1. **FOR:**

- **Refrigerator Cooling:** A can of soda (soft drink) at temperature $25°C$ is placed in a refrigerator, where the ambient temperature F is $10°C$. A standard way of determining how the soda temperature changes over a period of time is to subdivide the time interval into a number of small steps, each of duration Δt. If T_i is the temperature at the beginning of step i, the following model can be used to determine T_{i+1}:

$$T_{i+1} = T_i + K\,\Delta t\,(F - T_i),$$

where K is the conduction coefficient, a parameter that depends on the insulating properties of the can and thermal properties of the soda. Write a script to compute and plot the soda temperature with $K = 0.05$ and $\Delta t = 1$minute. This cooling problem has an exact mathematical solution:

$$T_{exact} = F + (T_0 - F)e^{-Kt}$$

Edit your script so that it plots both the exact and numerical temperatures on one graph.

- **Signal Filtering:** Denoting a noisy input sample signal by $x(k)$ and a filtered output sample signal by $y(k)$, a *3-point moving average filter* is represented by:

$$\begin{aligned} y(1) &= x(1), \\ y(2) &= \frac{1}{2}[x(1) + x(2)], \\ y(k) &= \frac{1}{3}[x(k) + x(k-1) + x(k-2)], \qquad \text{if} \qquad k \geq 3. \end{aligned}$$

Write a MATLAB script file to test this filtering method. Use

$$x = \sin\left(\frac{2\pi}{5}t\right) + Noise,$$

where t is a vector with 400 elements starting from $t = 0$ & ending at $t = 10$ and where $Noise$ is a set of normally distributed random numbers with mean zero and standard deviation 0.1: the MATLAB command to create a vector of such random numbers with the same length as vector t is:

$$Noise = 0.1 * \text{randn}(1, \text{length}(t)), \qquad \text{or} \qquad Noise = 0.1 * \text{randn}(\text{size}(t)).$$

Use subplot commands to plot (i) top graph: the noisy input signal x against t and (ii) bottom graph: the filtered signal y against t.

- **Iteration:** Consider the problem of solving the equation $f(x) = x$ where $f(x)$ is specified by the nonlinear function:

$$f(x) = \frac{1}{2}\,x\,(4 - x)\,\sin\left(\pi\sqrt{\frac{x}{2}}\right).$$

(i) Write a short script m-file to plot (on the same diagram) the graphs of $y = x$ and $y = f(x)$ in the interval $0 \leq x \leq 1.5$; use 100 points for all vectors in the plot.

(ii) In a separate script m-file, use the iteration formula:

$$x_0 = 0.1,$$

$$x_{i+1} = \frac{1}{2}\, x_i\,(4 - x_i)\, \sin\left(\pi \sqrt{\frac{x_i}{2}}\right), \qquad \text{for} \qquad 0 \le i \le N-1,$$

to compute (after N iterations) the root of the equation $x = f(x)$ that lies in the interval $1 < x < 1.5$. Use several values of N (say $N = 5, 8, 15, 20, 50, 100$) and each time display the last element in the vector x using the **disp** command.

(iii) Modify your script file above so that it plots (on the same diagram) using **hold on** and **hold off** commands:

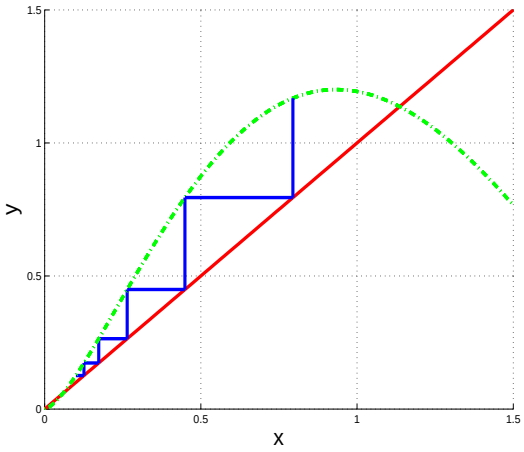
(a) the line $y = x$, in the interval $0 \le x \le 1.5$ using 100 points for both vectors in the plot,

(b) the curve $y = f(x)$, also in $0 \le x \le 1.5$ and using 100 points for both vectors in the plot,

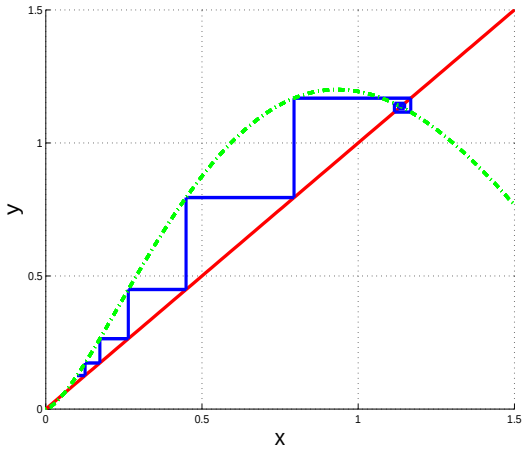
(c) the series of horizontal lines joining the points $(x_i, f(x_i))$ and $(x_{i+1}, f(x_i))$,

(d) the series of vertical lines joining the points $(x_{i+1}, f(x_i))$, $(x_{i+1}, f(x_{i+1}))$.

Your graph should look something like the following:



$N = 5$



$N = 10$

2. **IF:**

- View and copy the lecture m-files **vote.m**, **leap.m** and **Qroots2.m** on Vula under Resources. Look at how the if statements are used. Run these m-files once or twice.
- **FOR with IF:**
  Write an m-file **evensum.m** to sum the even numbers between from 1 to 100:

```
% evensum.m
clc
clear

sum1 = 0;
```

```
for n = 1 : 100;
    if rem(n,2) == 0
        sum1 = sum1 + n;
    end
end
disp(['the required sum is ', num2str(sum1),' using the for/if loop'])
```

Make the most minor change to the script so that it sums the odd numbers instead (there are 3 possible ways, can you find them all?)

- The m-file **maxind.m** searches a given matrix for the element with the largest value and return the indices (position) of that element. Copy it into your working directory and run it a few times.

```
% maxind.m returns row (r) and column (c) indices of
% the maximum-valued element in matrix A
clc
clear

A = rand(10,8)
[m n] = size(A);
Amax = A(1,1);
r = 1; c = 1;
for rowind = 1 : m
    for colind = 1 : n
        if A(rowind , colind) > Amax
            Amax = A(rowind , colind);
            r = rowind;
            c = colind;
        end
    end
end
disp(['row index is ', num2str(r)])
disp(['column index is ', num2str(c)])
```

- Modify the above m-file (and call the new m-file MinInd.m) so that it searches a given vector $V = \text{rand}(10, 1)$ for the minimum valued element.

3. **Refrigerator Cooling:** Redo question 1(a) but in this case stop the computation as soon as the temperature (T) reaches or just falls below $15°C$.

   A script to compute and plot the soda temperature with $F = 10$, $K = 0.05$ and $\Delta t = 0.5$minutes is given in the m-file, **T5Q3.m**. Copy this file from Vula, fill in the missing information and run it.

   The display commands in the script file should print the final time and temperatures as is shown below:

```
   time         T      Texact

  22.0000   14.9237   14.9931
```