Taller 1

Karen Celis, Camilo Muñoz, David Lopez

February 11, 2019

1 Primer punto

1.1 Problema

a).Hallar P(x) en el valor indicado y el numero minimo de operaciones a realizarlo, b)Demuestre que el numero minimo de multiplicaciones es n siendo n el grado del polinomio.

1.1.1 Solución

En este ejercicio se evalúa los polinomios con un valor especifico, de igual manera se cuenta la cantidad de operaciones realizadas.Cabe aclarar que se tomó la operación de exponente como múltiples multiplicaciones. Por lo que, por ejemplo, si se tiene x^3 se toma esto como $x * x * x$ siendo esto 2 operaciones (Multiplicaciones)

1.1.2 Solución a

1.1.3 Primer polinomio

$(x) = 2x^4 - 3x^2 + 3x - 4$
$x_0 = -2$

1.1.4 Segundo polinimio

$P(x) = 7x^5 + 6x^4 - 6x^3 + 3x - 4$
$x_0 = 3$

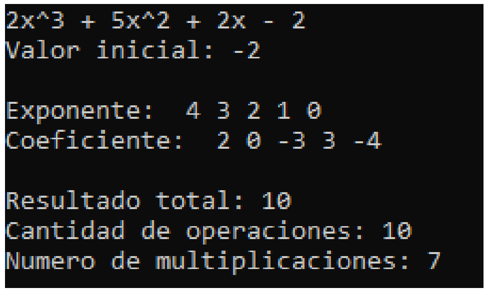

Figure 1: Primer polinomio

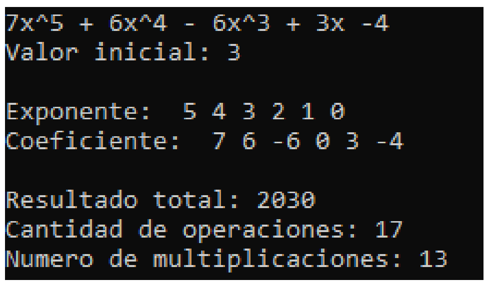

Figure 2: Segundo polinomio

1.1.5 Tercer polinimio

$P(x) = -5x^6 + 3x^4 + 2x^2 - 4x$
$x_0 = -1$

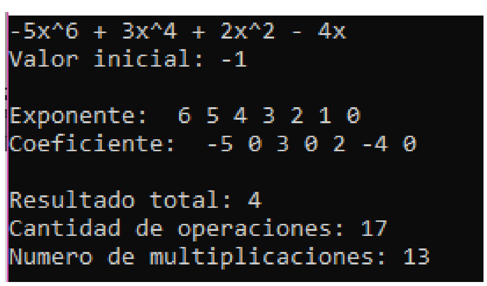

Figure 3: Tercer polinomio

1.1.6 Solución b

n min multiplicaciones = grado polinomio
Debido a que se debe multiplicar la cantidad de veces que el primer exponente (el mayor), entonces se estaría igualando la cantidad de multiplicaciones con el grado del exponente. SI el polinomio tiene mas términos y estos también tienen multiplicación (debido a exponentes o coeficientes) también se suma a la cantidad de multiplicaciones totales del polinomio.
Por lo que siempre será o mayor o igual la cantidad de multiplicaciones que el grado del polinomio.

2 Segundo punto

2.1 Problema

La eficiencia del algoritmo esta denotada por T(n), dado el algoritmo: a)Recorra el algoritmo con n=73, b) Suponga que T(n) representa la cantidad de operaciones aritmeticas de division que se realizan para resolver el problema de tamaño n. Encuentre T(n) y expresela con la notacion o() para obtener T(n) observe el hecho de que en cada ciclo el valor de n se reduce aproximadamente a la mitad.

2.1.1 Solución a

Para este ejercicio se pasa el código a R. Donde se toma un numero 'n', seguido de esto se saca el módulo 2 de 'n' el resultado de esto se guarda en 'd'. Por último, se divide n entre dos y al final se imprime 'd'. Esto se realiza solo hasta que el numero 'n' sea mayor a 0.

Al saber que el código sirve para convertir un numero decimal a un numero binario, se puede saber la cantidad de operaciones realizadas mirando su representación binaria.

2.1.2 Solución b

Se sabe que esta usa una notación de 1 y 0, por lo que al hacer combinatoria con estas dos opciones para poder abarcar todos los números se representaría 2^x. En este caso, como queremos saber la cantidad de operaciones que se realiza lo que desconocemos es x y tenemos el resultado de esto $2^x = 73$. Se realiza la operación contraria para poder hallar x, lo que sería igual a $x = log2(n)$, siendo en este caso n = 73. Quedando $x = log_2(73)$ y dando como resultado x = 6,18. Al no poderse realizar 0,18 de una operación se

```
#Funcion
mod <- function(x,y)
{
  return (x %% y)
}
#Fix
fix <- function(x,y)
{
  return (x %/% y)
}

#Divisiones
divisiones<-function(x)
{
  n<-x
  d<-0
  i<-0
  while(n>0){
    d<-mod(n,2)
    n<-fix(n,2)
    i<-i+1
    cat("d:=",d,"\n")
  }
  cat("Número de iteraciones:",i,"\n")
}

divisiones(73);
```

Figure 4: Codigo Punto 2

```
> divisiones(73);
d:= 1
d:= 0
d:= 0
d:= 1
d:= 0
d:= 0
d:= 1
Número de iteraciones: 7
```

Figure 5: Salida Punto 2

redondea al número superior, dándonos así que la cantidad de operaciones (de división) realizadas para el numero 73 es igual a 7.

$T(n) = log_2(n)$
$O(n) = log(n)$

3 Tercer punto

3.1 Problema

Utilice R y el metodo de Newton para resolver el problema, muestre graficamente como se comporta la convergencia a la solucion.

3.1.1 Solución

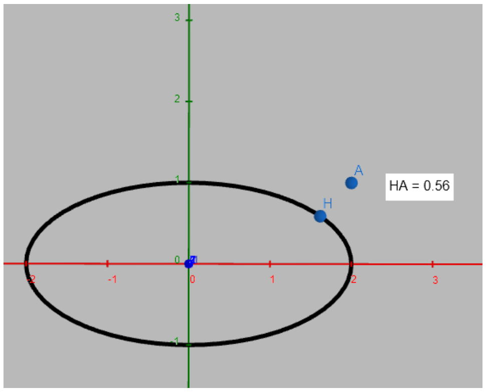

Figure 6: Vista a los ejes ‘x’ y ‘y’

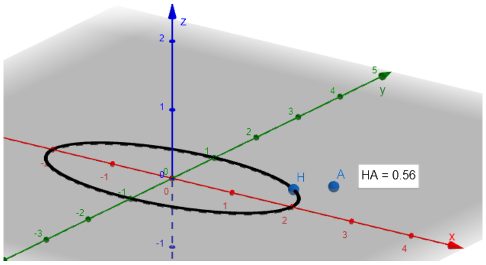

Figure 7: Vista a los ejes ‘x’, ‘y’ y ‘z”

El tiempo en el que la partícula se encuentra más cerca del punto (2,1,0) es en las coordenadas:
R(t) = (X (t), Y (t), Z(t)) = (2cos(t), sen(t), 0)
X (t) = 2cos (0.63) = 1.62
Y (t) = sen (0.63) = 0.59
Z (t) = 0
R (0.63) = (1.62, 0.59, 0)

4 Cuarto punto

4.1 Problema

Solución en R. Para la siguiente ecuación, utilice dos métodos diferentes, grafique las soluciones y comparar. Encuentre una intersección de las siguientes ecuaciones en coordenadas polares.

4.2 Solución

Los métodos usados para este ejercicio fueron el método de bisección y el método de Newton

Figure 8: Grafica Polar

Como se puede apreciar en la imagen anterior, la cual es la gráfica de las dos funciones anteriormente nombradas graficadas en forma polar, acá podemos ver que ambas gráficas tienen dos puntos de intersección los cuales a continuación serán evaluados a desde distintos métodos, por el lado del método de bisección se usará ya que este nos arroja un resultado muy útil como aproximación inicial de otros métodos y por el otro lado tenemos el método de Newton debido a su precisión en los resultados y en su gran uso para ecuaciones no lineales.

Cabe aclarar también que se usó el código entregado por la profesora respectiva del curso para la elaboración de la gráfica polar

5 Quinto punto

5.1 Problema

Teniendo en cuenta lo anterior, encuentre el error de redondeo para x=0.4

5.2 Solución

- ¿Cómo se ajusta un número binario infinito en un número finito de bits? A este caso particular se le denomina la coma flotante donde la computadora que tiene que operar esta número infinito hace lo siguiente y es que después de haber hecho a operación al final se le coloca una e ya bien puede ser minuscula o mayuscula y luego se le pone su respectivo exponente el cual significa que el número está siendo multiplicado por diez a la x. También hay que tener en cuenta que la máquina guarda un espacio de un bit para saber si el número es infinito positivo o negativo.

- ¿Cuál es la diferencia entre redondeo y recorte?

Redondear un número quiere decir reducir el número de cifras manteniendo un valor parecido. El resultado es menos exacto, pero más fácil de usar. Ejemplo: 73 redondeado a la decena más cercana es 70, porque 73 está más cerca de 70 que de 80, este tipo de casos suele pasar mas seguido con los decimales de un número.

El recorte por su lado es sacarle la función al número deseado es decir que solo le estará sacando la porción entera del número, en donde se usa más este tipo de funciones es para la programación con la función floor().

- Indique el número de punto flotante (IEEE) de precisión doble asociado a x, el cual se denota como fl(x); para x(0.4)

Tomando en cuenta el proceso que la computadora usa de la función a piso, es decir sacando la parte entera del número la representación binario de un número queda de la siguiente forma 1001001, pero para que un número de mayor dimensión sea posible representarlo en una máquina este debe pasar por las reglas IEEE 754 para representaciones de número flotante haciendo la partición de la siguiente forma :

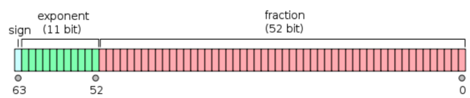

Figure 9: Grafica

Por lo tanto se tiene que segun el estandar IEEE fl(0.4) es en aproximado a (0011111111101100110011001100) en base 2.

- Error de redondeo. En el modelo de la aritmética de la computadora IEEE, el error de redondeo relativo no es más de la mitad del epsilon de máquina

$$\frac{|fl(x) - x|}{|x|} \leq \frac{1}{2}\epsilon_{maq}$$

Figure 10: Grafica

// Encontrar el error de redondeo para x=0.4. Este punto será desarrollado en un programa R.

6 Sexto punto

6.1 Problema

Encontrar una formula iterativa de convergencia cuadratica y defina un intervalo de convergencia apropiado para calcular la raiz real n-esdima de un numero real. Debe incluir solamente operaciones aritmetica elementales.

6.2 Solución

Archivo PuntoSexto.py

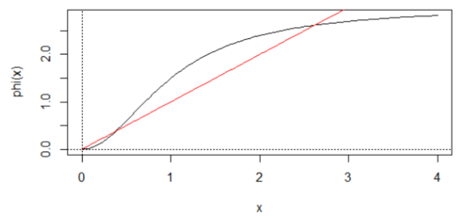

Figure 11: Grafica Sexto Punto

7 Septimo punto

7.1 Problema

a)De manera formal escriba las condiciones necesarias para que la raiz exista, sea unica y pueda ser calculada, b) Indique el orden de convergencia y estime el factor de convergencia del método, c)Describa el procedimineto anterior en notacion algoritmica, o en MATLAB o en Python.

7.2 Solución a

Las condiciones para que exista la raíz en el intervalo (a,b) es que la función evaluada en el valor a multiplicada por la función evaluada en el valor b de un número menor a 0. f(a)*f(b)¡0

7.3 Solución b

El algoritmo se encuentra implementado en el repositorio con nombre SeptimoPunto.py

8 Manual de compilación

Dentro de la carpeta Taller1 se encuentran los codigos de los problemas 1,2,3,4,5,6 y 7.