FILTERS AND FREQUENCY RESPONSES 5

Electrical Engineering 20N
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley

HSIN-I LIU, JONATHAN KOTKER, HOWARD LEI, AND BABAK AYAZIFAR

1 Introduction

In this lab session, we will use LabVIEW to help us explore the functioning of various discrete-time filters. Filters are ubiquitous in various disciplines, and are used to remove or to amplify different aspects of either a signal or a medium, such as water. While water filtration removes dirt and other particles, a filter in electrical engineering may remove noise or boost a certain frequency, as is done by a bass boost on a CD player. In electrical engineering, filters are represented by complex functions, whose magnitude and phase show the differences between the original signal and the filtered signal. Using these complex functions, we can analyze the frequency responses of the corresponding filters; from these frequency responses, we can extrapolate the behavior of these filters. Finally, we will analyze the performance of the discrete-time filters in the time domain, using dataflow programming.

1.1 Lab Goals

- Learn how to plot the magnitude and the phase of the frequency responses of different filters.
- Use LabVIEW to verify magnitude and phase response plots of filters.
- Use LabVIEW to emulate discrete-time filters and verify their properties.

1.2 Checkoff Points

2. In-Lab Section .....
    1. With Great Power Comes Great Frequency Response .....
    2. Discrete-Time Filters, Reloaded .....
    3. Filter Represent!: LCCDEs and DAG Block Diagrams .....
    4. Frequency Domain Analysis .....(15%)
    5. Radians Per ... Sample? .....
    6. Variations on a Discrete-Time Filter .....(15%)
    7. Time Domain Analysis .....(20%)

2 In-Lab Section

2.1 With Great Power Comes Great Frequency Response

In lab 04, we explored the concept of the impulse response of a discrete-time LTI system, which was defined to be the response of the system to the Kronecker delta signal, $\delta(n)$. We then deduced that if an LTI system H has an impulse response $h(n)$, then the input signal $x(n)$ and the output signal $y(n)$ are related by

$$y(n) = (x * h)(n) = \sum_{k=-\infty}^{\infty} x(k)h(n-k) = \sum_{k=-\infty}^{\infty} h(k)x(n-k).$$

The impulse response can thus be used to completely specify and characterize a system. In other words, the impulse response encapsulates everything that we need to know about a system and its responses to various inputs.

Since the signal $x(n) = e^{i\omega_0 n}$ has only one frequency (at ω_0), we can determine how a discrete-time LTI system acts on signals containing different frequencies by determining its response to the complex exponential signals at those frequencies. We thus consider another, special kind of response: the response of the discrete-time LTI system H to the input signal $x(n) = e^{i\omega n}$ at any arbitrary frequency ω. From the discussion above, we conclude that

$$\begin{aligned} y(n) &= \sum_{k=-\infty}^{\infty} h(k)x(n-k) \\ &= \sum_{k=-\infty}^{\infty} h(k)e^{i\omega(n-k)} \\ &= \sum_{k=-\infty}^{\infty} h(k)e^{i\omega n} \cdot e^{i\omega(-k)} \\ &= e^{i\omega n} \sum_{k=-\infty}^{\infty} h(k)e^{-i\omega k} \\ &= H(\omega)e^{i\omega n}, \end{aligned}$$

where

$$H(\omega) = \sum_{n=-\infty}^{\infty} h(n)e^{-i\omega n}.$$

H is a function of ω and is known as the **frequency response**. In order to plot the frequency response

FREQUENCY RESPONSE

against ω, we would need three axes (why?). Nonetheless, we recall that any complex number z can be written in the form

$$z = |z|e^{i\angle z},$$

where $|z|$ is the **magnitude** of z, a nonnegative real number, and $\angle z$ is the **phase** of z, a real number. Since H is a complex-valued function of ω, we can accomplish something similar. At any value of ω, MAGNITUDE PHASE

$$H(\omega) = |H(\omega)|e^{i\angle H(\omega)}.$$

We can then examine the behavior of $H(\omega)$ merely by examining the behavior of its components, $|H(\omega)|$ and $\angle H(\omega)$. Notice that $|H(\omega)|$ and $\angle H(\omega)$ are also functions of ω, and since these functions are real-valued, we can plot them against ω on a two-dimensional surface.

2.2 Discrete-Time Filters, Reloaded

In the post-lab of lab 04, you analyzed the input-output relationships of three discrete-time filters: the two-point moving average filter, the two-point moving difference filter, and a three-point filter that was simply the cascade of the moving average and moving difference filters. Recall that you observed the relationships between the inputs and outputs of each filter, and saw that the two-point moving average filter was a low-pass filter, the two-point moving difference filter was a high-pass filter, and the three-point filter was a mid-pass, or band-pass, filter. In this lab, you will attempt to more fully understand why the filters have these effects on input signals at certain frequencies by analyzing them in the frequency domain, via their frequency responses. You will also experiment with variations on these basic filters by incorporating feedback terms in the implementation of these filters.

2.3 Filter Represent!: LCCDEs and DAG Block Diagrams

Each filter that we explore in this class can be characterized by a *linear* equation that shows the relationships between consecutive values of the input and output signals. The scaling *coefficients* of the terms in the linear equation are *constant*. Stringing these properties together, we call such equations **linear constant-coefficient difference equations (LCCDEs)**. For the low-pass filter $\mathsf{H_L}$, for example, we have the LCCDE LCCDE

$$\forall n \in \mathbb{Z}, \quad y(n) = \frac{1}{2}\left(x(n) + x(n-1)\right).$$

The corresponding LCCDE of the high-pass filter $\mathsf{H_H}$ is

$$\forall n \in \mathbb{Z}, \quad y(n) = \frac{1}{2}\left(x(n) - x(n-1)\right).$$

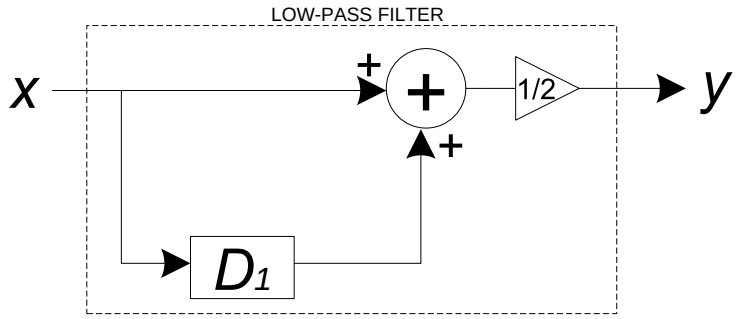
Finally, for the mid-pass, or band-pass filter $\mathsf{H_M}$, we have the LCCDE

$$\forall n \in \mathbb{Z}, \quad y(n) = \frac{1}{2}\left(x(n) - x(n-2)\right).$$

LCCDEs thus serve as one way to describe a filter. Another convenient way to describe a filter is to draw its **delay-adder-gain (DAG) block diagram**, such as the one for the low-pass filter as shown in Figure 1. DAG BLOCK DIAGRAM

Notice that the diagram shows the input signal x entering the filter and getting manipulated to produce the output signal y leaving the filter. This manipulation uses *delay elements*, D_N, which delay their inputs by N samples. For instance, the DAG block diagram of the low-pass filter uses a delay element that delays its input by 1 sample. We also use *adder elements* that add their inputs and *gain elements* that scale their inputs. Can you construct similar DAG block diagrams for the other filters?

Figure 1 Delay-Adder-Gain Block Diagram of the Low-Pass Filter



2.4 Frequency Domain Analysis

In this lab, we will use the LCCDEs to derive and implement the frequency responses $H_L(\omega)$, $H_H(\omega)$, and $H_M(\omega)$ for the low-pass, high-pass, and mid-pass filters respectively. To do so, we need to begin with the impulse responses $h_L(n)$, $h_H(n)$, and $h_M(n)$ for the three filters. If you do not recall what the impulse responses are, you can derive them by matching the LCCDEs with the convolution sum.

1. On a separate sheet of paper, determine expressions for frequency responses $H_L(\omega)$, $H_H(\omega)$, and $H_M(\omega)$ for the low-pass, high-pass, and mid-pass filters, respectively. Each should be a function of the discrete-time frequency variable ω, which has units of radians per sample. Using these expressions, extract the magnitudes and phases of the frequency responses.

   For each filter, your magnitude response expression should involve no more than the absolute value of one trigonometric function, with no additional terms. Your phase response expressions should involve no more than a linear function of ω. However, you should be careful of phase-shifts, where your linear phase functions may shift by π at certain values of ω. This is usually caused by sign-changes in the trigonometric expressions for the magnitudes.

2.5 Radians Per ... Sample?

Notice that the unit of discrete-time frequency ω is radians per sample. Remember that in discrete-time, we concern ourselves with *samples* (n), not *seconds* (t), where a sample is defined to be the value of a signal at a certain point in time, and samples are numbered with the integers.

The unit of radians per sample denotes the rate at which successive samples of a discrete-time signal advance through a period of 2π radians. For instance, if ω were 2π radians per sample, such as with the signal $e^{i2\pi n} = 1$, then each sample advances through a whole period by itself, implying that the period of the discrete-time signal is only 1 sample. If ω were π radians per sample, then the period becomes two samples, because each sample only covers half the period. Recall your experiences with the signal $e^{i\pi n} = (-1)^n$. Here, the discrete-time frequency is π, and the signal simply repeats every two samples.

2. Create a new VI called `DT Frequency Responses.vi`. Use a `MathScript Node` to model the frequency responses of each of the three discrete-time filters.

   (a) There will be no input for your `MathScript Node`.

   (b) Inside your `MathScript Node`, define an array `w` that represents the angular frequency ω sweeping from $-\pi$ to π.

   > In order to achieve as accurate a plot as possible, use a small step size: we recommend `0.03`, but you are free to experiment.

(c) The `MathScript Node` should have four outputs:

- `Hl`, which stores the array for the frequency response of the low-pass filter $H_L(\omega)$.
- `Hh`, which stores the array for the frequency response of the high-pass filter $H_H(\omega)$.
- `Hm`, which stores the array for the frequency response of the mid-pass filter $H_M(\omega)$.
- The array of angular frequencies `w`.

Note that you do not need `MathScript Node` outputs for the magnitude and phase responses of the filters separately. As you will see, LabVIEW allows you to extract the magnitude and phases of arrays of complex numbers.

> Remember a common source of errors in LabVIEW VIs: the data types for all inputs and outputs to any `MathScript Node` should be correctly assigned. In this case, the data type for `Hl`, `Hh`, and `Hm` should be a 1-D array of **complex** numbers (`CDB 1D`), instead of the usual 1-D array of real numbers.

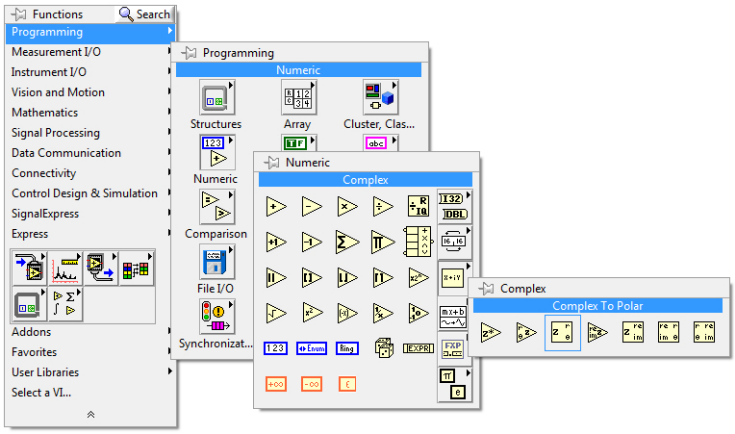
(d) Employ `MathScript` functionality in the `MathScript Node` to generate `Hl`, `Hh`, and `Hm` for the array of frequencies `w` used, based on the expressions for their respective frequency responses that you derived by hand in step 1. Note that you are not using the magnitude and phase response expressions that you derived separately. Rather, you only need to use the unsimplified expressions for the frequency responses.

> Recall from lab 03 that it is possible to perform operations on an entire array at once. As a result, the command `exp(x)` will generate an array whose i^{th} element is the value e^{x_i}, x_i being the i^{th} element of the array `x`. Again, the command `x .* y` will pointwise multiply arrays `x` and `y` together. Be careful of the difference between using the regular multiplication operator (`*`) and the pointwise multiplication operator (`.*`), when working with arrays.

3. Use the outputs of the `MathScript Node` from step 2 to determine the magnitude and the phase of the frequency response of each of the filters.

(a) Since the `Mathscript Node` outputs complex arrays, we can use the `Complex to Polar` block, as shown in Figure 2, available under the `Programming` → `Numeric` → `Complex` subpalette. This block breaks complex numbers into its polar components.

Figure 2 `Complex to Polar` block



(b) Connect the outputs `Hl`, `Hh`, and `Hm` to separate `Complex to Polar` blocks.

4. Plot the magnitude and the phase plots of $H_L(\omega)$, $H_H(\omega)$, and $H_M(\omega)$ against frequency ω.

(a) Recall from lab 04 that you can use either the `XY Graph` or the `Express XY Graph`, present under the `Graph` subpalette of the `Modern` palette, and available only on the front panel.

(b) In all, you should have six (6) `XY Graph` blocks on your front panel (one for each of the magnitude and phase plots of the three filters).

(c) Make sure to edit the labels on the x-axis to read `Frequency` and the labels on the y-axis to read either `Magnitude` or `Phase`.

Use the behaviors of the plots at frequencies 0, $\pm\pi/2$, and $\pm\pi$ to confirm the expressions you derived by hand in step 1.

5. Based on the magnitude and phase plots, can you see why the low-pass, high-pass, and mid-pass filters are given their respective names? What is your prediction of the effect of each of the filters on the signals $\cos(0\pi n)$, $\cos(\frac{\pi}{2}n)$, and $\cos(\pi n)$?

2.6 Variations on a Discrete-Time Filter

We will consider the following variations of our three discrete-time filters, which also depend on past values of the output signal $y(n)$. This is an example of a system with *feedback*. These filters are also LTI (can you verify this?), with LCCDEs given as follows:

For the low-pass filter, we have

$$\forall n \in \mathbb{Z}, \quad y(n) = \frac{1}{2}(x(n) + x(n-1)) + \frac{1}{2}y(n-1). \tag{1}$$

For the high-pass filter, we have

$$\forall n \in \mathbb{Z}, \quad y(n) = \frac{1}{2}(x(n) - x(n-1)) + \frac{1}{2}y(n-1). \tag{1}$$

For the mid-pass filter, we have

$$\forall n \in \mathbb{Z}, \quad y(n) = \frac{1}{2}(x(n) - x(n-2)) + \frac{1}{2}y(n-1). \tag{1}$$

Let us denote the frequency response of the variations of the discrete-time low-pass, high-pass, and mid-pass filters as $H_{LV}(\omega)$, $H_{HV}(\omega)$, and $H_{MV}(\omega)$, respectively.

1. Draw DAG block diagrams corresponding to the systems that implement these filters.

2. Derive expressions for the frequency responses $H_{LV}(\omega)$, $H_{HV}(\omega)$, and $H_{MV}(\omega)$.

> Hint: Recall that if we set our input signal $x(n)$ to be $e^{i\omega n}$, then our output signal $y(n)$ from the discrete-time LTI system is $H(\omega)e^{i\omega n}$. Then, to determine the frequency response expressions, substitute $e^{i\omega n}$ as the input signals and $H(\omega)e^{i\omega n}$ as the output signals into the LCCDEs for the filters. Now, solve for $H(\omega)$. Your final expressions should be functions involving powers of $e^{i\omega}$. If you have any term involving $e^{i\omega n}$, you may want to recheck your derivation, because $H(\omega)$ should be a function of ω alone.

3. Repeat the procedure in section 2.4 to create frequency response plots for the variations of the discrete-time filters using `Mathscript Node`.

4. How do the magnitude and phase plots of the filter variations *qualitatively* differ from the plots of the original filters? In other words, how does introducing a delayed version of the output $y(n-1)$ change the magnitude and phase of the frequency responses qualitatively?

2.7 Time Domain Analysis

We now move into the time domain, where we observe the effects of the discrete-time filters on actual signals of varying frequencies. However, we will only work with the low-pass filter variation described in section 2.6.

1. Create a new LabVIEW VI called `DT Low-Pass Filter Variation.vi`. We will model this VI on the VIs you created in the lab 04 post-lab, where you fed the inputs $\cos(0\pi n)$, $\cos(\frac{\pi}{2}n)$, and $\cos(\pi n)$ into discrete-time filters.

2. Use a `For Loop` to implement the block diagram for the low-pass filter variation that you saw in section 2.6.

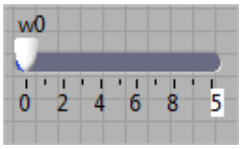
   > Note that in addition to delaying the input $x(n)$, you must also delay the output $y(n)$. Hence, you must use a separate set of shift registers for the input and the output. Recall that you can add delay elements to your shift registers to access values from beyond the previous iteration of the `For Loop`. How many delay elements will you need for the input? How many delay elements will you need for the output?

   > If LabVIEW begins to introduce `Feedback Node`s into your VI, do not be concerned: you are, after all, building a system with feedback.

3. This time around, we will not be feeding in signals with pre-designated frequencies. We will allow the user to determine the frequency of the input signal. To this end, create a `Horizontal Pointer Slide` on the front panel, available in the `Controls` palette under `Programming` → `Numeric`. Label this slide `w0` and edit its range to span from `-5` to `5`.

   > You can edit the range of the slider by editing the values on either end, as shown in Figure 3.

Figure 3 Editing the Range of a Slider



4. We will use 30 samples of $\cos(\omega_0 n)$ as our discrete-time input signal, where the argument ω_0 will be adjusted using the control `w0` from the previous step. Feed this input signal into your `For Loop` just as you did in the lab 04 post-lab.

5. Use either the `XY Graph` or the `Express XY Graph` to plot the output of your filter for particular values of ω_0. Be sure to use the correct labels for the axes of the graphs in the front panel.

6. Enclose the input signal, the `For Loop`, and the output `XY Graph` or `Express XY Graph` all in a `While Loop`, with a front panel control to stop the loop. This ensures that your VI runs continuously so that you can instantaneously observe changes in the output relating to changes in the input. Also, use a `Wait (ms)` block with a delay of 100 milliseconds to ensure that the CPU is not constantly occupied.

7. Run the VI that you created, while adjusting your ω_0 parameter. What happens as you sweep the parameter from $-\pi$ to π? What happens if the parameter moves outside the range of $-\pi$ to π? You should be able to relate your observations with your plot of $H_{LV}(\omega)$ in the previous exercise.

3 Post-Lab Section

In the post-lab section, we will continue our exploration of discrete-time filters by examining a comb filter and an all-pass filter.

3.1 Comb Filter

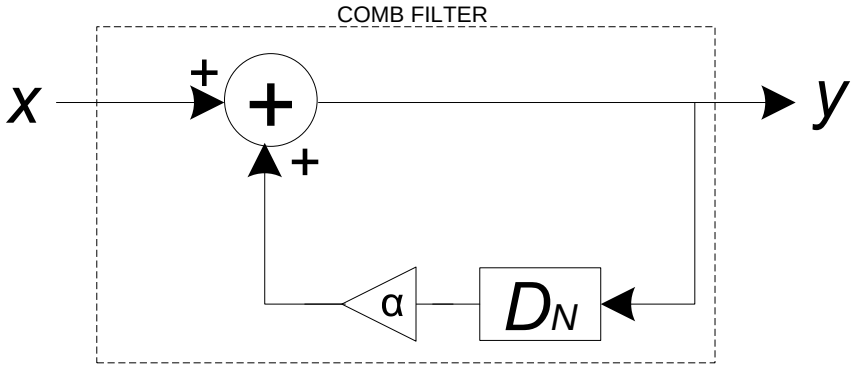
3.1.1 Theory

Consider a discrete-time system described by the following linear constant-coefficient difference equation:

$$\forall n \in \mathbb{Z}, \quad y(n) - \alpha y(n-N) = x(n).$$

The DAG block diagram representation of this system is shown in Figure 4.

Figure 4 A basic discrete-time comb filter.



This system is commonly used to model echo effects in sound signals. The parameter $\alpha \in \mathbb{R}$ is used to model the attenuation or amplification of the sound signal with each echo. The parameter N is a positive integer used to model the amount of time delay in every echo.

3.1.2 Exercises

1. Suppose that we define the frequency response of this system by $H_C(\omega)$. Show that the corresponding frequency response H_C is
$$\forall \omega \in \mathbb{R}, \quad H_C(\omega) = \frac{e^{i\omega N}}{e^{i\omega N} - \alpha}.$$

2. By performing a graphical analysis in the complex plane, using the 'tic-tac-toe' method that you have seen in lecture, provide well-labeled sketches of the magnitude response values $|F_C(\omega)|$ and phase response values $\angle F_C(\omega)$, $\forall \omega \in [-\pi, \pi)$, and $\alpha = \frac{1}{2}$.

3. Repeat the steps in section 2.4 to confirm your answers in the frequency domain, in a VI called `CF Frequency Domain.vi`. In addition, define the parameters α and N to be tunable by creating horizontal sliders for them. The allowed ranges for these values should be $-1 \leq \alpha \leq 1$ and $0 \leq N \leq 10$.

4. What are the effects on the magnitude response as α approaches the values 0, 1, and -1? What happens to the magnitude response as α changes from positive to negative?

5. What are the effects of changing N on the magnitude response? With this observation in mind, as well as noticing the *shape* of the magnitude response, conjecture why the filter is called a *comb filter*?

6. Moving into the time domain, make a copy of the `DT Low-Pass Filter Variation.vi` VI under the name `CF Time Domain.vi`. Make the appropriate changes to the LabVIEW block diagram to match the block diagram representation in Figure 4. You may find the `Y[i] = X[i-n] PtByPt` block useful, located under `Signal Processing` → `Point By Point` → `Signal Operation` `PtByPt`, useful. Make sure to use the `Y[i] = X[i-n] PtByPt` block, not the `Y[i] = X[i-n]` block!

7. Run the VI by varying your ω_0 parameter, and determine whether the results match your expectations.

3.2 All-Pass Filter

3.2.1 Theory

Consider a discrete-time system described by the following linear constant-coefficient difference equation:

$$\forall n \in \mathbb{Z}, \quad y(n) + \alpha y(n-1) = \alpha x(n) + x(n-1),$$

where the constant $\alpha \in \mathbb{R}, 0 < |\alpha| < 1$.

3.2.2 Exercises

1. Draw a DAG block diagram representation of the system, similar to the block diagram shown in Figure 4.

2. Suppose that we denote the frequency response of this system by $F_A(\omega)$. Show that the corresponding frequency response F_A is

$$\forall \omega \in \mathbb{R}, \quad F_A(\omega) = \frac{\alpha + e^{-i\omega}}{1 + \alpha e^{-i\omega}}.$$

3. By performing a graphical analysis in the complex plane, provide a well-labeled sketch of the magnitude response values $|F_A(\omega)|, \forall \omega \in [-\pi, \pi)$, and $\alpha = \frac{1}{2}$. As before, in the next few steps, you will confirm your answer in both the frequency and the time domains.

   > Hint: In order to make the analysis simpler, pull out a factor of $e^{-i\omega}$ from the numerator. Now, you are left with a numerator and a denominator that are conjugates of each other. What do we know about the relationship between the magnitude of a complex number and that of its conjugate?

4. Repeat the steps in section 2.4 to confirm your answers in the frequency domain, in a VI called `APF Frequency Domain.vi`. Plot *both* the magnitude and the phase responses.

5. What are the effects on the magnitude response due to changes in α? With this in mind, why is this filter called an *all-pass filter*? Why would such a filter be useful? As a hint, consider what you would do if you wanted a particular signal, and you did manage to obtain that signal, but with its phase shifted?

6. Moving into the time domain, make a copy of the `DT Low-Pass Filter Variation.vi` VI under the name `APF Time Domain.vi`. Make the appropriate changes to the LabVIEW block diagram to match the block diagram representation that you made in step 1.

7. Run the VI by varying your ω_0 parameter, and determine whether the results match your expectations.

3.3 Submission Rules

1. Late submissions will *not* be accepted, except under unusual circumstances.

2. These exercises are recommended to be done *in groups of two*. Only one person need submit the required files, however.

3.4 Submission Instructions

Answer the questions presented in section 3.1.2 and section 3.2.2, supplemented with appropriate screenshots of relevant front panels and relevant block diagrams, and turn in your answers *on paper* to your lab TA (or however your TA wishes to have you turn in), labeled with the names of the students who worked together. Templates for this assignment are available, in DOC and TEX formats, as part of the lab 5 resources on bSpace, but you need not use them.

This assignment is due **10 minutes** after the beginning of your lab session during the week of **April 4, 2011**.

4 Acknowledgments

Special thanks go out to the teaching assistants (TAs) of the Spring 2009 semester (Vinay Raj Hampapur, Miklos Christine, Sarah Wodin-Schwartz), of the Fall 2009 semester (David Carlton, Judy Hoffman, Mark Landry, Feng Pan, Changho Suh), and of the Spring 2010 semester (Xuan Fan, Brian Lambson, Kelvin So) for providing suggestions, ideas, and fixes to this lab guide.