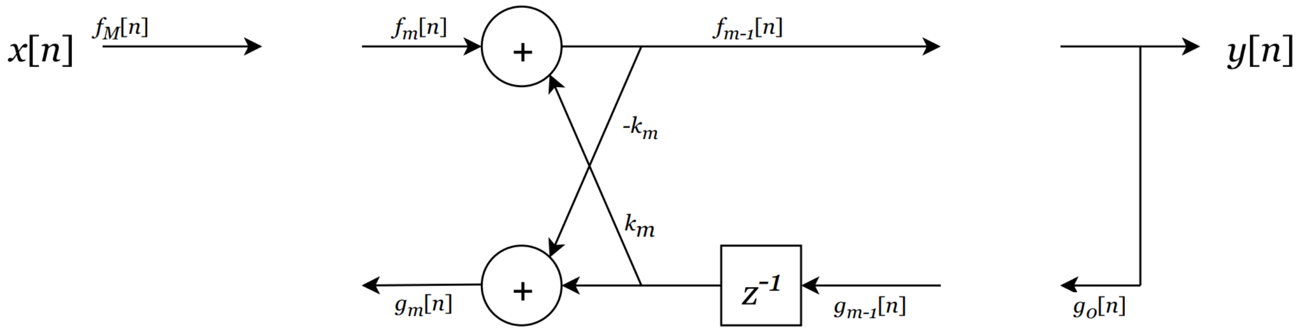
4. Set $m = M$
5. Compute $A_{m-1}(z) = \frac{1}{1-k_m^2}\left[A_m(z) - k_m B_m(z)\right]$. Notice that this is where the algorithm fails because if $k_m = 1 \rightarrow k_m^2 = 1$ then we will have division by zero.
6. Compute $k_{m-1} = a_{m-1}$ where a_{m-1} is the last coefficients of $A_{m-1}(z)$
7. Compute $B_{m-1}(z)$ by flipping the coefficients of $A_{m-1}(z)$. Alternatively, compute $B_{m-1}(z) = z^{-m-1}A_{m-1}\left(\frac{1}{z}\right)$
8. Set $m = m - 1$
9. Go to step 5 if $m \neq 0$
10. We know that $A_0(z) = B_0(z) = 1$

See page 513

All-pole lattice structure

An ***all-pole*** lattice models an IIR system.

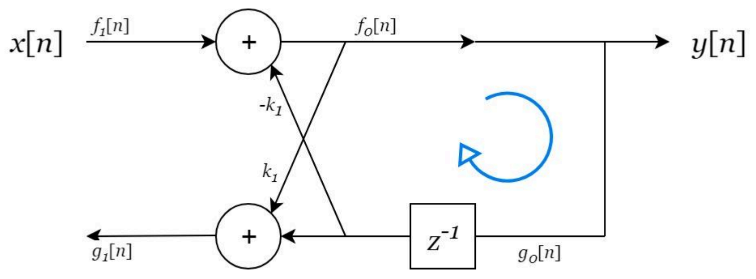
The figure below shows how an all-pole lattice structure looks like



$f_{m-1}[n] = f_m[n] - k_m g_{m-1}[n-1]$

$g_m[n] = k_m f_{m-1}[n] + g_{m-1}[n-1]$

When $M = 1$



The difference equation for when $M = 1$ is given as:

$$f_0[n] = f_1[n] - k_1 g_0[n-1]$$

Since $f_0[n] = g_0[n] = y[n]$ and $f_1[n] = x[n]$ then we have the following difference equation

$$y[n] = x[n] - k_1 y[n-1]$$

This is the difference equation for a IIR filter!

If we take the z-transform, we get:

$$Y(z) = X(z) - k_1 Y(z) z^{-1}$$

We can compute the transfer function:

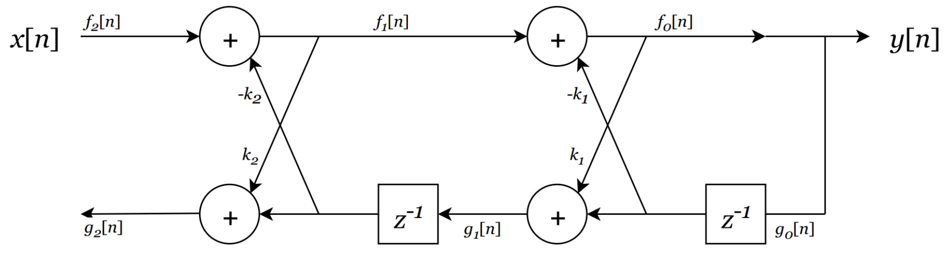
$$H(z) = \frac{Y(z)}{X(z)} = \frac{1}{1 + k_1 z^{-1}} = \frac{1}{A(z)}$$

Let us look at the other system $g_1[n]$:

$$g_1[n] = k_1 f_0[n] + g_0[n-1]$$
$$g_1[n] = k_1 y[n] + y[n-1]$$

It is clear that $g_1[n]$ is a FIR filter of the output.

When $M = 2$



We compute the difference equations from the output:

$$f_0[n] = f_1[n] - k_1 g_0[n-1]$$

$$g_1[n] = k_1 f_0[n] + g_0[n-1]$$

The next stage:

$$f_1[n] = f_2[n] - k_2 g_1[n-1]$$

$$g_2[n] = k_2 f_1[n] + g_1[n-1]$$

We can substitute:

$$f_0[n] = f_2[n] - k_2 g_1[n-1] - k_1 g_0[n-1]$$
$$f_0[n] = f_2[n] - k_2(k_1 f_0[n-1] + g_0[n-2]) - k_1 g_0[n-1]$$

Since $f_0[n] = g_0[n] = y[n]$ and $f_2[n] = x[n]$ then we have:

$$y[n] = x[n] - k_2(k_1 y[n-1] + y[n-2]) - k_1 y[n-1]$$
$$y[n] = x[n] - k_1(1+k_2)y[n-1] - k_2 y[n-2]$$

If we take the z-transform, we get:

$$Y(z) = X(z) - k_1(1+k_2)Y(z)z^{-1} - k_2 Y(z)z^{-2}$$

The transfer function becomes:

$$\frac{Y(z)}{X(z)} = \frac{1}{1 + k_1(1+k_2)z^{-1} + k_2 z^{-2}}$$

What are the advantages of lattice structures?

When is an all-pole filter stable?

The all-zero and all-pole lattice structures require twice the number of multiplications per output sample as the direct forms. However, they have two unique advantages compared to direct form structures.

The first advantage follows from the following theorem:

Theorem 9.4.1 The roots of the polynomial $A_M(z)$ are inside the unit circle if and only if

$$|k_m| < 1. \quad m = 1, 2, \ldots, M \qquad (9.87)$$

Advantage 1: If the lattice parameters satisfy Eq. (9.87) then the all-zero lattice filter is **minimum-phase** and the all-pole filter is **stable**.

Advantage 2: Lattice structures are insensitive to quantization of the *k*-parameters.

[✔] ADSI Problem 2.1: Implementation of All-zero lattice filters

Work you way through the MATLAB code in Figure 9.26 and make sure you understand what is going on.

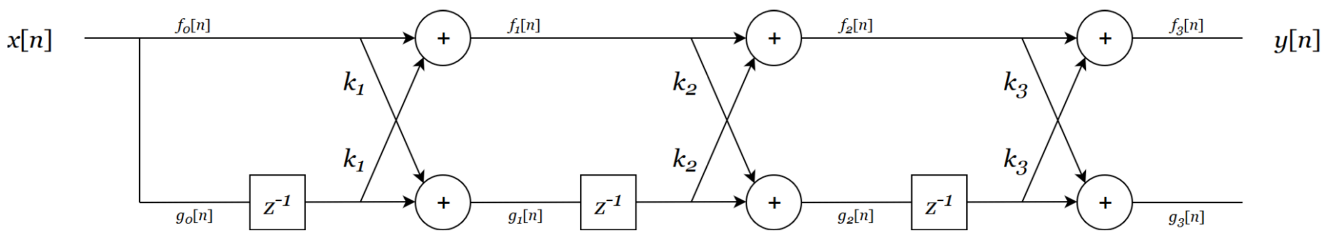
```
function [y] = azlatfilt(k, x, G)
% AZLATFILT      Function for implementation of an all-zero FIR lattice filter
% From Figure 9.26 (p 515)

    M = length(k);

    % Create an array for the sequence f_m[n]
    f = zeros(1,M);

    % Create an array for the sequence g_m[n]
    g = f;

    % Create an array for the sequence g_m[n-1]
    oldg = zeros(1,M);

    % Keeps track of x[n-1]
    oldx = 0;
    x = G*x;
    y = zeros(size(x));

    for n=1:length(x)
        % Compute f1[n] = x[n] + k1x[n-1] -> Equation (9.57a)
        f(1) = x(n)+k(1)*oldx;

        % Compute g1[n] = k1x[n] + x[n-1] -> Equation (9.57b)
        g(1) = k(1)*x(n)+oldx;
        oldx = x(n);
        for m = 2:M
            % Compute: fm[n] = fm?1[n] + km gm?1[n ? 1] -> Equation (9.55a)
            f(m) = f(m-1)+k(m)*oldg(m-1);

            % Compute: gm[n] = km fm?1[n] + gm?1[n ? 1] -> Equation (9.55b)
            g(m) = k(m)*f(m-1)+oldg(m-1);

            % Delay
            oldg(m-1) = g(m-1);
        end

        % y[n] = fM[n] -> Equation (9.56b)
        y(n) = f(M);
    end
end
```

[✔] ADSI Problem 2.2: Find impulse response of an all-zero lattice filter

Consider an all-zero (FIR) lattice filter with $k_1 = 0.65$, $k_2 = -0.34$ and k_3=0.8.

1. Find the impulse response of the filter by tracing an impulse $x(n) = \delta(n)$ through the filter.

Since the lattice filter has three reflection coefficients, it is a 3-stage all-zero lattice filter.



Here are the formulas needed to trace through the filter.

$$x[n] = f_0[n] = g_0[n],$$

$$y[n] = f_M[n].$$

$$f_m[n] = f_{m-1}[n] + k_m g_{m-1}[n-1], \quad m = 1, 2, \ldots, M$$

$$g_m[n] = k_m f_{m-1}[n] + g_{m-1}[n-1]. \quad m = 1, 2, \ldots, M$$

The input is $x[n] = \delta[n] = [1, 0, 0, 0]$

$n = 1$
$f_0(n) = g_0(n) = x(1) = 1$
$f_1(n) = f_0(n) + k_1 g_0(n-1) = 1 + 0 = 1$
$g_1(n) = k_1 f_0(n) + g_0(n-1) = 0.65 + 0 = 0.65$
$f_2(n) = f_1(n) + k_2 g_1(n-1) = 1 - 0.34 \cdot 0 = 1$
$g_2(n) = k_2 f_1(n) + g_1(n-1) = -0.34 \cdot 1 + 0 = -0.34$
$f_3(n) = f_2(n) + k_3 g_2(n-1) = 1 + 0.8 \cdot 0.0 = \underline{\mathbf{1}}$
$g_3(n) = k_3 f_2(n) + g_2(n-1) = 0.8 \cdot 1 + 0 = 0.8$

$n = 2$
$f_0(n) = g_0(n) = x(2) = 0$
$f_1(n) = 0 + 0.65 \cdot 1 = 0.65$
$g_1(n) = 0 + 1 = 1$
$f_2(n) = 0.65 - 0.34 \cdot 0.65 = 0.429$
$g_2(n) = -0.34 \cdot 0.65 + 0.65 = 0.429$
$f_3(n) = 0.429 + 0.8 \cdot -0.34 = \mathbf{0.157}$
$g_3(n) = 0.8 \cdot 0.429 - 0.34 = 0.0032$

$n = 4$
$f_0(n) = g_0(n) = x(2) = 0$
$f_1(n) = 0$
$g_1(n) = 0$
$f_2(n) = 0$
$g_2(n) = 0$
$f_3(n) = \mathbf{0.8}$
$g_3(n) = 1$

We can also use the MATLAB function **azlatfilt** given in Figure 9.26 in the book.

```
k = [0.65, -0.34, 0.8];
x = [1, 0, 0, 0];
G = 1;
y = azlatfilt(k, x, G)
```

```
y = 1×4
    1.0000    0.1570    0.0032    0.8000
```

The impulse response of the given filter is:

$h[n] = [1, 0.157, 0.0032, 0.8]$

[✔] ADSI Problem 2.3: Find the reflection coefficients for an all-zero lattice filter

An all-zero lattice filter has the following system function

$$H(z) = 1 + \frac{13}{24}z^{-1} + \frac{5}{8}z^{-2} + \frac{1}{3}z^{-3}$$

1. Find the reflection coefficients k_1, k_2 and k_3 for the corresponding lattice filter.

Can be computed using MATLAB function ***tf2latc*** to convert transfer function filter parameters to lattice filter for:

```
b = [1, 13/24, 5/8, 1/3];
a = 1;
k = tf2latc(b, a)
```

```
k = 3×1
    0.2500
    0.5000
    0.3333
```

We can also try the algorithm (Figure 9.24) from the book

```
k = fir2lat(b)'
```

```
k = 3×1
    0.2500
    0.5000
    0.3333
```

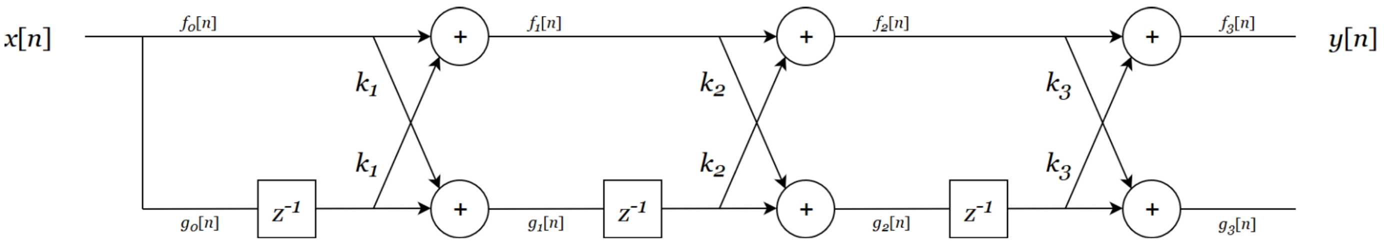
We can also do it by hand using the recursive algorithm:

The recursive algorithm works as follows:

1. Compute $A_M(z) = \dfrac{H(z)}{h[0]}$

2. Compute $k_M = a_M$ where a_M is the last coefficients of $A_M(z)$. For example, if $A_M(z) = 1 + 0.06z^{-1} - 0.42z^{-2} + 0.5z^{-3}$ then $k_M = 0.5$
3. Compute $B_M(z)$ by flipping the coefficients of $A_M(z)$
4. Set $m = M$
5. Compute $A_{m-1}(z) = \frac{1}{1-k_m^2}\left[A_m(z) - k_m B_m(z)\right]$. Notice that this is where the algorithm fails because if $k_m = 1 \rightarrow k_m^2 = 1$ then we will have division by zero.
6. Compute $k_{m-1} = a_{m-1}$ where a_{m-1} is the last coefficients of $A_{m-1}(z)$
7. Compute $B_{m-1}(z)$ by flipping the coefficients of $A_{m-1}(z)$. Alternatively, compute $B_{m-1}(z) = z^{-m-1}A_{m-1}\left(\frac{1}{z}\right)$
8. Set $m = m - 1$
9. Go to step 5 if $m \neq 0$
10. We know that $A_0(z) = B_0(z) = 1$

In this problem, we have a 3-stage all-zero lattice filter:



We are given its transfer function:

$$H(z) = 1 + \frac{13}{24}z^{-1} + \frac{5}{8}z^{-2} + \frac{1}{3}z^{-3}$$

Since this is a FIR system of the form:

$$H(z) = \sum_{k=0}^{M} h[k]z^{-1}$$

We know that $M = 3$, $h[k] = \left[1, \frac{13}{24}, \frac{5}{8}, \frac{1}{8}\right]$ and $h[0] = 1$.

Step 1: Compute $A_3(z)$

$$A_M(z) = A_3(z) = \frac{H(z)}{h[0]} = \frac{H(z)}{1} = 1 + \frac{13}{24}z^{-1} + \frac{5}{8}z^{-2} + \frac{1}{3}z^{-3}$$

Step 2: Compute k_3

The last coefficients of $A_3(z)$ is $\frac{1}{3}$ so $k_3 = \frac{1}{3}$

Step 3: Compute $B_3(z)$

Once we have $A_M(z)$, we can compute $B_M(z)$ by simply flipping the coefficients of $A_M(z)$.

$$B_M(z) = B_3(z) = \frac{1}{3} + \frac{5}{8}z^{-1} + \frac{13}{24}z^{2} + z^{-3}$$

Step 4: Set $m = M$

$$m = 3$$

Step 5: Compute $A_2(z)$

Using the formula $A_{m-1}(z) = \frac{1}{1-k_m^2}\left[A_m(z) - k_m B_m(z)\right]$ we can compute $A_2(z)$

$$A_2(z) = \frac{1}{1-k_3^2}\left[A_3(z) - k_3 B_3(z)\right]$$

$$A_2(z) = \frac{1}{1-\left(\frac{1}{3}\right)^2}\left[\left(1 + \frac{13}{24}z^{-1} + \frac{5}{8}z^{-2} + \frac{1}{3}z^{-3}\right) - \frac{1}{3}\left(\frac{1}{3} + \frac{5}{8}z^{-1} + \frac{13}{24}z^{2} + z^{-3}\right)\right]$$

$$A_2(z) = \frac{9}{8}\left[1 - \frac{1}{3}\cdot\frac{1}{3} + \left(\frac{13}{24} - \frac{1}{3}\cdot\frac{5}{8}\right)z^{-1} + \left(\frac{5}{8} - \frac{1}{3}\cdot\frac{13}{24}\right)z^{-2} + \left(\frac{1}{3} - \frac{1}{3}\right)z^{-3}\right]$$

$$A_2(z) = \frac{9}{8}\left[\frac{8}{9} + \frac{1}{3}z^{-1} + \frac{4}{9}z^{-2} + 0z^{-3}\right]$$

$$A_2(z) = 1 + \frac{3}{8}z^{-1} + \frac{1}{2}z^{-2}$$

Step 6: Compute k_2

We can read of the last coefficient of $A_2(z)$ which is $k_2 = \frac{1}{2}$

Step 7: Compute $B_2(z)$

Once we have $A_2(z)$, we can compute $B_2(z)$ by simply flipping the coefficients of $A_2(z)$.

$$B_2(z) = \frac{1}{2} + \frac{3}{8}z^{-1} + z^{-2}$$

Step 8: Set $m = 2$

Step 9: Go to step 5

Step 5: Compute $A_1(z)$

We can compute $A_1(z)$ as follows

$$A_1(z) = \frac{1}{1-k_2^2}\left[A_2(z) - k_2 B_2(z)\right]$$

$$A_1(z) = \frac{1}{1-\left(\frac{1}{2}\right)^2}\left[\left(1 + \frac{3}{8}z^{-1} + \frac{1}{2}z^{-2}\right) - \frac{1}{2}\left(\frac{1}{2} + \frac{3}{8}z^{-1} + z^{-2}\right)\right]$$

$$A_1(z) = \frac{4}{3}\left[1 - \frac{1}{2}\cdot\frac{1}{2} + \left(\frac{3}{8} - \frac{1}{2}\cdot\frac{3}{8}\right)z^{-1} + \left(\frac{1}{2} - \frac{1}{2}\right)z^{-2}\right]$$

$$A_1(z) = \frac{4}{3}\left[\frac{3}{4} + \frac{3}{16}z^{-1} + 0z^{-2}\right]$$

$$A_1(z) = 1 + \frac{1}{4}z^{-1}$$

Step 6: Compute k_1

We can read of the last coefficient of $A_1(z)$ which is $k_1 = \frac{1}{4}$

We are done! The result is $k_1 = \frac{1}{4}$, $k_2 = \frac{1}{2}$ and $k_3 = \frac{1}{3}$

The reflection coefficients are $k_1 = \frac{1}{4}, k_2 = \frac{1}{2}, k_3 = \frac{1}{3}$

[✔] ADSI Problem 2.5: All-zero lattice filter, find coefficients from system function (division by zero problem)

A filter has the system function

$$H(z) = 1 + 2z^{-1} + z^{-2}$$

1. Calculate k_1 and k_2 for the corresponding lattice filter and draw the lattice structure.

```
clear variables;
```

We cannot use a recursive algorithm if one of the reflection coefficients is equal to 1. In this problem, $k_2 = 1$ so if we attempt to compute $A_1(z)$, it will lead to division by zero:

$$A_1(z) = \frac{1}{1-k_2^2}[A_2(z) - k_2B_2(z)] = \frac{1}{1-1}[A_2(z) - k_2B_2(z)]$$

We can try out it out in MATLAB function:

```
b = [1, 2, 1];
a = 1;
% tf2latc(b) % This fails because one coeff. is 1
```

Instead we can use a cascade of two single-stage filters.

First, we factorise the original FIR filter to two cascade filters. This can be done by finding the roots and using the formula:

$$H(z) = \left(1 - r_1 z^{-1}\right)\left(1 - r_2 z^{-1}\right) \cdots \left(1 - r_M z^{-1}\right)$$ where r_i are the roots of the transfer function.

Find the roots:

```
roots(b)
```

```
ans = 2×1
    -1
    -1
```

Factorise the original FIR filter into two cascading filters:

$$H(z) = (1 - (-1)z^{-1})(1 - (-1)z^{-1}) = (1 + z^{-1})(1 + z^{-1})$$

Recall that that a single stage all-zero filter has following difference equation:

$$y[n] = x[n] + k_1 x[n-1]$$

Taking the z-transform we get:

$$Y_{\text{az}}(z) = X_{\text{az}}(z) + k_1 X_{\text{az}}(z) z^{-1}$$

The system function of a single stage all-zero lattice filter is therefore:

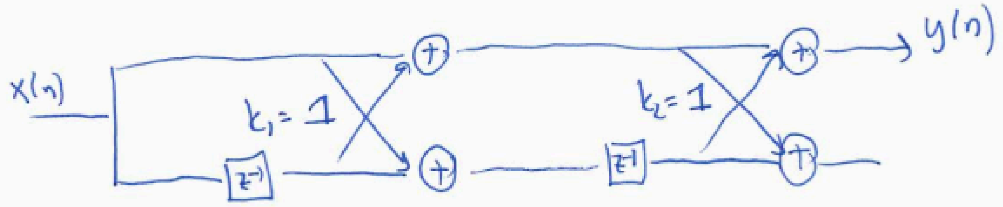
$$H_{\text{az}}(z) = \frac{Y_{\text{az}}(z)}{X_{\text{az}}(z)} = 1 + k_1 z^{-1}$$

It turns out that we can cascade two single stage all-zero lattice filters the same way as we cascade FIR filters.

instead we can factorize $H(z) = A_2(z)$

$H(z) = (1 + z^{-1})(1 + z^{-1})$

so we can instead implement as a cascade of two single stages with $k_1 = 1$
and it turns out that it also works in a single two-stage lattice



So in our case, we have $k_1 = 1$ and $k_2 = 1$.

Let us check if we can get the impulse response of the original filter.

```
k = [1, 1];
lat2fir(k, 1)
```

```
ans = 1×3
     1     2     1
```

```
latc2tf(k) % Same as lat2fir
```

```
ans = 1×3
     1     2     1
```

Success!

[✔] ADSI Problem 2.7: Find system function an all-pole filter from reflection coefficients

Determine the system function for an all-pole filter with the reflection coefficients $k_1 = 0.6$, $k_2 = 0.3$, $k_3 = 0.5$ and $k_4 = 0.9$.

```
clear variables;
```

Use MATLAB to compute the transfer function:

```
k = [0.6, 0.3, 0.5, 0.9];
[num, dem] = latc2tf(k, 'allpole')
```

```
num = 1×5
     1     0     0     0     0
dem = 1×5
    1.0000    1.3800    1.3110    1.3370    0.9000
```

The system function is:

$$H(z) = \frac{1}{1 + 1.38z^{-1} + 1.311z^{-2} + 1.337z^{-3} + 0.9z^{-4}}$$

Alternatively, do it by hand using following two formulas:

$$f_{m-1}[n] = f_m[n] - k_m g_{m-1}[n-1]$$

$$g_m[n] = k_m f_{m-1}[n] + g_{m-1}[n-1]$$

We start computing the difference equations of the last section:

$$f_0[n] = f_1[n] - k_1 g_0[n-1]$$

$$g_1[n] = k_1 f_0[n] + g_0[n-1]$$

Then we compute the next stage:

$f_1[n] = f_2[n] - k_2 g_1[n-1]$

$g_2[n] = k_2 f_1[n] + g_1[n-1]$

Next, we can substitute $f_1[n]$ into the expression $f_0[n]$:

$f_0[n] = f_2[n] - k_2 g_1[n-1] - k_1 g_0[n-1]$
$f_0[n] = f_2[n] - k_2(k_1 f_0[n-1] + g_0[n-2]) - k_1 g_0[n-1]$

And so on.

Finally, we will get a large expression in which we replace:

- $g_0[n]$ and $f_0[n]$ with $y[n]$
- $f_4[n]$ with $x[n]$

[✔] ADSI Problem 2.10: Linear prediction and lattice filters

The idea behind linear prediction is than the signal $\hat{x}(n)$ can be estimated as a weighted linear combination of the p previous samples

$$\hat{x}(n) = -\sum_{k=1}^{p} a_p(k)x(n-k)$$

```
clear variables;
```

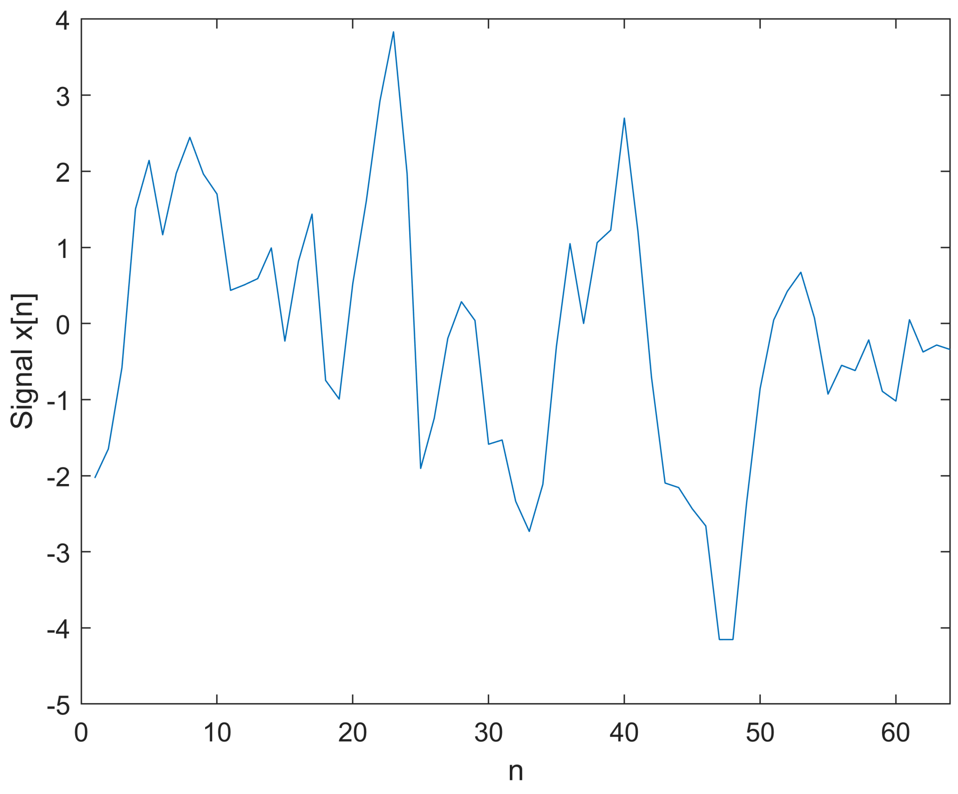
1) Compute autocorrelation from a signal

The file `signal1.dat` contains 64 samples from an artificial signal.

1. Load `signal1.dat` into MATLAB. Create an autocorrelation of the signal using `xcorr` and plot it. What does the autocorrelation tell you about the signal?

Let us first plot the signal.

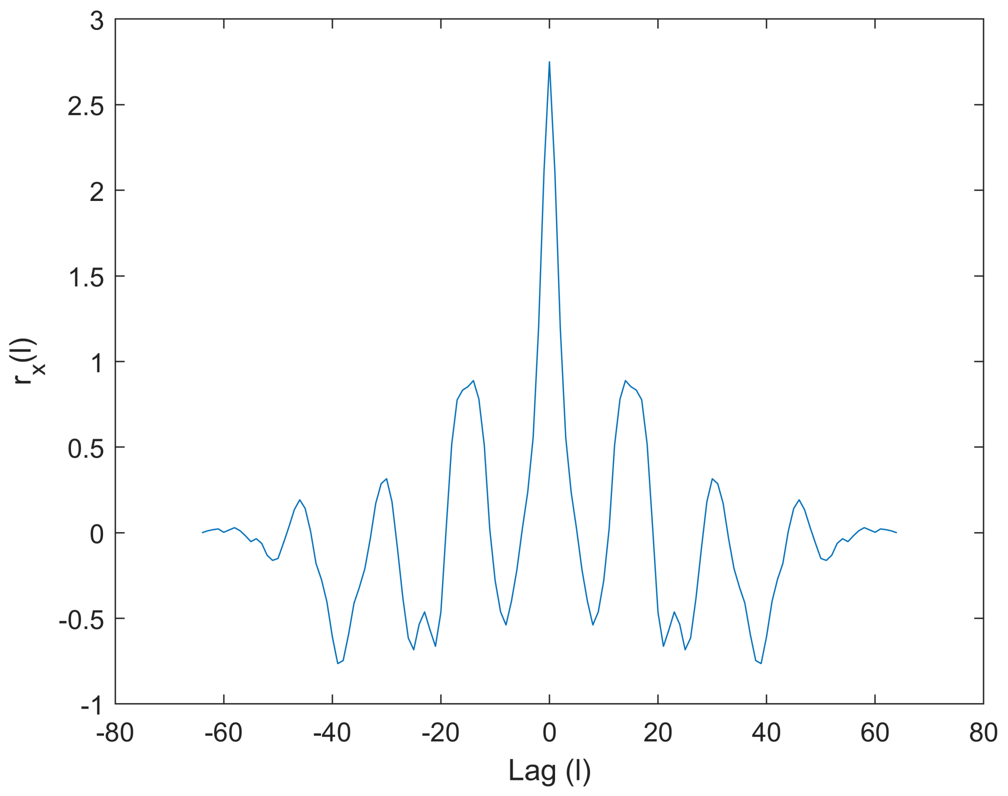
```
x = importdata('signal1.dat');
plot(x);
xlabel('n');
ylabel('Signal x[n]');
xlim([0, numel(x)]);
```



Autocorrelation is the cross-correlation between a signal $x[n]$ with the same signal shifted by some amount $x[n-k]$. So if we start with $k = 1$, we compute the correlation between $x[n]$ and $x[n-1]$. For $k = 2$, we compute the correlation between $x[n]$ and $x[n-2]$ etc.

Autocorrelation is symmetrical around zero.

```
[r,lags] = xcorr(x,64,'biased');
plot(lags,r);
xlabel('Lag (l)');
ylabel('r_x(l)');
```



From the plot, we see that when the signal is shifted e.g. by 4, the correlation is low.

```
r(4)
```

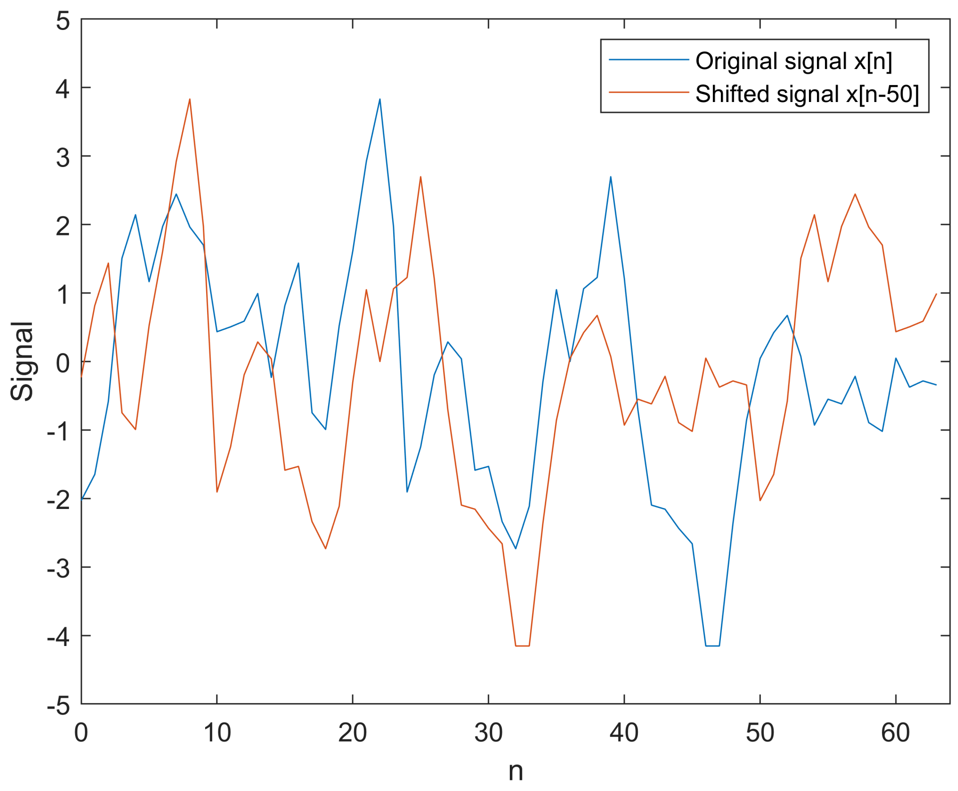
```
ans = 0.0223
```

```
r(50)
```

```
ans = 0.8533
```

This means that the original signal $x[n]$ and the shifted signal $x[n-4]$ are not very similar. The correlation is higher when the signal is shifted by 50:

```
n = linspace(0, 63, 64)';
x_shifted_50 = circshift(x, 50);
plot(n, x, n, x_shifted_50);
legend('Original signal x[n]','Shifted signal x[n-50]')
xlabel('n');
ylabel('Signal');
xlim([0, numel(x)]);
ylim([-5, 5])
```



Shifting the signal by 64 yields the highest correlation. This makes sense because if we shift a signal of length 64 by 64 then we get the original signal $x[n] = x[n-64]$.

```
r(64+1)
```

```
ans = 2.7507
```

2) Compute the coefficients for the optimum 2nd-order linear predictor

Based on the autocorrelation values we can calculate the coefficients for the optimum pth order linear predictor with the normal equations (which we will derive later in the course)

$$r_x(l) = -\sum_{k=1}^{p} a_p(k) r_x(l-k), \qquad l = 1, \dots, p$$

Let $p = 2$.

2. Compute $a_2(1)$ and $a_2(2)$.

When $p = 2$ then we get following two equations:

$r_x(1) = -a_2(1)r_x(0) - a_2(2)r_x(-1)$
$r_x(2) = -a_2(1)r_x(1) - a_2(2)r_x(0)$

In exercise 1) we computed the correlation values r_x. So we know the values for $r_x(-1)$, $r_x(0)$, $r_x(1)$ and $r_x(2)$. This means that we have to solve two equations with two unknowns.

Let us rewrite it:

$$r_x(1) + a_2(1)r_x(0) + a_2(2)r_x(-1) = 0$$
$$r_x(2) + a_2(1)r_x(1) + a_2(2)r_x(0) = 0$$

Solve it in MATLAB:

```
syms a2_1 a2_2

% The autocorrelation is centered around zero
% Find the index where n=0
centerIx = find(lags==0);

% Retrive the values r(-1), r(0), r(1) and r(2) in order
rx_m1 = r(centerIx - 1);
rx_0 = r(centerIx + 0);
rx_1 = r(centerIx + 1);
rx_2 = r(centerIx + 2);

eq1 = rx_1 + a2_1*rx_0 + a2_2*rx_m1;
eq2 = rx_2 + a2_1*rx_1 + a2_2*rx_0;

sol = solve([eq1, eq2], [a2_1, a2_2]);

a1 = sol.a2_1;
a2 = sol.a2_2;
% Print the value of a2(1)
vpa(sol.a2_1)
```

ans = $-1.0554484225972357733576990520746$

```
% Print the value of a2(2)
vpa(sol.a2_2)
```

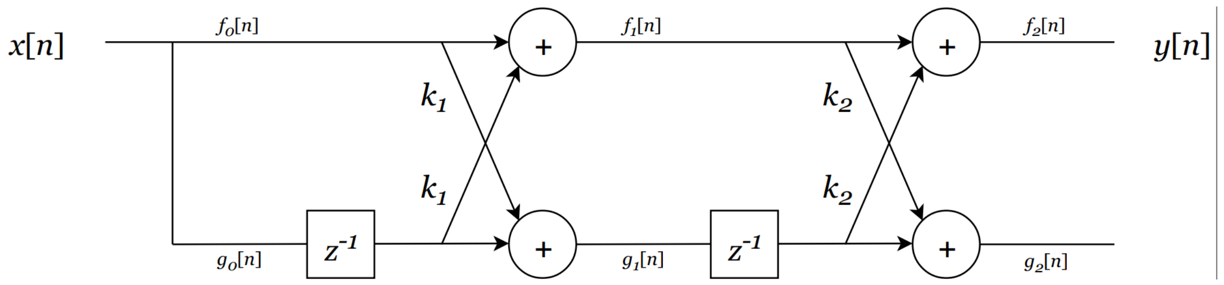
ans = $0.37530590522885965170506927041028$

The coefficents of the predictor are $a_2(1) \approx -1.0554$ and $a_2(2) \approx 0.3753$

3) Compare the linear predictor with a two-stage all-zero lattice filter

Next, consider a two-stage all-zero lattice filter.

3. What is the relation between the output of the filter, $f_2(n)$, and the above linear prediction?



We know that the output of a two-stage all-zero lattice filter is:

$$y[n] = x[n] + k_1(1+k_2)x[n-1] + k_2x[n-2]$$

The p'th order linear predictor has the following form:

$$\hat{x}[n] = \sum_{k=1}^{p} a_k \cdot [n-k]$$

So the second-order linear predictor has the form:

$$\hat{x}[n] = a_1x[n-1] + a_2x[n-2]$$

Notice that 2nd order linear predictor looks very much like the difference equation for the two-stage all-zero lattice filter.

There is a one-to-one correspondance between their coefficients where $a_1 = k_1(1+k_2)$ and $a_2 = k_2$.

4) Compute reflection coefficients from the linear predictor

4. Compute the reflection coefficients k_1 and k_2 from the $a_2(1)$ and $a_2(2)$ values found above.

From 3) we found that $a_1 = k_1(1+k_2)$ and $a_2 = k_2$ so:

$$k_2 = a_2$$

$$k_1 = \frac{a_1}{1+a_2}$$

In 2) we found that values for a_1 and a_2. We will use these to compute k_1 and k_2:

```
k1 = a2/(1+a2);
k2 = a2;
vpa(k1)
```

```
ans = 0.27288903785111455734521239333629
```

```
vpa(k2)
```

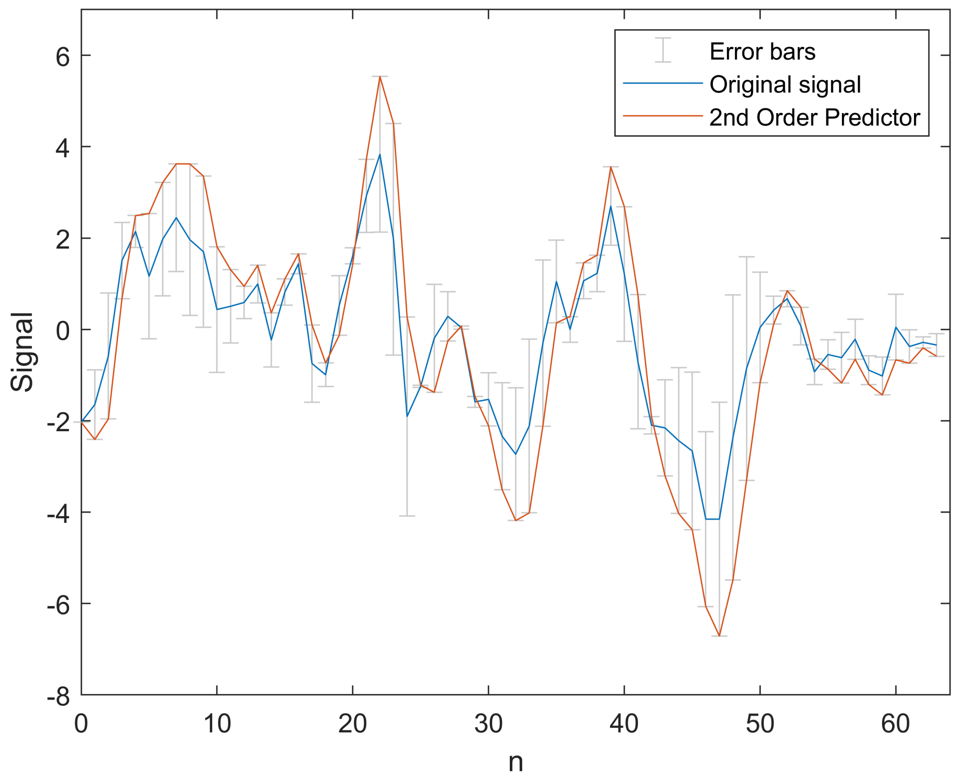
```
ans = 0.37530590522885965170506927041028
```

So we have $k_1 \approx 0.272889$ and $k_2 \approx 0.3753$

5) Compute prediction error

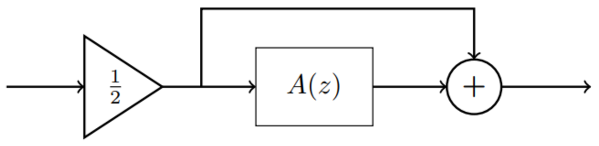
5. Send the signal through a 2nd order all-zero lattice filter and analyze the prediction error. Is the signal predictable? Repeat for a higher order predictor.

```
figure;

% Compute x_hat and err
n = linspace(0, 63, 64);
x = importdata('signal1.dat');
x_hat = azlatfilt([k1, k2], x, 1);
err = x_hat - x;

% Plot
errorbar(n, x, err, 'LineStyle','none', 'Color', [0.8, 0.8, 0.8]);

hold on;
plot(n, x);
plot(n, x_hat);
hold off;

xlabel('n');
ylabel('Signal');
xlim([0, numel(x)]);
ylim([-8, 7])
legend('Error bars', 'Original signal', '2nd Order Predictor');
```



Exam 2016 Problem 3: All-pole Lattice Filter

Consider the digital signal processing system shown below where $A(z)$ is a 2^{nd} order all-pass filter.



1) Explain why a system is a digital notch filter

Since *A*(*z*) is an all-pass filter, all frequencies pass un-attenuated but a frequency dependent phase shift is imposed on the signal.

Assume an input signal $x(n) = \alpha\cos(\omega n)$. If the two signals at the summing point are in phase the output is just equal to the input. For any other phase shift, the two signals will partly cancel out with the degree of cancellation depending on the phase shift.

Since $A(z)$ is an all-pass filter all frequencies pass un-attenuated but a frequency dependent phase shift is imposed on the signal. Assume an input signal $x(n) = \alpha\cos(\omega n)$. If the two signals at the summing point are in phase the output is just equal to the input. For any other phase shift, the two signals will partly cancel out with the degree of cancellation depending on the phase shift. For a phase shift of $(2z+1)\pi$, $z \in \mathbb{Z}$ the two signals will be equal but with opposite sign and the output becomes zero giving the notch filter. Mathematically, this is seen from

$$y(n) = \frac{1}{2}\alpha\cos(\omega n) + \frac{1}{2}\alpha\cos(\omega n + \phi) = \alpha\cos\left(\frac{\phi}{2}\right)\cos\left(\omega n + \frac{\phi}{2}\right).$$

2) Find reflection coefficients for the all-pole lattice filter

The all-pass filter can be implemented as an all-pole lattice filter. Assume that

$$A(z) = \frac{0.3 + 0.91z^{-1} + z^{-2}}{1 + 0.91z^{-1} + 0.3z^{-2}}.$$

2. Find $A_2(z)$, $A_1(z)$ and the reflection coefficients for the all-pole lattice filter.

The recursive algorithm works as follows:

1. Compute $A_M(z) = \dfrac{H(z)}{h[0]}$
2. Compute $k_M = a_M$ where a_M is the last coefficients of $A_M(z)$.
3. Compute $B_M(z)$ by flipping the coefficients of $A_M(z)$
4. Set $m = M$
5. Compute $A_{m-1}(z) = \dfrac{1}{1-k_m^2}\left[A_m(z) - k_m B_m(z)\right]$.
6. Compute $k_{m-1} = a_{m-1}$ where a_{m-1} is the last coefficients of $A_{m-1}(z)$
7. Compute $B_{m-1}(z)$ by flipping the coefficients of $A_{m-1}(z)$. Alternatively, compute $B_{m-1}(z) = z^{-m-1}A_{m-1}\left(\dfrac{1}{z}\right)$
8. Set $m = m - 1$
9. Go to step 5 if $m \neq 0$
10. We know that $A_0(z) = B_0(z) = 1$

the difference equations for an all-pole system and an all-zero system with

$$H_{\text{ap}}(z) = \frac{Y(z)}{X(z)} = \frac{1}{A_2(z)}, \quad H_{\text{az}}(z) = \frac{G_2(z)}{Y(z)} = z^{-2}A_2(1/z), \tag{9.84}$$

where

$$A_2(z) \triangleq 1 + a_1^{(2)} z^{-1} + a_2^{(2)} z^{-2}. \tag{9.85}$$

From the system function of the all-pass filter, we know that:

$A_2(z) = 1 + 0.91z^{-1} + 0.3z^{-2}$ and $k_2 = a_2^{(2)} = 0.3$

We compute B_2:

$$B_2 = 0.3 + 0.91z^{-1} + z^{-2}$$

We compute A_1:

$$A_1(z) = \frac{1}{1-k_2^2}[A_2(z) - k_2B_2(z)]$$

$$A_1(z) = \frac{1}{1-0.3^2}[(1 + 0.91z^{-1} + 0.3z^{-2}) - 0.3(0.3 + 0.91z^{-1} + z^{-2})]$$

$$A_1(z) = \frac{1}{1-0.3^2}[1 + 0.91z^{-1} + 0.3z^{-2} - 0.09 - 0.2730z^{-1} - 0.3z^{-2}]$$

$$A_1(z) = 1.0989[0.91 + 0.6370z^{-1}]$$

$$A_1(z) = 1 + 0.7z^{-1}$$

with $k_1 = a_1^{(1)} = 0.7$

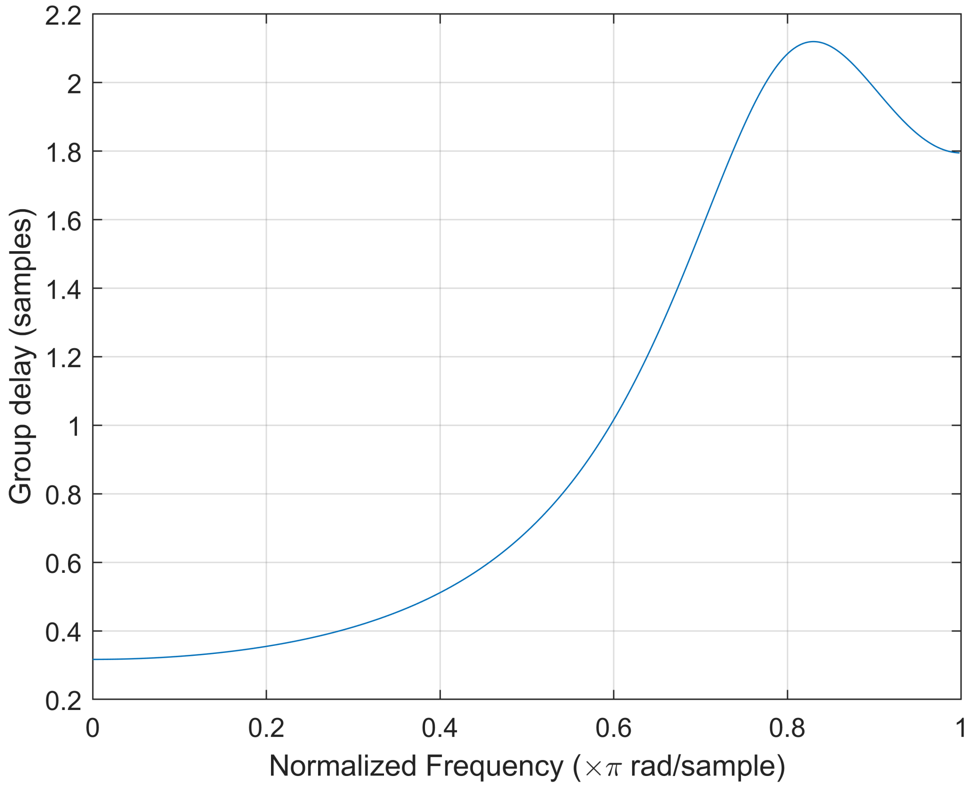
3) Determine the group delay in the digital notch filter

The transfer function of the entire system is needed to determine the group delay

$$\begin{aligned} H(z) &= \frac{1}{2}\left(1 + \frac{0.3 + 0.91z^{-1} + z^{-2}}{1 + 0.91z^{-1} + 0.3z^{-2}}\right) \\ &= \frac{1}{2}\left(\frac{1 + 0.91z^{-1} + 0.3z^{-2}}{1 + 0.91z^{-1} + 0.3z^{-2}} + \frac{0.3 + 0.91z^{-1} + z^{-2}}{1 + 0.91z^{-1} + 0.3z^{-2}}\right) \\ &= \frac{0.65 + 0.91z^{-1} + 0.65z^{-2}}{1 + 0.91z^{-1} + 0.3z^{-2}}. \end{aligned}$$

The group delay is then found using Matlab:

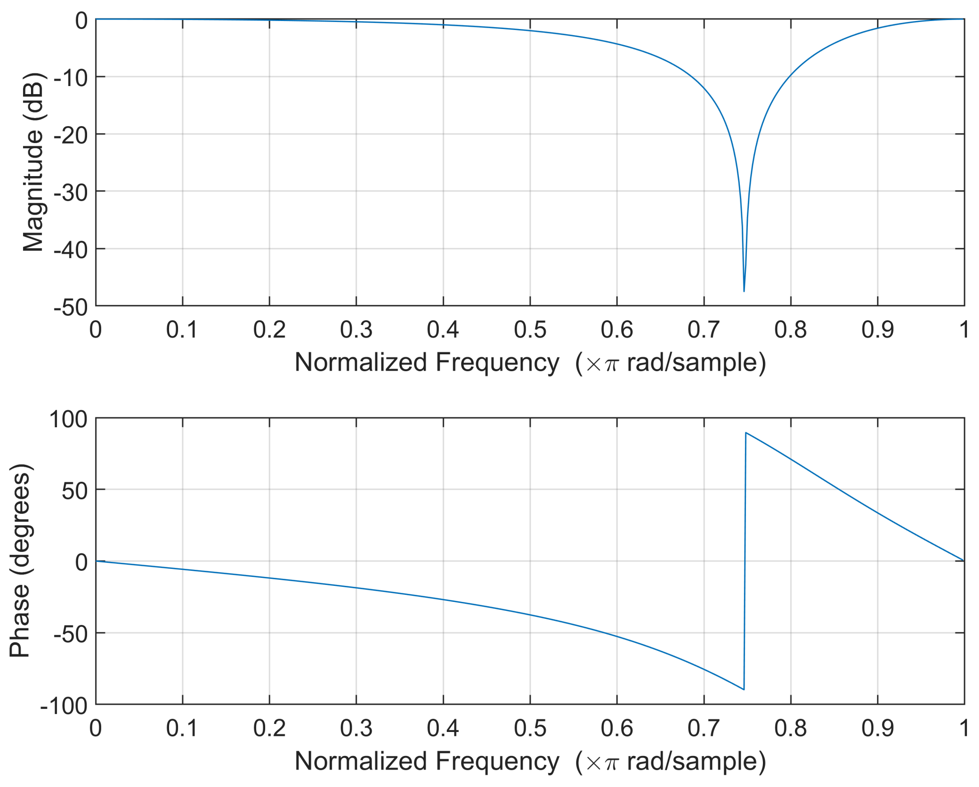
```
grpdelay([0.65 0.91 0.65],[1 0.91 0.3])
```



From which it is seen that the group delay at $\omega = \frac{\pi}{4}$ is 0.3788.

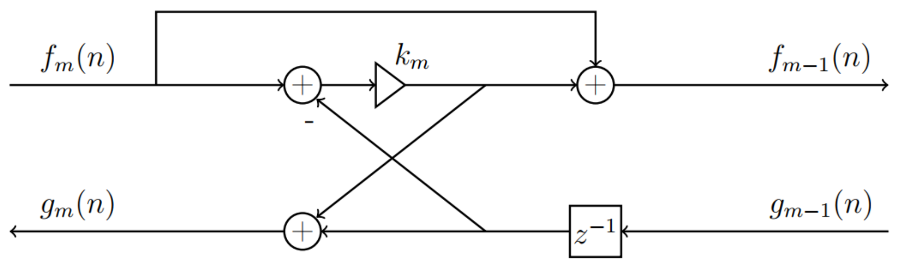
For completeness, the performance of the notch �lter can be plotted with freqz.

```
freqz([0.65 0.91 0.65],[1 0.91 0.3])
```



Exam 2017 Problem 2: Lattice Structures

Many different lattice structure have been investigated over the years, often tailored to specific requirements. One particular example is based on minimizing hardware requirements and uses only one multiplication per stage. The structure of the *m*'th stage of a one-multiplier lattice filter is shown below. The input and output is connected as for standard all-pole filters.



```
clear variables;
```

1) Determine the two equations relating outputs to inputs

1. Determine the two equations relating outputs $f_{m-1}(n)$ and $g_m(n)$ to inputs $f_m(n)$ and $g_{m-1}(n)$.

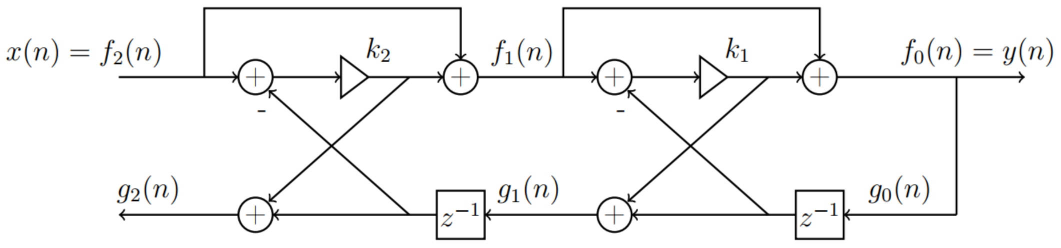
$$\begin{aligned} f_{m-1}(n) &= f_m(n) + k_m(f_m(n) - g_{m-1}(n)) \\ &= f_m(n) + k_m f_m(n) - k_m g_{m-1}(n-1) \\ &= (1+k_m) f_m(n) - k_m g_{m-1}(n-1) \end{aligned}$$

$$\begin{aligned} g_m(n) &= g_{m-1}(n-1) + k_m(f_m(n) - g_{m-1}(n)) \\ &= g_{m-1}(n-1) + k_m f_m(n) - k_m g_{m-1}(n) \\ &= k_m f_m(n) + (1-k_m) g_{m-1}(n-1) \end{aligned}$$

2) When is the first non-zero sample observed at the output?

2. If the input of a two-stage, one-multiplier lattice filter with reflection coefficients k_1 and k_2 is excited by an impulse, $x(n) = \delta(n)$, when is the first non-zero sample observed at the output $y(n)$?

The two-stage, one multiplier lattice looks like this:



We found that the equation relating output to input is:

$$f_{m-1}(n) = (1+k_m) f_m(n) - k_m g_{m-1}(n-1)$$

Since we want the first output, we do not concern ourselves with the $g_{m-1}(n-1)$.

First, we compute $f_1(n)$:

$$f_1(n) = (1+k_2) f_2(n) = (1+k_2) x(n)$$

Next, we compute $f_0(n)$:

$$f_0(n) = (1+k_1) f_1(n) = (1+k_1)(1+k_2) x(n) = y(n)$$

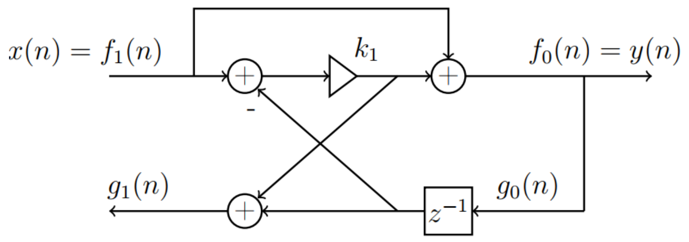
Since there is a direct path without any delays from input $x(n)$ to the output $y(n)$, the first non-zero sample appears instantaneously at the output.

$$y(n) = (1+k_1)(1+k_2) x(n)$$

3) Calculate the transfer function for a one-stage lattice filter

3. Calculate the transfer function $H_{G_1}(z) = G_1(z)/X(z)$ for a one-stage, one-multiplier, lattice filter with reflection coefficients k_1 and show that it corresponds to an all-pass filter.

The one-stage, one multiplier lattice looks like this.



The equations relating the inputs and outputs are:

$$f_0(n) = (1 + k_1)f_1(n) - k_1 g_0(n-1)$$

$$g_1(n) = k_1 f_1(n) + (1 - k_1)g_0(n-1)$$

To calculate transfer functions, the two equations are first *z*-transformed:

$$F_0(z) = (1 + k_1)F_1(z) - k_1 z^{-1} G_0(z) \quad \text{Eq. (1)}$$

$$G_1(z) = k_1 F_1(z) + (1 - k_1)z^{-1} G_0(z) \quad \text{Eq. (2)}$$

We know that $F_1(z) = X(z)$ and $F_0(z) = G_0(z) = Y(z)$.

We use the identity $F_0(z) = G_0(z)$ to rewrite our Eq. (1):

$$G_0(z) = (1 + k_1)F_1(z) - k_1 z^{-1} G_0(z)$$

$$G_0(z) + k_1 z^{-1} G_0(z) = (1 + k_1)F_1(z)$$

$$\left(1 + k_1 z^{-1}\right)G_0(z) = (1 + k_1)F_1(z)$$

$$G_0(z) = \frac{(1 + k_1)F_1(z)}{1 + k_1 z^{-1}}$$

Substitutde the new expression into Eq. (2)

$$G_1(z) = k_1 F_1(z) + \frac{(1 - k_1)(1 + k_1)z^{-1}}{1 + k_1 z^{-1}} F_1(z)$$

We know that $F_1(z) = X(z)$:

$$G_1(z) = k_1 X(z) + \frac{(1 - k_1)(1 + k_1)z^{-1}}{1 + k_1 z^{-1}} X(z)$$

$$G_1(z) = X(z)\left(k_1 + \frac{(1-k_1)(1+k_1)z^{-1}}{1+k_1z^{-1}}\right)$$

$$\frac{G_1(z)}{X(z)} = k_1 + \frac{(1-k_1)(1+k_1)z^{-1}}{1+k_1z^{-1}}$$

$$\frac{G_1(z)}{X(z)} = \frac{k_1\left(1+k_1z^{-1}\right)}{1+k_1z^{-1}} + \frac{(1-k_1)(1+k_1)z^{-1}}{1+k_1z^{-1}}$$

$$\frac{G_1(z)}{X(z)} = \frac{k_1\left(1+k_1z^{-1}\right)+(1-k_1)(1+k_1)z^{-1}}{1+k_1z^{-1}}$$

```
syms k z
expand(k*(1+k*z)+(1-k)*(1+k)*z)
```

ans = $k + z$

$$\frac{G_1(z)}{X(z)} = \frac{k_1+z^{-1}}{1+k_1z^{-1}}$$

An allpass filter has the form:

$$H_k(z) = z^{-1}\frac{1-p_k^*z}{1-p_kz^{-1}} = \frac{z^{-1}-p_k^*}{1-p_kz^{-1}} \qquad (5.157)$$

Higher order allpass systems can be obtained by cascading multiple first-order sections, as

$$H_{\text{ap}}(z) = e^{j\beta}\prod_{k=1}^{N}\frac{z^{-1}-p_k^*}{1-p_kz^{-1}}, \qquad (5.158)$$

Exam 2014 Problem 1

Functions

```
function [k,G] = fir2lat(h)
% Converts FIR filter coefficients to lattice coefficients.
% From Figure 9.24 (p. 514)
    G = h(1);
    a = h/G;
    M = length(h)-1;
    k(M) = a(M+1);
    for m = M:-1:2
        b = fliplr(a);
        a = (a-k(m)*b)/(1-k(m)^2); a = a(1:m);
        k(m-1) = a(m);
    end
end


function [h] = lat2fir(k, G)
% Converts lattice coefficients to FIR filter coefficients.
% From Figure 9.25 (p. 515)
    a = 1;
    b = 1;
    M = length(k);
    for m = 1:1:M
        a = [a,0]+k(m)*[0,b];
        b = fliplr(a);
    end
    h = G*a;
end


function [y] = azlatfilt(k, x, G)
% AZLATFILT      Filters x with an all-zero lattice filter with
%                coefficients k. From Figure 9.26 (p 515)

    M = length(k);

    % Create an array for the sequence f_m[n]
    f = zeros(1,M);

    % Create an array for the sequence g_m[n]
    g = f;

    % Create an array for the sequence g_m[n-1]
    oldg = zeros(1,M);

    % Keeps track of x[n-1]
    oldx = 0;
    x = G*x;
    y = zeros(size(x));

    for n=1:length(x)
        % Compute f1[n] = x[n] + k1x[n-1] -> Equation (9.57a)
        f(1) = x(n)+k(1)*oldx;

        % Compute g1[n] = k1x[n] + x[n-1] -> Equation (9.57b)
        g(1) = k(1)*x(n)+oldx;
        oldx = x(n);
        for m = 2:M
            % Compute: fm[n] = fm?1[n] + km gm?1[n ? 1] -> Equation (9.55a)
            f(m) = f(m-1)+k(m)*oldg(m-1);

            % Compute: gm[n] = km fm?1[n] + gm?1[n ? 1] -> Equation (9.55b)
            g(m) = k(m)*f(m-1)+oldg(m-1);

            % Delay
            oldg(m-1) = g(m-1);
        end

        % y[n] = fM[n] -> Equation (9.56b)
        y(n) = f(M);
    end
end

function y = aplatfilt(k,G,x)
% All-pole Lattice Filter Implementation
% From Figure 9.28

    M = length(k); g = zeros(1,M); f = g;
    oldy = 0; x = G*x; y = zeros(size(x));
    for n = 1:length(x)
        f(M) = x(n);
        for i = 1:M-1
            m = M+1-i;
            f(m-1) = f(m)-k(m)*g(m-1);
            g(m) = k(m)*f(m-1)+g(m-1);
        end
        y(n) = f(1)-k(1)*oldy;
        g(1) = k(1)*y(n)+oldy;
        oldy = y(n);
    end
end
```