Taller # 1

Análisis Numérico

Pontificia Universidad JAVERIANA

Presentado a:

Eddy Herrera Daza

Presentado por:

Sergio Posada

Mateo Galvis

11/02/2019

Bogotá D.C

Tabla de Contenido

Nota: Todos los códigos del los respectivos puntos del taller #1 se encuentran en el repositorio.

Punto 1

1.1. Descripción del problema

Evaluar el valor de un polinomio es una tarea que involucra para la máquina realizar un número de operaciones la cual debe ser mínimas. Para cada uno de los siguientes polinomios,

a. Hallar P(x) en el valor indicado y el número mínimo de operaciones para realizarlo

b. Demuestre que el número mínimo de multiplicaciones es n siendo n el grado del polinomio.

- $P(x) = 2x^4 - 3x^2 + 3x - 4 \quad en \ Xo = -2$
- $P(x) = 5x^5 + 6x^4 - 6x^3 + 3x - 4 \quad en \ Xo = 3$
- $P(x) = -5x^6 + 3x^4 + 2x^2 - 4x \quad en \ Xo = -2$

1.2. Solución

Con el fin de encontrar una solución óptima para el problema se utilizó el código que se verá a continuación tomando como base el código del algoritmo Horner el cual sirve para evaluar de forma eficiente funciones polinómicas de una forma monomial. Código adjunto en el repositorio.

1.3. Resultados

Se obtuvieron los siguientes resultados para los polinomios:

Función: $P(x) = 2x^4 - 3x^2 + 3x - 4 \quad en \ Xo = -2$
Resultado: 10 con un número mínimo de operaciones de 4.
Multiplicaciones: 4 teniendo un polinomio de grado 4.

Función: $P(x) = 5x^5 + 6x^4 - 6x^3 + 3x - 4 \quad en \ Xo = 3$
Resultado: 2030 con un número mínimo de operaciones de 5.
Multiplicaciones: 5 teniendo un polinomio de grado 5.

Función: $P(x) \;=- 5x^6 + 3x^4 + 2x^2 - 4x \qquad en \;\; Xo \;=- 2$
Resultado: 4 con un número mínimo de operaciones de 6.
Multiplicaciones: 6 teniendo un polinomio de grado 6.

Punto 2

2.1. Descripción del problema.

Dado un fragmento de código realizar:

A. Recorra el algoritmo con n= 73.
B. Suponga que T(n) representa la cantidad de operaciones aritméticas de división que se realizan para resolver el problema de tamaño n. Encuentre T(n) y exprésela con la notación O(). Para obtener T(n) observe el hecho de que en cada ciclo el valor de n se reduce aproximadamente a la mitad.

2.2. Solución

A. Al correr el algoritmo adjunto en el repositorio con n=73 se obtiene que el número de divisiones necesarias son 7.
B. Se tiene que:

$$T(n) \;=\; 1 \;+\; T(n/2) \qquad\qquad caso\ base: \quad T(1) = 1$$
$$T(n) \;=\; T(n/2) \;+1$$
$$T(n) \;=\; T(n/2^k) + k \qquad k\ es\ el\ número\ de\ iteraciones$$
$$T(n) \;=\; T(n/2^{(log2(n))}) \;+\; log2(n)$$
$$T(n) \;=\; T(1) \;+\; log2(n)$$
$$T(n) \;=\; 1 \;+\; log2(n) \quad aplicando\ el\ caso\ base$$

Según la inducción se evidencia que T(n) es de orden O(log n).

Punto 3

Utilice R y el método de Newton para resolver el problema, muestre gráficamente cómo se comporta la convergencia a la solución.

Ejemplo. Una partícula se mueve en el espacio con el vector de posición **R(t) = (2cos(t), sen(t), 0)**. Se requiere conocer el tiempo en el que el objeto se encuentra más cerca del punto **P(2, 1, 0).** Utilice el método de Newton con cuatro decimales de precisión.

3.1 Explicación

Se aplicó la fórmula de distancia entre dos puntos y un vector obteniendo:

$$R(t) = (\ 2\cos(t)\ , sen(t), 0\ )$$

$$P = (2, 1, 0)$$

$$d(t) = \sqrt{(2 - 2\cos(t)\ ) + (1 - sen(t)\ )}$$

Nota: La distancia se elevó al cuadrado para poder manejar las ecuaciones de una forma más eficiente.

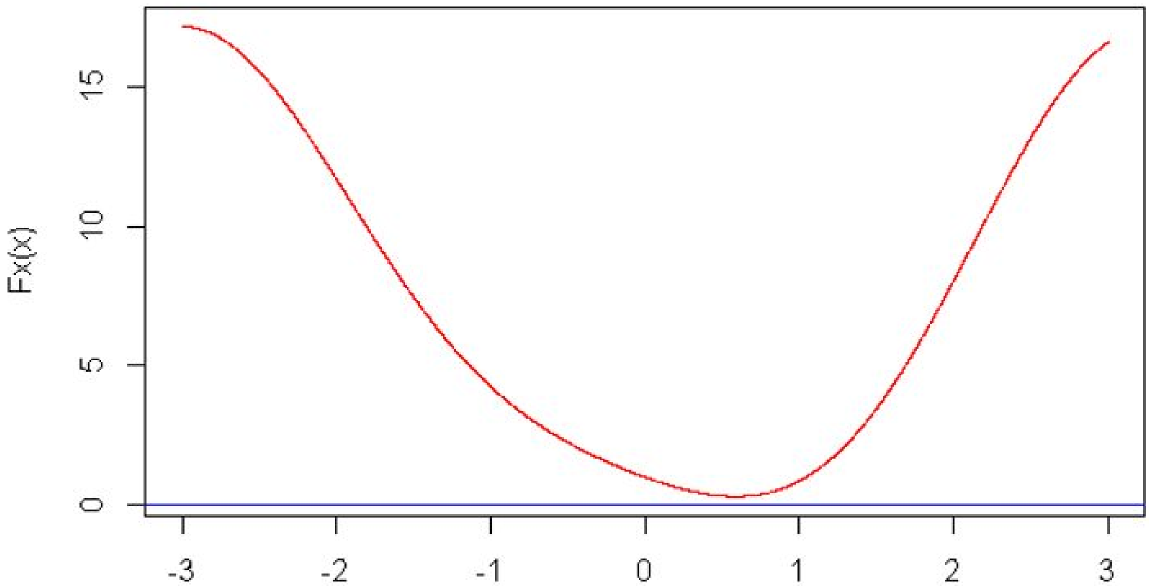
Ecuación #1:

$$d(t)^2 = (2 - 2\cos(t)\ ) + (1 - sen(t)\ )$$

Ecuación# 2: La derivada de d(t)

$$d(t)^2 = (8\sin\ (t) - 3\sin(t)\ \ - 2cos(t)$$

3.2 Gráfico



3.3Resultados

Consiguiendo estos resultados:

```
X= 0.1901352     E= 0.4209628    Iteraciones= 2
X= 0.611098      E= 2.359146     Iteraciones= 3
X= -1.748048     E= 1.106167     Iteraciones= 4
X= -0.64188      E= 0.7723581    Iteraciones= 5
X= 0.130478      E= 0.4411047    Iteraciones= 6
X= 0.5715828     E= 3.701403     Iteraciones= 7
X= 4.272986      E= 1.351403     Iteraciones= 8
X= 5.624389      E= 0.7761369    Iteraciones= 9
X= 6.400526      E= 0.4462677    Iteraciones= 10
X= 6.846794      E= 2.473068     Iteraciones= 11
X= 9.319862      E= 4.842178     Iteraciones= 12
X= 4.477684      E= 1.149347     Iteraciones= 13
X= 5.627031      E= 0.7755574    Iteraciones= 14
X= 6.402588      E= 0.4454426    Iteraciones= 15
X= 6.848031      E= 2.606033     Iteraciones= 16
X= 9.454064      E= 10.7298      Iteraciones= 17
X= -1.275735     E= 0.8894977    Iteraciones= 18
X= -0.386237     E= 0.6914224    Iteraciones= 19
X= 0.3051854     E= 0.406988     Iteraciones= 20
>
```

Donde X= La convergencia del valor, E= El error absoluto y Iteraciones= es el número de ciclos antes del alcanzar el valor deseado.

Punto 4

Solución en R estudio para las siguientes ecuaciones , utilizando dos métodos diferentes, grafique las soluciones y comprare los resultados. Encuentre la intersección de las siguientes ecuaciones en coordenada polares.

4.1 Explicación

Ecuación # 1 :

$$r = 2c + cos(3t)$$

Ecuación # 2:

$$r = 2 - e^{t}$$

Para cada uno de las ecuaciones se utilizó el método gráfico en coordenada polares para su respectiva representación. Se utilizaron 64 puntos empezando desde 0 hasta el 9,6 aumentado en un margen de 0,15 por cada punto. En repositorio de taller# 1 en Github aparecen todos los datos y cómo se obtuvo las gráficas.

Además se encontró el punto de intersección de entre las dos ecuaciones, mediante la siguiente ecuación:

$$cos(3t) + e^{t} = 0$$

Para obtener las respectiva solución a la intersección de las curvas se utilizó dos métodos:

- Bisección

- Secante

4.2 Gráfico

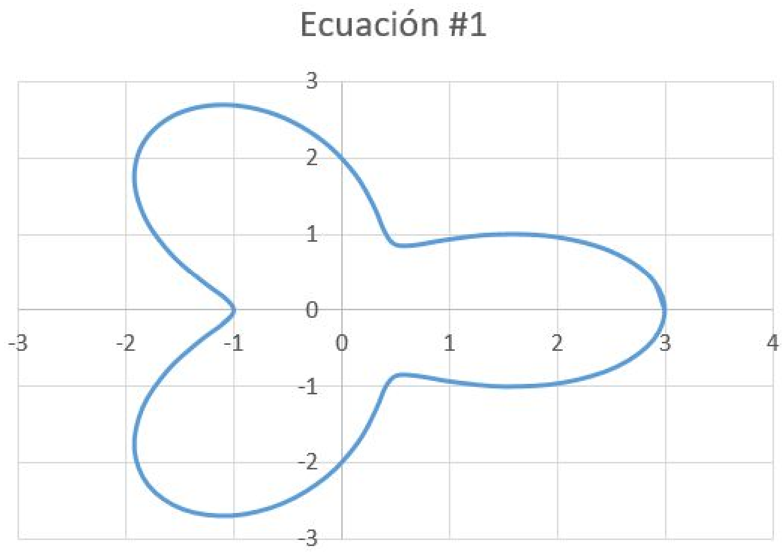


Gráfico de la primera ecuación en coordenadas polares.

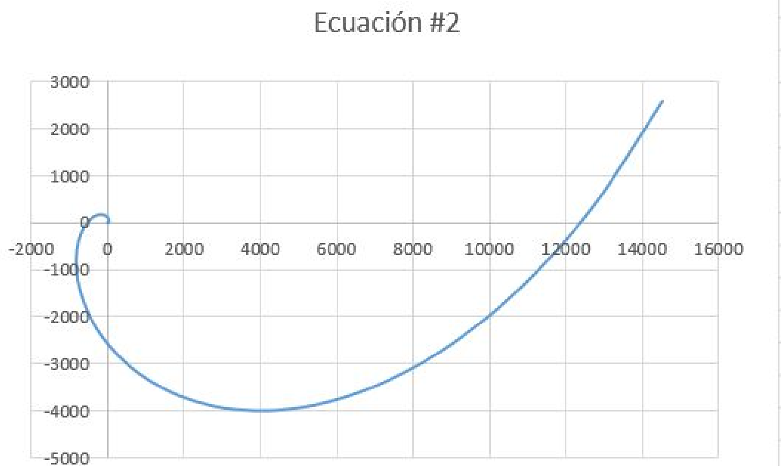


Gráfico de la segunda ecuación en coordenadas polares.

4.5 Resultados

Bisección Resultados:

```
X= 0    E= 3            Iteración= 1
X= -1.5         E= 1.5          Iteración= 2
X= -2.25        E= 0.75                 Iteración= 3
X= -2.625       E= 0.375                Iteración= 4
X= -2.8125      E= 0.1875               Iteración= 5
X= -2.71875     E= 0.09375              Iteración= 6
X= -2.671875    E= 0.046875             Iteración= 7
X= -2.648438    E= 0.0234375            Iteración= 8
X= -2.636719    E= 0.01171875           Iteración= 9
X= -2.642578    E= 0.005859375          Iteración= 10
X= -2.639648    E= 0.002929688          Iteración= 11
X= -2.641113    E= 0.001464844          Iteración= 12
X= -2.641846    E= 0.0007324219                 Iteración= 13
X= -2.641479    E= 0.0003662109                 Iteración= 14
X= -2.641663    E= 0.0001831055                 Iteración= 15
X= -2.641754    E= 9.155273e-05                 Iteración= 16
```

Secante Resultados:

```
X= -0.5738372    E= 0.1170793   Iteración: 1
X= -0.6910792    E= 0.006219456         Iteración: 2
X= -0.6968762    E= 0.0004527942        Iteración: 3
X= -0.6973269    E= 2.238416e-06        Iteración: 4
```

Se puede observar en los resultados que el método de secante resultada ser más eficiente en comparación con el método de bisección en temas de iteración el primero necesito 16 iteración mientras el segundo tan solo 4.

Punto 5

5.1. Descripción del problema

Teniendo en cuenta la elipson de una máquina, encuentre el error de redondeo para x = 0.4.

5.2. Solución

Según el código la base para resolver este problema fue hallar tanto el error de redondeo como el error absoluto mediante las fórmulas específicas.

Punto 6

6.1. Descripción del problema

Encuentre una fórmula iterativa de convergencia cuadrática y defina un intervalo de convergencia apropiado para calcular la raíz n-ésima de un número real. El algoritmo solamente debe incluir operaciones aritméticas elementales.

6.2. Solución

Con el fin de encontrar una solución óptima para este problema nos basamos en el método de Newton que permite obtener la raíz n-ésima de un número. Se tiene como base la siguiente ecuación:

$$f(x) = x^n - a$$

En el código adjunto se muestra el respectivo proceso.

6.3. Resultados

Con la entrada n=4, a=256, x=5, pre=0.0001
Se obtuvo que la raíz n-ésima (n=4) de 256 es 4 con un total de 4 iteraciones

Punto 7

7.1. Descripción del Problema

14. El siguiente es un procedimiento intuitivo para calcular una raíz real positiva de la ecuación **f(x) = 0** en un intervalo **[a, b]** con precisión **E**:

A partir de **x = a** evalúe **f(x)** incrementando **x** en un valor **d**. Inicialmente **d = (b – a)/10**
Cuando **f** cambie de signo, retroceda **x** al punto anterior **x – d**, reduzca **d** al valor **d/10**
y evalúe nuevamente **f** hasta que cambie de signo Repita este procedimiento hasta que **d** sea menor que **E**.

a) De manera formal escriba las condiciones necesarias para que la raíz exista, sea única y pueda ser calculada.
b) Indique el orden de convergencia y estime el factor de convergencia del método.
c) Describa el procedimiento anterior en notación algorítmica, o en MATLAB o en Python

7.2. Solución

a) 1. Determinar un intervalo adecuado mediante la ayuda gráfica o por algún aspecto matemático que caracteriza la función a estudiar.
2. Partir de un punto que esté en el intervalo de [a,b]
3. Para que la raíz se única en el intervalo solo puede haber un cambio de signo

b) Para que exista un grado de convergencia el error absoluto tiene que ir disminuyendo en cada interacción. Hasta llegar a un error tolerable según lo que se esté buscando. En dado caso de que el error aumente se puede afirmar que la función estudiada en el intervalo [a, b] no converge a ningún valor, sino que diverge a infinito.

c) Código en Rstudio:

```
#Función
fx<-function(x)exp(x)-pi

#Main
raiz<-function(a,b,E)
{
  error<-1
  x1<-a
  while(error<E)
  {
    i<-i+1
    d<-(b-a)/10
    x2<-d
    if(f(x1)*fx(x2)<0)
    {
      x2<-x2-d
      d<-d/10
    }
    error<-error(fx1)/error(fx2)
    x1<-x2
    cat("X=",x,"\tE=",error,"\t\tIteración=",i,"\n")
  }


}
raiz(0,1,0.000000001)
```