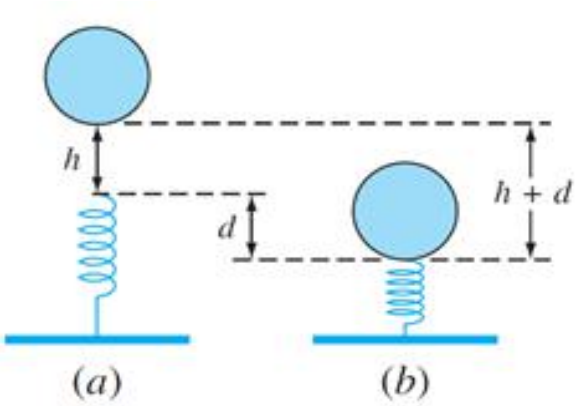
PROBLEM A1

Real mechanical systems may involve the defections of non-linear springs. In the Figure below, a mass m is released a distance h above a non-linear spring. The resistance force F of the spring is given by

$$F = -(k_1 d + k_2 d^{\frac{3}{2}})$$



Conservation of energy can be used to show that

$$0 = \frac{2k_2 d^{\frac{5}{2}}}{5} + \frac{1}{2}k_1 d^2 - mgd - mgh$$

Let $f(d)$ be a function such that,

$$f(d) = \frac{2k_2 d^{\frac{5}{2}}}{5} + \frac{1}{2}k_1 d^2 - mgd - mgh \quad (1)$$

To Find : We need to solve for d using the Bisection method

Given : $k_1 = 50000\frac{g}{s^2}$, $k_2 = 40\frac{g}{s^2 m^{0.5}}$, $m = 90g$, $g = 9.81\frac{m}{s^2}$, $h = 0.45m$

By putting the given values in the $f(d)$ we get,

$$f(d) = 16d^{\frac{5}{2}} + 25000d^2 - 882.9d - 397.305$$

Procedure : We will use the below steps to solve for $f(d) = 0$ using bisection method.

Step 1: Choose lower x_l and upper x_r guesses for the root such that the function changes sign over the interval. This can be checked by ensuring that $f(x_l)f(x_u) < 0$.

Step 2: An estimate of the root x_r is determined by $x_r = \frac{x_l + x_u}{2}$

Step 3: Make the following evaluations to determine in which subinterval the root lies:

1. If $f(x_l)f(x_r) < 0$, the root lies in the lower subinterval. Therefore, set xu = xr and return to step 2.
2. If $f(x_l)f(x_r) < 0$, the root lies in the upper subinterval. Therefore, set xl = xr and return to step 2.
3. If $f(x_l)f(x_r) = 0$, the root equals xr; terminate the computation.

- To compute the above procedure of the Bisection method, we can either use the recursion process where we will call the function after each iteration or we can use the looping process. For the sake of this problem, we will use a looping process using a while loop.
- Using the Bisection method, we know that there might be a chance that can't reach the exact answer of the equation. For example, if the root is 12.3426789 or any irrational number, the computer will require to do many iterations, and it might cause a runtime error.
- So there need to be some stopping criteria within the loop after which the program will exit and we can get a sufficiently close answer to our actual solution.
- So we can use two ways or criteria as a threshold condition in our program.
  - I. Set a limit to the number of iterations that our program will perform by defining a $itermax$ variable such that if $iter >= itermax$ then stop the loop and return x_r.
  - II. Use approximation error(ε_a) to keep track on x_r. Define a value for the variable ε_s such that if $\varepsilon_a < \varepsilon_s$ the loop will end and will return the optimised root x_r.

    Where $\varepsilon_a(\%) = \frac{x_r^{new} - x_r^{old}}{x_r^{new}} \times 100.$

Solution :

- So to solve the problem we first need to define the values of stopping criteria for our program.
- Let us take the max number of iteration to be 100 and the minimum value of approximation error ε_a that we want to achieve be 0.05%.

$$itermax = 100 \; and \; \varepsilon_s = 0.05$$

- The next step is to guess a lower x_l and upper x_u such that $f(x_l)f(x_u) < 0$.

  So let perform some hit and trial method to get guess. Take $x_l = 0 \; and \; x_u = 0.5$.

$$f(x_l)f(x_u) = f(0)f(0.5) \approx -2.15 \times 10^6$$

- Hence the condition is satisfied and we can take $x_l = 0$ and $x_u = 0.5$
- The result of the bisection method is shown below

Iteration	x_l	x_u	x_r	ea(%)
1	0.000000	1.000000	0.500000	-
2	0.000000	0.500000	0.250000	100.0
3	0.000000	0.250000	0.125000	100.0
4	0.125000	0.250000	0.187500	33.333333
5	0.125000	0.187500	0.156250	20.0
6	0.125000	0.156250	0.140625	11.111111
7	0.140625	0.156250	0.148438	5.263158
8	0.140625	0.148438	0.144531	2.702703
9	0.144531	0.148438	0.146484	1.333333
10	0.144531	0.146484	0.145508	0.671141
11	0.144531	0.145508	0.145020	0.3367
12	0.144531	0.145020	0.144775	0.168634
13	0.144775	0.145020	0.144897	0.084246
14	0.144897	0.145020	0.144958	0.042105

- You can see the number of iteration required to come to closest answer to the real solution is 14.
- The final answer that we got is $d = 0.144958\, m$.
- Note that at the end of the last iteration, the percentage approximation error is less than the decided value of ε_s i.e. 0.05%.

Note : If you want to get more precise answer, you can lower the value of ε_s further more Suppose we take $\varepsilon_s = 0.005\%$ the answer we get is $d = 0.144932\, m$ which is more precise than the earlier. Just the number of iteration will keep on increasing as we go on reducing ε_s.

Observation :

I. If we decrease ε_s , the number of iterations will increase and we will get an answer more closer to our actual true solution. For example if we take $\varepsilon_s = 0.005$, then the output will be

Iteration	x_l	x_u	x_r	ea(%)
1	0.000000	1.000000	0.500000	-
2	0.000000	0.500000	0.250000	100.0
3	0.000000	0.250000	0.125000	100.0
4	0.125000	0.250000	0.187500	33.333333
5	0.125000	0.187500	0.156250	20.0
6	0.125000	0.156250	0.140625	11.111111
7	0.140625	0.156250	0.148438	5.263158
8	0.140625	0.148438	0.144531	2.702703
9	0.144531	0.148438	0.146484	1.333333
10	0.144531	0.146484	0.145508	0.671141
11	0.144531	0.145508	0.145020	0.3367
12	0.144531	0.145020	0.144775	0.168634
13	0.144775	0.145020	0.144897	0.084246
14	0.144897	0.145020	0.144958	0.042105
15	0.144897	0.144958	0.144928	0.021057
16	0.144928	0.144958	0.144943	0.010527
17	0.144928	0.144943	0.144936	0.005264
18	0.144928	0.144936	0.144932	0.002632

II. If we decrease the boundness of the initial gusses(x_l *and* x_r) that we made, their might not be any significant increase in the precision of the solution or any increase in number of iterations.