NUMERICAL ANALYSIS

PROJECT – 3

Name – Mrinal Saini	**Roll No. – BS1920**	**Semester – II**

Newton Raphson Method: The Newton Raphson is a powerful technique for solving equations numerically. It solves the equation $f(x) = 0$, assuming that we can compute $f'(x)$. This method requires one appropriate starting point x_0 as an initial assumption of the root of the function $f(x) = 0$. At $(x_0, f(x_0))$ a tangent to $f(x) = 0$ is drawn. Equation of this tangent is given by

$$\boldsymbol{y = f'(x_0)(x - x_0) + f(x_0)}$$

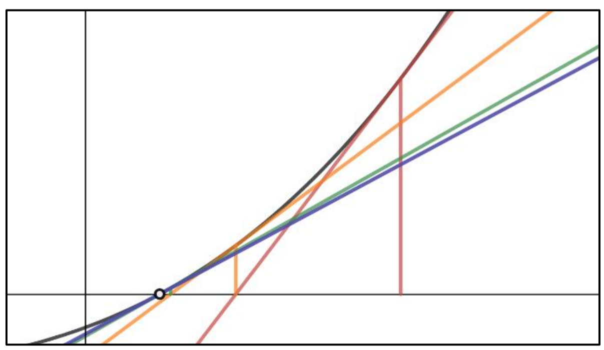
The point of intersection, say x_1, of this tangent with x – axis is taken to be the next approximation to the root of $f(x) = 0$. So, on substituting $y = 0$ in the tangent equation we get

$$\boldsymbol{x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}}$$

Now the successive approximations $x_2, x_3, x_4, \ldots \ldots ..$ etc. may be calculated by the iterative formula

$$\boldsymbol{x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}}$$

We iterate the above process till we get two consecutive iterations approximately equal i.e. $x_{n+1} \sim x_n$. The geometrical description of the method may be clearly visualized in the figure below:



Project: A certain trait in rabbits is controlled by a pair of alleles, **a** and **A**. Each rabbit receives one of these from the father and another from the mother. Thus, the possible pairs are **aa**, **aA** and **AA**. The probability that a parent gives an **a** to the offspring is p and probability of an **A** is $q(= 1 - p)$. The father's contribution is independent of the mother's, and so the probabilities of **aa**, **aA** and **AA** in the offspring are p^2, $2pq$ and q^2 respectively. Our aim is to estimate p. Unfortunately, it is impossible to detect the pair an offspring has. It is only possible to detect if an offspring has at least one **A**, *i.e.*, whether **aa** or {**aA**, **AA**}. The probabilities are p^2 and $q^2 + 2pq$ respectively. In a random sample of 100 offspring only 23 are without **A**. The probability of this is

$$\boldsymbol{L(p) = p^{46}\left(q^2 + 2pq\right)^{77} = p^{46}\left(1 - p^2\right)^{77}}$$

The value of $p \in (0,1)$ for which this is the maximum is called the **maximum likelihood estimator** of p. Find it.

Procedure:

1.) Find the first and the second derivatives of the likelihood.
2.) Since, at the maxima the derivative is zero, therefore find the root of the derivative.
3.) Now, to find the root apply Newton-Raphson Method on the derivative of the likelihood by choosing a suitable value for x_0.
4.) The root of the derivative found is the required **maximum likelihood estimator** of p.

Python Code:

```python
# PROJECT 3

## Importing Libraries
from sympy import Symbol, diff, lambdify

## Newton Raphson Iterations
def Newt_iter(f, df, p):
    x = p-(f(p)/df(p))
    i = 1
    print("x%d ="%(i), x)
    while x != p:
        i = i+1
        p = x
        x = x-(f(x)/df(x))
        print("x%d ="%(i), x)
    return x

## Finding the derivatives of the likelihood
p = Symbol('p')
l = (p**46)*(1-p**2)**77
l_1 = diff(l, p)
l_2 = diff(l_1, p)

L = lambdify(p, l)
dL = lambdify(p, l_1)
d2L = lambdify(p,l_2)

print("OBJECTIVE : To find p for which L(p) = p\u2074\u2076(1-p\u00b2)\u2077\u2077 is maximum\n")

## Finding the MLE of p using Newton Raphson Method
print("Newton-Raphson iterations :")
x = Newt_iter(dL,d2L,float(input("x0 = ")))
print("\nMaximum likelihood estimator of p =", x)
print("Derivative at MLE =", dL(x))
print("Second derivative at MLE =", d2L(x))
```

Output:

```
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 23:03:10) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: D:\Notes\Numerical Analysis\Projects\Project3_MLEBiostatistics\Project3_MLEBiostatistics.py
OBJECTIVE : To find p for which L(p) = p⁴⁶(1-p²)⁷⁷ is maximum

Newton-Raphson iterations :
x0 = 0.5
x1 = 0.47413793103448276
x2 = 0.47967323094732295
x3 = 0.47958315360494114
x4 = 0.47958315233127197
x5 = 0.47958315233127197

Maximum likelihood estimator of p = 0.47958315233127197
Derivative at MLE = 0.0
Second derivative at MLE = -1.9728557540578027e-21
>>>
```