The Open University of Sri Lanka

ECD 2216 Digital Computer Simulation

Saturday 18th March 2006, Time: 0930 hrs. – 1230 hrs.

045

Three hours

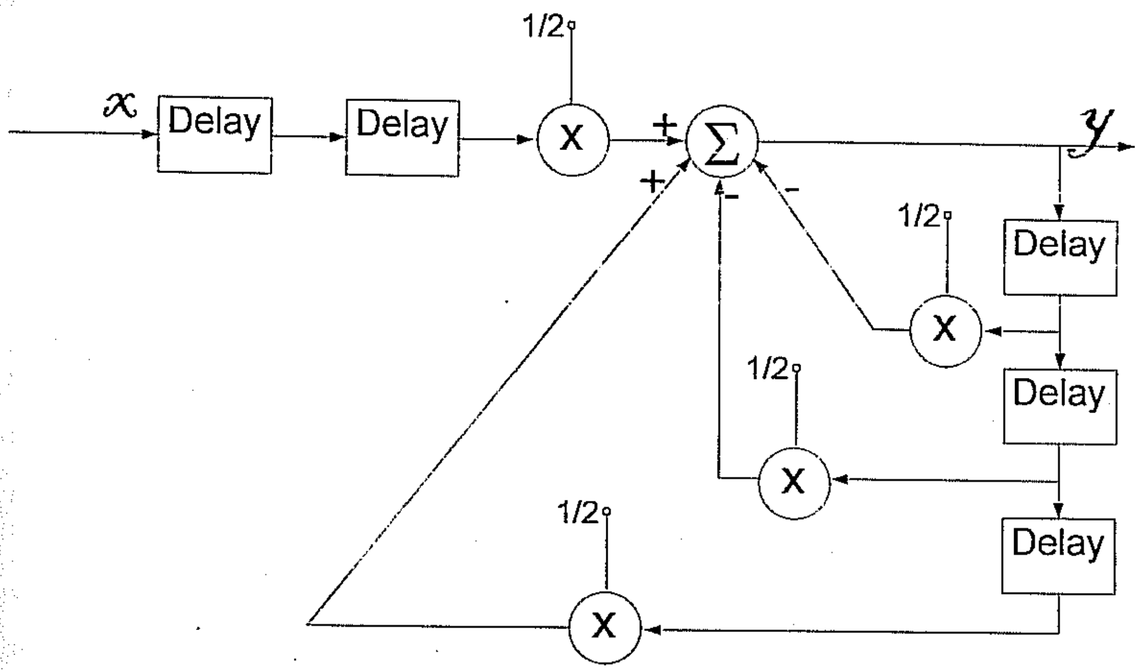
Up to five questions may be attempted, selecting at least **two** questions from each section. However, full credit may be obtained for exceptionally good answers to only **four** questions. All questions carry equal marks.

Section A

1. Examine the roots of the following transfer functions and hence determine whether the filters represented by them are stable:

   i. $1 / (9z^2 - 1)$
   ii. $z^2 / (z^2 + 2z + 1)$
   iii. $z^2 / (z^2 - z - 2)$

2. Examine the digital filter shown in the figure.



   a. Does it represent a recursive filter or a non-recursive filter? Give reasons for your answer.
   b. Write down the z transform representation of the filter.
   c. Determine whether it is stable.

3. A non-recursive digital filter has an impulse response {1,2,1}, and is excited by an input sequence {1,-1,1}. Use the Discrete Fourier Transform to obtain the output sequence of the filter. How would you modify these sequences if you were to use the Fast Fourier Transform?

4. State the Sampling Theorem. What is meant by "aliasing" or "frequency folding"? Explain, using frequency density spectra where necessary, how a low pass filter may be used to recover a continuous signal from a properly sampled signal.

Section B

Read the note on "Approximating analog filters" reproduced at the end of this paper before you answer questions 5 to 8. Unfortunately, you will not be able to view the demos referred to in the text.

5. The paper describes three techniques for obtaining a discrete filter from an analog filter. Which of them generates a FIR filter? Give reasons for your answer.

6. "Analog filters are usually described by either their impulse response or their differential equation. Digital filters are typically described by either their impulse response or their difference equation."

   Which of the three techniques described in the note attempts to obtain a digital filter by matching of impulse responses, and which attempts to do so by matching the differential equation to a corresponding difference equation?

7. The mappings of the s plane to the z plane in the three methods are given by:

   a. $z=e^{s/fs}$ or $s=fs*\ln(z)$
   b. $s=(1-z^{-1})*fs$
   c. $s=2fs(1-z^{-1})/(1+z^{-1})$

   State the locus on the z plane, of poles on the imaginary axis of the s plane, for each of these transformations.

8. Sketch three simple flow diagrams to illustrate the recommended procedure for obtaining a digital filter, in the three techniques.

Approximating Analog Filters

There are many ways to convert analog filters such as Butterworth, Chebychev, and Elliptic into digital filters. None of the conversion methods are optimal, none of the methods are perfect, none of the methods are optimal, and none of the methods produce identical results. The three major techniques are:

- Impulse Invariance - Windowing Method
- Derivative Approximation
- Integral Approximation - Bilinear Z transform

Impulse Invariance - Windowing Method

This analog-filter to digital-filter conversion process is based on a simple and intuitively pleasing idea:

If the impulse response of the digital filter, h[n], looks like a sampled version of the impulse response of the analog filter, h(t), then the two filters should do equivalent things to the input signal.

While this statement is approximately true, it is not exactly correct. There are some problems with this approach that limit where it is used. To see these problems, go back and review the material on FIR filter design using weighting windows. When using this technique to design filters we:

1. Find an equation for the impulse response of a good, or even optimal, analog filter
2. Sample the impulse response at fs samples/second
3. Truncate the impulse response to make it finite length
4. Multiply by some smooth weighting window
5. Fourier Transform the impulse response to see how close it came to ideal
6. Modify steps 3 and/or 4 to get a better impulse response if needed.

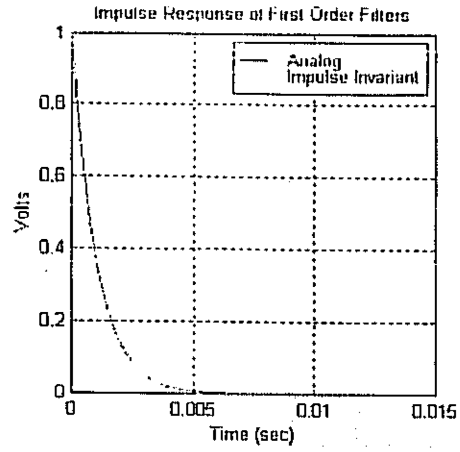
The impulse response of most ideal filters (ideal low pass, ideal bandpass, ideal notch), and most standard filters (Butterworth, Chebychev, Elliptic) extend to t=+infinity, and some times to t=-infinity. If we use a FIR digital filter, we will have to truncate the signal, and this will lead to distortion. This distortion rounded off sharp transitions and caused side lobes. Weighting windows let us trade-off these two types of distortion.

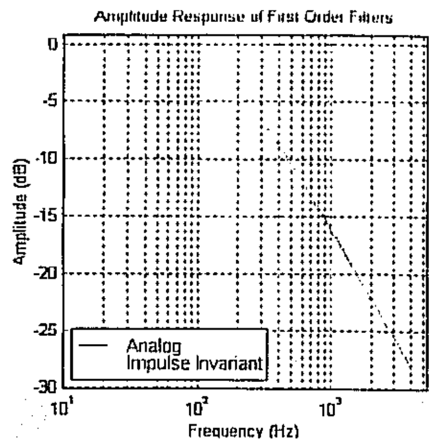
Sometimes, we can design IIR filters using this technique. A simple example is a simple R/C low pass filter. This filter has an impulse response that is a decaying exponential. We can generate samples of this impulse response by using a first order IIR digital filter. There is no distortion due to windowing, because the impulse response of the digital filter goes out to n=infinity, just as the impulse response of the analog filter went out to t=infinity. In many cases it will be very difficult or impossible to find the correct feedback coefficients to make this work.

Even if we get lucky, and find the right feedback coefficients to avoid windowing problems, these digital filters still distort the signal. The problem is with the sampling rate of the impulse response. The sampling frequency of the impulse response, h(t), matches the sampling frequency of the input signal, x(t). We always assume x(t) has no power above fh Hz, and we set the sampling rate to be at least 2fh. For the digital filter to be ideal, we must make sure that fs is also greater than two times the highest frequency in h(t). Unfortunately, all of the standard analog filters have H(f) that extend to f=+/-infinity, so we will always have aliasing problems when sampling h(t).

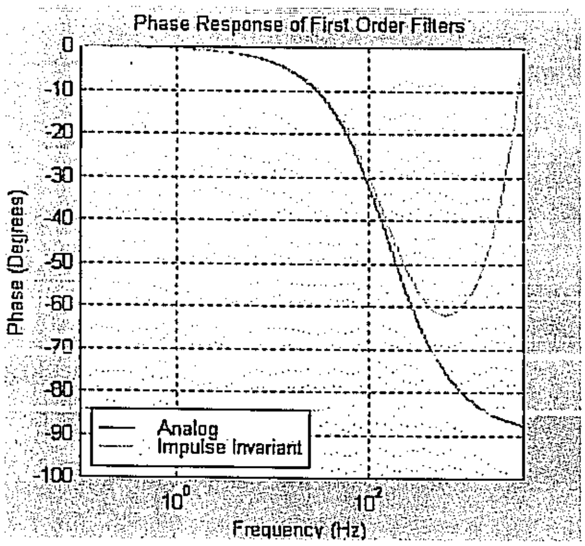
To see how this technique works for an R/C lowpass filter, run script demo0807.m. As promised, the digital impulse response exactly matches the analog impulse response.

At low frequencies, these two filters look identical, but aliasing becomes a problem close to the Nyquist frequency. It is common for these filters to pull away from the ideal response in this way





The phase response shows even a more dramatic deviation from the ideal response at high frequencies. The phase must get back to zero degrees at the Nyquist frequency, since the spectrum must be conjugate symmetric about both DC and fs/2.



Many times we would like to see how a mapping like impulse invariance moves poles/zeros from the S plane to the Z plane. From the discussion on Z transforms, we see that the Laplace transom integrates $x(t)e^{-st}$ while the Z transform sums $x[n]z^{-n}$. The two transforms look similar if we use the substitution $z=e^{s/fs}$ or $s=fs*\ln(z)$. The script demo0806.m allows the user to add poles and zeros to the S plane and see how they are mapped to the Z plane. Notice that Stable analog filters (poles in left half plane) will be mapped to stable digital filters (poles inside unit circle).

Poles close to the imaginary axis in the S plane at frequency f1 (filters with high gain at f1), will be mapped to poles close to the unit circle in the Z plane at frequency f1 (filters with high gain at f1). The rule given above only applies when f1 is between DC and fs/2. For higher frequencies aliasing occurs (the pole shows up in the Z plane at frequency f1-fs, or f1-2fs, or f1-3fs, or...)

Derivative Approximation

Analog filters are usually described by either their impulse response or their differential equation. Digital filters are typically described by either their impulse response or their difference equation. The impulse invariant design tried to translate analog filters to digital filters by matching impulse responses. The derivative approximation tries to translate the filters by matching the differential equation to the difference equation.

Earlier we saw how the derivative could be approximated by a finite difference: d/dt x(t) is approximately equal to the finite difference (x[n]-x[n-1])*fs = rise/run. The Laplace transform of d/dt of x(t) is sX(s). The Z transform of (x[n]-x[n-1])*fs is $(1-z^{-1})*fs*X(z)$. From here we can see that one way to approximate a filter is to:

Find impulse response, h(t), for analog filter
Find transfer function, H(s), for analog filter
Use the substitution $s=(1-z^{-1})*fs$ to find H(z)
Build a digital filter with the specified H(z)

This technique will usually generate a different filter than the impulse invariant design. Script demo0803.m shows how the impulse invariant and d/dt approximation filter compare. Both do a good job at low frequencies, but deviate from ideal near the Nyquist rate.

The impulse invariant design suffers from aliasing, but the d/dt approximation design has a different type of distortion. To see the effects of this distortion, look at the mapping of poles and zeros from the s plane to the z plane. Script demo0806.m may help you visualize this transformation. Notice the following:

- Poles on the imaginary axis of the S plane get mapped to a circle of radius 1/2 centered at 1/2.
- As analog frequency ranges from DC to infinity, the points in the Z plane move from 1 to 0 - there is no aliasing.
- Poles near s=0 get mapped to poles near z=0, at the correct frequencies. Poles far from s=0 all get moved to the wrong frequencies and all end up near z=0.
- Stable analog filters generate stable digital filters.
- Some unstable analog filters generate stable digital filters

Derivative approximation only works well when the finite difference is very close to the true derivative. This will only be true when the sampling rate is 10 to 100 times as high as required by the Nyquist theorem.

Integral Approximation / Bilinear Z Transform

Since the derivative approximation is such a lousy way to translate analog filters to digital filters, people kept looking for better ways. A better method was discovered simultaneously by Dr.s Biliano and Chzefnear, and is now known as the Biliano-Chzefnear, or Bilinear, transform. This technique is very similar to the d/dt approximation, only now we focus on estimating the integration operation. Recall that the LT of the integral of x(t) is X(s)/s. A digital integrator that performs trapezoidal integration keeps a running total of the input signal, so it has the transfer function y[n]=y[n-1]+x[n]/fs. This leads to the transfer function $Y(z)=z^{-1}Y(z)+X(z)/fs$, or $H(z)=1/fs(1-z^{-1})$. This leads us to the translation of $s=fs(1-z^{-1})$. But there is a better way to perform integration - namely trapezoidal integration. In this technique y[n]=y[n-1]+(x[n]+x[n-1])/2fs, so $H(z)=(1+z^{-1})/\ 2fs(1-z^{-1})$, leading to the transformation $s=2fs(1-z^{-1})/(1+z^{-1})$. Although this is very similar to the d/dt approximation, it produces much better digital filters. When you ask a computer tool like Matlab to design a filter, it uses this technique. Script demo0803.m shows how the impulse invariant, d/dt approximation and Bilinear transform compare for a first order low pass filter. Notice that unlike all other filters, the Bilinear filter gain goes to 0 (- infinite dB) at the Nyquist frequency. You can see why this happens by looking at the mapping of poles and zeros from the S plane to the Z plane in script demo0806.m. This transformation is a mix of the previous two.

Similar to the impulse invariant design, the imaginary axis in the S plane gets mapped to the unit circle in the Z plane.

- Similar to the impulse invariant design, stable analog filters are mapped to stable digital filters, and unstable analog filters are mapped to unstable digital filters.
- Similar to the d/dt approximation - there is frequency warping, rather than frequency aliasing. All analog frequencies from DC to infinity are mapped to the range DC to fs/2 in the digital filter. High analog frequencies never 'alias' and show up as low digital frequencies. The mapping of analog to digital frequencies follows an arctangent curve. For zonal filters, people usually prefer frequency warping to aliasing.