1. Consider the degree m polynomial $c_1 + c_2x + c_3x^2 + \cdots + c_{m+1}x^m$. An efficient method for evaluating this polynomial, called Horner's method, is introduced in section 0.1 of the textbook. This approach nests the coefficients of the polynomial as

$$c_1 + x\left(c_2 + x\left(c_3 + x\left(\cdots + x\left(c_m + x\left(c_{m+1}\right)\right)\cdots\right)\right)\right).$$

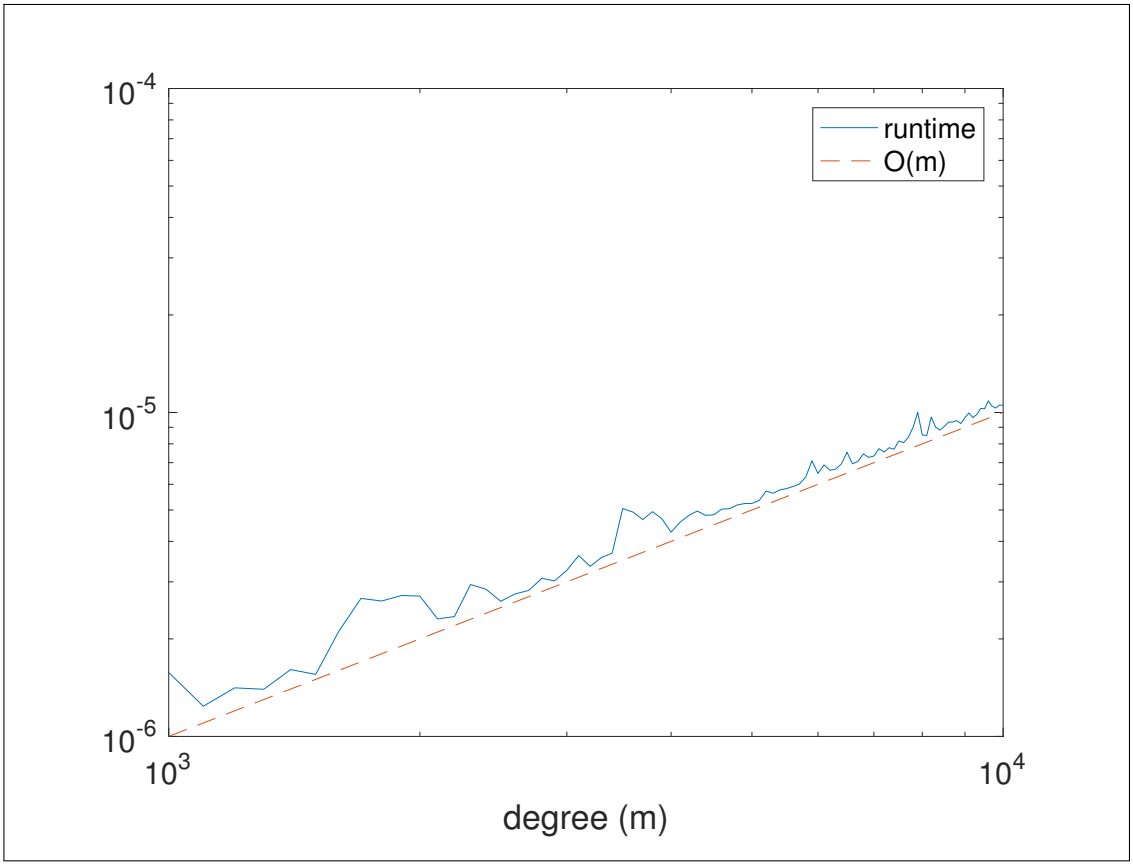
   (a) Write a Matlab functions with definition `y = horner(c,x)` implementing Horner's method for evaluating a degree m polynomial with coefficients in `c`. Include a listing of each function in your solutions. The input `c` is a vector containing the $m+1$ coefficients and `x` is a vector of values at which to evaluate the polynomial. The output `y` is the result of the evaluations.

Solution: The following listing gives my implementation of Horner's method.

```
1 function y = horner(c,x)
2
3 y = c(end);
4
5 for k = 1:length(c)-1
6   y = y.*x + c(end-k);
7 end
8
9 end
```

   (b) Determine the asymptotic runtime as the degree m gets large by measuring the time it take to execute each command at a variety of values of m. You will need to average over many executions in order to get meaningful data. Include a log-log plot of runtime versus m which demonstrates your solution for each command.

Solution: The following plot gives the runtime of Horner's method for values of m ranging from 10^3 to 10^4. I averaged the runtime over 10^4 evaluations for each value of m in order to account for typical variations. As can be seen from the plot, the runtime for the method asymptotically follows a line proportional to m. Therefore this is an $\mathcal{O}(m)$, or linear time, algorithm.


10^{-4}
10^{-5}
10^{-6}
10^{3}
10^{4}
runtime
O(m)
degree (m)