Midterm A1 / A2 / B1 / B2
Section 4 / 2 / 4 / 2

November 9, 2014

Problem 2 / 3 [10pt]: Finish the following function. For input: f is **a** function so that $f(x_j)$ returns the value of **an** interpolating polynomial at x_j and x is a row vector of values. For output: y is a row vector of the values $f(x_j)$ for each element in x. Assume that $f(x_j)$ only accepts *one value at a time*.

function y = evaluateF(f, x)

end

This is a *very* tricky problem. When I sat down to take this exam, I almost missed this one completely. In the previous problem, you either wrote the function $f(x)$ as a linear system (A1 / A2) or you found a Lagrangian polynomial named $f(x)$ (B1 / B2). In any case, you've done a lot of work to accurately describe $f(x)$. If this problem were stated like this

Problem 2 / 3 [10pt]: Finish the following MATLAB function named "evaluateF." For input: g is a function so that $g(t_j)$ returns the value of **an** interpolating polynomial at t_j and t is a row vector of values. For output: y is a row vector of the values $g(t_j)$ for each element in t. Assume that $g(x_j)$ only accepts one value at a time.

```
function y = evaluateF( g, t )


end
```

then what we were asking may have been a bit more clear. The first part of solving this problem is recognizing that the f in this problem is *not necessarily* the same f from the last problem. With that in mind, let's look back over the statement of **Problem 2 / 3** so that we can identify all of our given information, and decipher what output we want to produce.

The problem statement tells us that we are given two input variables. One is a function f which takes only one value at a time, the other is a row vector, x. Since f takes only one value at a time, we can't do this $f([1, 2])$, or this $f([x_1, x_2, x_3])$, or this $f([x_1, x_2, \ldots, x_n])$, or so on. The best we can hope to do is throw one value into f at a time, like this $f(1), f(2)$, or this $f(x_1), f(x_2), f(x_3)$, or this, $f(x_1), f(x_2), \ldots, f(x_n)$.

We also know that x is a row vector. But we don't know how many entries it has. So as I'm getting my head around this problem, I'm just going to write down what x looks like in general. We have

$$x = [x_1, x_2, \ldots, x_m]$$

for some integer m with $m \geq 1$. That is, x may have 100 entries, it may have 1000 entries, or it may have only one entry. In any case, we want our code to be able to take a given row vector (of any length) and compute ... what? Well, the problem says that the output, y, needs to be a row vector of values $f(x_j)$ for *each* element in x. Since I just wrote $x = [x_1, x_2, \ldots, x_m]$ above, it's pretty obvious how y will look. We'll get

$$y = [f(x_1), f(x_2), \ldots, f(x_m)]$$

Stop! After doing some analysis, let's review the problem statement and make sure we're on the right track.

Have we accounted for every bit of information given in the statement above? We've looked at x, f, and we've got a descent candidate for y. Since the statement tells us that y is a row vector of the values $f(x_j)$ for *each* element in x, and since x has the form $x = [x_1, x_2, \ldots, x_m]$, then we can conclude that the vector y (which we want to make our output) needs to have the form $y = [f(x_1), f(x_2), \ldots, f(x_m)]$. But we know that we can't simply do this

$$y = f(x)$$

because $x = [x_1, x_2, \ldots, x_m]$. We were told *explicitly* that $f(x) = f([x_1, x_2, \ldots, x_m])$ can **not** work because f takes only one value at a time. Then we'll need to come up with another way to fill a row vector y with the m values $f(x_j)$. Notice that we know how big our row vector y has to be. Each entry in y corresponds to a given entry in x. Then y and x must be row vectors of the same size. In summary, we have

$$x = [x_1, x_2, \ldots, x_m]$$

and we want

$$y = [f(x_1), f(x_2), \ldots, f(x_m)].$$

Stop! Did we get all of the information we need from the problem statement? I think so. The two relations above describe the entire problem.