MASSACHVSETTS INSTITVTE OF TECHNOLOGY
Department of Electrical Engineering and Computer Science
6.01—Introduction to EECS I
Fall Semester, 2007

Assignment Week 5, Issued: Tuesday, Oct. 2

To do this week: NOTE: TWO LECTURES, TWO SOFTWARE LABS!

...in Tuesday software lab, 10/2

1. Start writing code and test cases for the numbered questions in the software lab. Paste all your code, including your test cases, into the box provided in the "Software Lab" (Part 5.1) problem on the on-line Tutor. This will not be graded.

...before lecture on Thursday, 10/4

1. Nothing.

...in Thursday software lab, 10/4

1. Start writing code and test cases for the numbered questions in the software lab. Paste all your code, including your test cases, into the box provided in the "Software Lab" (Part 5.2) problem on the on-line Tutor. This will not be graded.

2. Get the handout for next Thursday's lab.

...No Lecture Tuesday, 10/9

...before the lab Thursday, 10/11

1. Do the on-line Tutor problems for week 5 that are due on Thursday (Part 5.3).

2. Do the writeup for the 10/2 and 10/4 software labs (in this handout) providing written answers (including code and test cases) for **every** numbered question in this handout.

...before lecture Tuesday, 10/16

1. Do the writeup for the 10/11 design lab (NOT in this handout).

2. Do the nanoquiz; it will be based on the material in the lecture notes and the on-line Tutor problems due on Thursday.

On Athena or the lab laptops make sure you execute:
`athrun 6.01 update`
so that you can get the `Desktop/6.01/lab5` directory which has the files mentioned in this handout.

- You need the file `differenceEquationWithInput.py`.
- You need the file `soar-graph.py`.

During this lab, please use the lab laptop for Tuesday's software lab. If you are using your own laptop for Thursday's software lab, download the files from the course Web site (on the Calendar page).

Tuesday Software Lab: Speed of difference equation solving

In this lab, we will be using python's features for timing software. Since our experience is that the timing functions are not very portable, please work in pairs and use the lab laptops.

In earlier lectures, you learned that the solution to a homogeneous difference equation can be represented in closed form using the equation's natural frequencies. In today's lecture you learned that when a difference equation has an input, that is the equation is not homogeneous, the solution is more complicated to compute and that computation is probably best accomplished using a program.

One general form for a difference equation with input (we will see another form just below) is

$$y(n) = \alpha_1 y(n-1) + \alpha_2 y(n-2) + \ldots \alpha_K y(n-K) + \beta_o x(n) + \beta_1 x(n-1) + \ldots \beta_L x(n-L)$$

where $x(n)$ is the input and is a given sequence. To determine $y(n)$ from the above difference equation, it is necessary to specify the values of the coefficients $\alpha_1 \ldots \alpha_K$ and $\beta_o \ldots \beta_L$, the initial values $y(0) \ldots y(K-1)$, and value of $x(n)$ for all n.

One way to specify the input x is use a procedure. For example, consider defining $x(n)$ using the python procedure

```
def x(n):
    return 1
```

In this case $x(n) = 1$ for all n.

In today's lab we will be examining and modifying an implementation of a difference equation class, and will then use that class to develop a feel for orders of growth. Examine the difference equation class defined in `differenceEquationWithInput.py`. You will notice that the difference class uses a different format for specifying a difference equation, one that turns out to be much more easily manipulated when using transform techniques to be described in a later lecture. Specifically, the difference equation format used in the difference equation class is

$$\sum_{k=0}^{K} a_k y(n+k) = \sum_{l=0}^{L} b_l x(n+l).$$

A note should be made about the order of the coefficients. In the polynomial class, the coefficients for a polynomial $z^2 + 2z + 3$ would be given as a list $[1, 2, 3]$. In following that convention, the

coefficients of a difference equation go from higher to lower order, so that a list of the y coefficients would be $[a_k, a_{k-1}, ..., a_0]$. Note that this same seemingly backward convention is true for initial conditions as well, the intial values are $[y(K-1), .., y(0)]$. This may seem confusing, as the k^{th} coefficient of a difference equation is not the k^{th} element of the list. We agree, but the common convention is against us.

Consider the Fibonacci example. In order to create an instance of the difference equation class that can be used to compute the Fibonacci numbers, it is necessary to create an instance associated with the difference equation

$$y(n) = y(n-1) + y(n-2).$$

Here, there is no input, and therefore $x(n) = 0$ for all n. Loading differenceEquationWithInput.py and then typing the following command in the interpreter:

```
>>> fib = DifferenceEquationWithInput([1, 0], [1, -1, -1], [1], lambda n: 0)
```

will create an difference equation instance which will generate the Fibonacci numbers.

Question 1. How are the α_k's and β_l's related to the a_k's and the b_l's in the above two forms of difference equations?

If you examine the *DifferenceEquationWithInput* class, you will notice that two methods are implemented for computing the n^{th} value of the solution to the difference equation. One method uses iteration, and the other uses recursion. To test your understanding of the class implementation, please answer the following questions.

Question 2. What does the line *vals = [nextVal] + vals[:-1]* in the *valIter* function accomplish?

Question 3. What does the line

```
dotProd(self.outCoefficients[1:], [self.valRecur(n-i-1) for i in range(self.order)])
```

in the *valRecur* function accomplish?

A clever plot

It can often be revealing to plot the series of values a difference equation generates. Please refer to the week 4 assignment for instructions on how to plot using SoaR (the assignment is on the course calendar page if you do not have it handy).

Question 4. Demonstrate that you understand how the difference equation class works by using the class to solve and plot the solution to the difference equation $y(n) = 0.5y(n-1) + x(n-1)$ where $y(0) = 0$ and $x(n) = 1$ for all n.

Question 5. Now use the difference equation class to solve and plot the solution to the difference equation $y(n) = -0.5y(n-1) + x(n-1)$ where $y(0) = 0$ and $x(n) = 1$ for all n.

Question 6. Explain why the two plots look the way they do, and explain the differences between the two plots.