# MTH308 Mini Project 1

Shivansh Rai

Roll No: 14658

## The problem

Given a parametric curve

$$C(t) = (X(t), Y(t)), 0 \leq t \leq 1 \tag{1}$$

and a point $(x_0, y_0)$, we need to compute the orthogonal projection of this point on the curve.
An orthogonal projection from a point to a curve is a point on the curve to which it is closest.

## Algorithm

The method used for the purpose of numerically solving this problem is a variant of the Steepest Descent algorithm.
The following steps describe the algorithm used -

1. Since the period of the given curve is 1,

$$f(t+1) = f(t) \tag{2}$$

   The distance squared function is given by -

$$f(t) = (X(t) - x_0)^2 + (Y(t) - y_0)^2 \tag{3}$$

   A period of 1 implies that the possible values of $t$ lie in the range $[0, 1]$. After this range, the behavior of the function will repeat.
   The values of $t$ are iterated from 0.0 to 0.9 with a step value of 0.1. The value which gives the least value of the distance function is chosen to be initial guess $t_0$ for $t$.

2. By gradient descent algorithm, the value of the parameter $t$ at $(n+1)^{th}$ iteration is given by

$$t_{n+1} = t_n - k\nabla f(t_n) \tag{4}$$

The above equation is dependent on k. An ideal value of k will be the one which minimizes the given parametric function

$$f(t_{n+1}) = f(t_n - k\nabla f(t)) \tag{5}$$

$\Rightarrow$ k satisfies the following equation

$$\nabla f(t_{n+1}).\nabla f(t_n) = 0 \tag{6}$$

Ideally in this case, we need to find the roots of the problem using a numerical method.
One approach for doing so will be to use Matlab's symbolic toolbox. However, this computation will add up to the total runtime.
Another approach for this will be to use a numerical method to solve for the roots. However, this will reduce the probability that the convergence criterion is met and the solution may diverge.

3. It should be noted that we initially computed the value of $t$ which gives a minimum value of distance function. Therefore, I replaced the value of $k$ with a constant value of 0.001. Since we start close to the solution, it is guaranteed that we'll converge in each iteration towards the solution.

The above use-case will be clear from an example.
Suppose the root of the problem lies to the left of the initial guess $t$. In this case the gradient of the curve is positive at $t$.

$$a > 0 \Rightarrow a\nabla f(t) > 0 \tag{7}$$

Therefore, in the $n^{th}$ iteration, the computed value of $t_{n+1}$ reduces (i.e. less than the previous value $t_n$, converging towards the solution.

Without loss of generality, the same can be proven for the case when the solution lies to the right of our initial guess. The computed value of $tn+1$ will move towards right again converging towards the solution. Hence, we make progress converging towards the solution at each iteration.

4. In case the number of iterations start getting larger, we limit them to a fixed value of 50. This is because the value of $t0$ which is initially chosen is fairly close to the solution, and hence it is certain that the function will definitely converge in these many iterations.