Recitation Problems – Week 8

440:127 – Spring 2015 – S. J. Orfanidis

You may wish to get started on this set in advance of your session.

1. Suppose that you start a new savings account with an initial deposit of \$1,000 and that from then on you deposit \$1,200 a month. The account collects 3 % annual interest. The accumulated balance at the end of the kth month can be represented by the recursion:

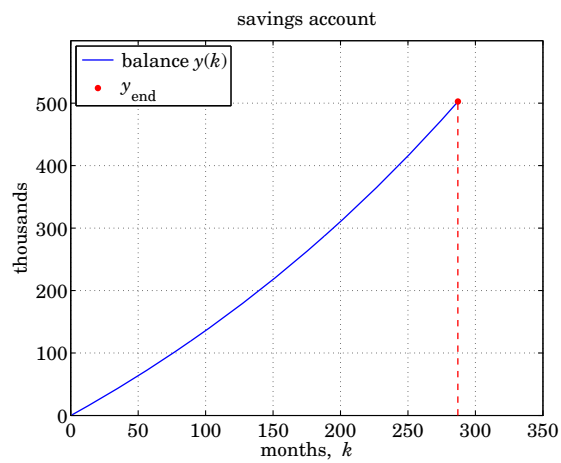
$$y(k) = y(k-1) + r \cdot y(k-1) + 1200, \quad k \geq 2$$

and initialized at $y(1) = 1000$, with effective monthly rate of $r = (3/100)/12 = 0.0025$.

   a. Using a for-loop determine how many months it would take to reach a desired balance of \$500,000.
   b. Repeat the previous part using a conventional while-loop.
   c. Repeat using a forever while-loop.
   d. Make a plot of the balance $y(k)$ versus k. Plot the horizontal axis in months, and the vertical axis in thousands of dollars. Add the ending balance on the graph.
   e. Write a function, **account**, that uses method (c) above, and given the initial balance y_0, annual percentage rate R, monthly deposit x, and desired final balance $y_{\max}$, it returns the vector of monthly balances y until the goal of $y_{\max}$ is reached. It must have usage:

```
y = account(y0, R, x, ymax)
```

   Note that the length N of the array $y(k)$ is the required number of months to reach the savings goal, and $y(N)$ is the ending balance, which should be just a bit higher than $y_{\max}$.



2. In the week-7 recitation, you solved the following equation using the built-in function **fzero**,

$$\exp(-x^2) = \frac{x}{2} \tag{1}$$

   a. Solve Eq. (1) iteratively by rearranging it in the equivalent way, $\exp(x^2) = 2/x$, or, $x^2 = \ln(2/x)$, or,

$$x = \sqrt{\ln\left(\frac{2}{x}\right)} \tag{2}$$

   and replacing it by the following iteration, initialized at some arbitrary value x_1 that lies somewhere in the range, $0 < x_1 \leq 1$, e.g., $x_1 = 1$,

$$x_{k+1} = \sqrt{\ln\left(\frac{2}{x_k}\right)}, \quad k = 1, 2, 3, \ldots \tag{3}$$

Implement this iteration using a *forever while-loop* that exits when two successive iterates become closer to each other than some specified error tolerance such as, $tol = 10^{-10}$, that is, the iteration terminates when, $|x_{k+1} - x_k| < tol$.

Determine the number of iterations N that were required for convergence. The last value $x(N)$ is the required solution of Eq. (1). Save the iterates into a MATLAB array $x(k)$ and plot it versus the iteration index $1 \le k \le N$. Also plot the error difference $e_k = |x_{k+1} - x_k|$, for $1 \le k \le N - 1$, using a **semilogy** plot and add the ending-point to the graph. See example graphs below.

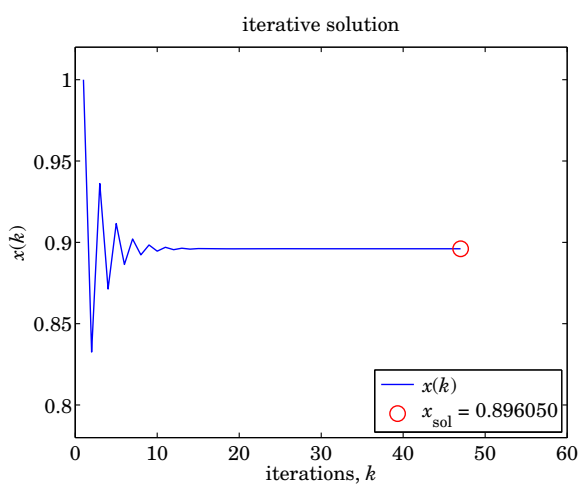
b. Implement the iteration of Eq. (3) using a *conventional while-loop*. Verify that this method produces the identical array $x(k)$ as method (a).

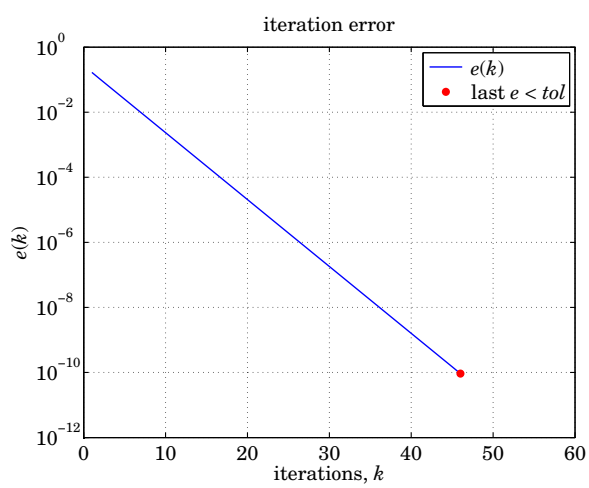
c. Solve Eq. (2) directly by using the built-in function **fzero**. Using trial-and-error determine an appropriate value for the error tolerance, *tol*, of part (a) such that the iterative and the **fzero** solutions agree in at least 12 decimal places.

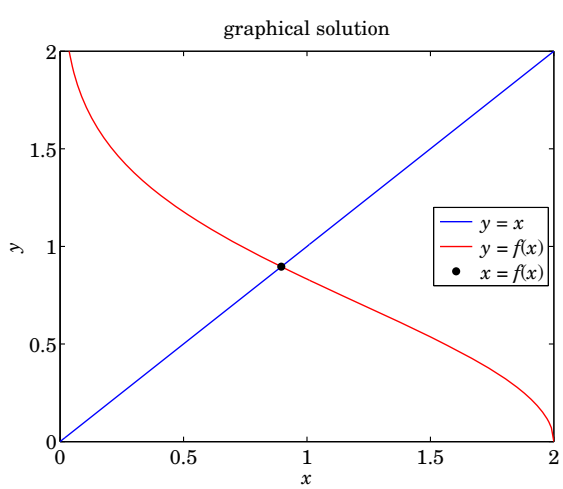
d. The solution of Eq. (1) lies at the intersection of the two curves:

$$y = x \quad \text{and} \qquad y = f(x) = \sqrt{\ln\left(\frac{2}{x}\right)}$$

Plot both curves on the same graph over $0 < x < 2$ and place the solution point on the graph.







3. The following infinite series converges to the number 100:

$$\sum_{n=0}^{\infty} (n+1)\,(0.9)^n = 100$$

Define the partial sums, for $k \geq 0$,

$$S_k = \sum_{n=0}^{k} (n+1)\,(0.9)^n = (0+1)\,(0.9)^0 + (1+1)\,(0.9)^1 + (2+1)\,(0.9)^2 + \cdots + (k+1)\,(0.9)^k$$

which can be cast in the recursive form:

$$\begin{aligned} S_0 &= (0+1)\,(0.9)^0 = 1 \\ S_k &= S_{k-1} + (k+1)\,(0.9)^k, \quad k \geq 1 \end{aligned} \tag{4}$$

a. Define the following function $f(k)$ in Matlab as a single-line vectorized anonymous function and plot it vs. k in the interval $0 \leq k \leq 300$ using **semilogy**, noting that it rapidly decreases with k,

$$f(k) = (k+1)\,(0.9)^k, \quad k \geq 0$$

Because $f(k)$ represents the kth term in the above series, that is, $f(k) = S_k - S_{k-1}$, one can estimate the minimum value of k beyond which the difference $|S_k - S_{k-1}|$ will become smaller than a specified error tolerance, say, $tol = 10^{-10}$, by solving the following equation for k:

$$f(k) = tol \quad \Rightarrow \quad (k+1)\,(0.9)^k = tol$$

Solve this equation for k using the function **fzero**, and using the function **ceil**, round the result up to the next integer, denoted by, K. This represents the maximum number of terms that we need to use in the series S_k to achieve an accuracy of $tol = 10^{-10}$. Indicate the point at $k = K$ on the graph of $f(k)$.

b. Using a *forever while-loop* whose stopping condition is the inequality $|S_k - S_{k-1}| \leq tol$, calculate S_k and determine the number K of iterations until the loop is exited.

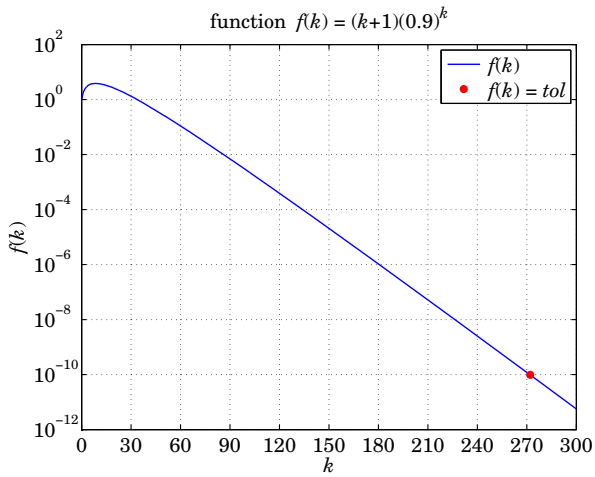
Save the iterates into a Matlab array, say $S(k+1)$, for $0 \leq k \leq K$, and plot it versus k, indicating the final computed value with a red marker, as well as the exact value of the limit. See example graph at end.

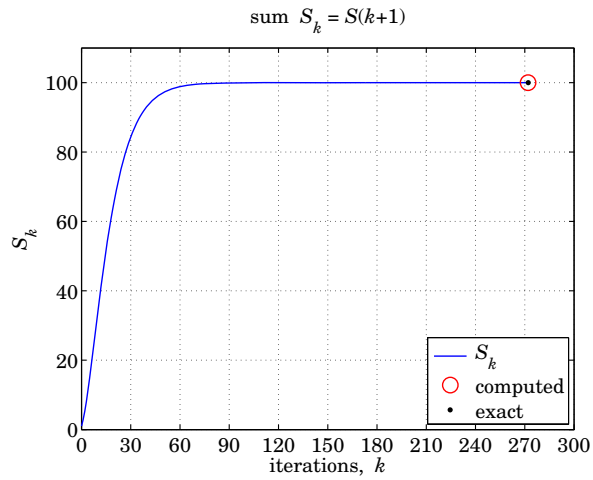
Note: The summation index $k = 0, 1, 2, \ldots$, is not a Matlab index because it starts from $k = 0$, and hence S_k is not a Matlab array. However, we can map it into a Matlab array by shifting the index, i.e., $S(k+1) = S_k$, for $k = 0, 1, 2, \ldots$. The recursion can then be rephrased in terms of the Matlab array:

Math Notation		Matlab Notation
$S_0 = (0+1)\,(0.9)^0 = 1$	$\Rightarrow$	$S(1) = (0+1)\,(0.9)^0 = 1$
$S_1 = S_0 + (1+1)\,(0.9)^1$		$S(2) = S(1) + (1+1)\,(0.9)^1$
$S_k = S_{k-1} + (k+1)\,(0.9)^k, \quad k \geq 1$		$S(k+1) = S(k) + (k+1)\,(0.9)^k, \quad k \geq 1$

c. Repeat part (b) using a *conventional while-loop*, whose continuation condition is the inequality, $|S_k - S_{k-1}| > tol$, that is, it will loop until this condition is violated. Determine the final value of k upon exit from the loop and compare it with the K of part (b).

d. Repeat parts (b-c) without saving the series values into an array $S(k)$, but rather using only two scalar variables S and S_{old} that are recycled in each iteration. Skip the plots of part (b) in this case, but in all cases do print out the final value of k and the final approximation error $E = |S - 100|$.


function $f(k) = (k+1)(0.9)^k$
$f(k)$
$f(k) = tol$
10^{2}
10^{0}
10^{-2}
10^{-4}
10^{-6}
10^{-8}
10^{-10}
10^{-12}
0 30 60 90 120 150 180 210 240 270 300
k


sum $S_k = S(k+1)$
S_k
computed
exact
0 20 40 60 80 100
0 30 60 90 120 150 180 210 240 270 300
iterations, k