Taller 1 Análisis Númerico Parte 1

Daniel Reyes
Edwin Turizo
Juan Pimienta
Cristobal Castrillon

February 14 2020

1. Problema 1

Suponga que un dispositivo solo puede almacenar únicamente los cuatro primeros dígitos decimales de cada número real, y trunca los restantes (esto es redondeo inferior). Calcule el error de redondeo si se quiere almacenar el número 536.78.

Solucion:

El error de redondeo se encuentra en el intervalo de: $0{,}1 < 8x10^{-5} < 0{,}1$

Algoritmo realizado en R.

```r
#Error Redondeo en 4 Cifras
errorRedondeo <- function (num, m, n){
  numero<- abs(num)

  while(num>1){
    num<-num/10
    n=n+1
  }
  E<-(num-trunc(num*10^4)/10^4)
  izq<-1*10^(n-m)
  der<-1*10^(n-m)
  cat("E se encuentra en el intervalo: ",izq,"<",E,"<",der)
}
errorRedondeo(536.78,4, 0)
```

2. Problema 2

Implemente en cualquier lenguaje el siguiente algoritmo que sirve para calcular la raíz cuadrada. Aplíquelo para evaluar la raíz cuadrada de 7, analice su precisión, como podría evaluar la convergencia y validez del algoritmo.

Solucion:

El resultado es: 2.645751 con error de 1e-08

Algoritmo realizado en R.

```r
#Metodo Raiz (caso: raiz de 7)
raiz <- function(n, E, x)
{
  y <- 0.5*( x + n / x);
  k<-abs(x-y)

  while(k > E){
    x <- y
    y <- (1/2)*(x+(n/x))
    k <- abs(x-y)
  }
  return(cat("El resultado es: ", y, " con error de ", E))
}
raiz(7,0.00000001, 100)
```

3. Problema 3

Utilizando el teorema de Taylor hallar la aproximación de $e^{0,5}$ con cinco cifras significativas.

Solucion:
El valor aproximado es: 1.6484
Algoritmo realizado en R.

```r
#Metodo Teorema de Taylor
aproximacion=function (n,x){
  suma=1
  i=n-1
  while(i>0){
    suma=1+x*suma/i
    i=i-1
  }
  cat(signif(suma,digits=5))
}
aproximacion(5,0.5)
```

4. Problema 4

Calcule el tamaño del error dado por las operaciones aritméticas, para la solución del siguiente problema

La velocidad de una partícula es constante e igual a **4 m/s**, medida con un error de **0.1 m/s** durante un tiempo de recorrido de **5 seg**. medido con error de **0.1 seg**. Determine el error absoluto y el error relativo en el valor de la distancia recorrida.

$v = 4$, $E_v = 0.1$	(velocidad)
$t = 5$, $E_t = 0.1$	(tiempo)
$d = vt$	(distancia recorrida)

Solucion:
La distancia recorrida es de: 20m

Con un error absoluto de: 0.9
Por lo cual la distancia tiene un rango de variacion: $19{,}1 \leq d \leq 20{,}9$
El error relativo es de: 4.5 %
Algoritmo realizado en Python.

```python
#Metodo de la Distancia con Errores
def distancia(vel,tiempo,errorV,errorT):
  distancia = vel * tiempo
  error_abs = (vel * errorV) + (tiempo * errorT)
  error_rel = ((errorV / vel) + (errorT / tiempo)) * 100
  print("La distancia recorrida es de: " + str(distancia))
  print("Con un error absoluto de: " + str(error_abs))
  print("Por lo cual la distancia tiene un rango de variacion: ")
  print("   " + str(distancia - error_abs) + " <= d <= " + str(distancia + error_abs))
  print("\nEl error relativo es de: " + str(error_rel) + "%")

distancia(5,4,0.1,0.1)
```

5. Problema 5

Evaluar el valor de un polinomio es una tarea que involucra para la maquina realizar un número de operaciones la cual debe ser mínimas. Como se puede evaluar el siguiente polinomio con el número mínimo de multiplicaciones

$$P(x) = 2x^4 - 3x^2 + 3x - 4 \qquad \text{en } x_0 = -2$$

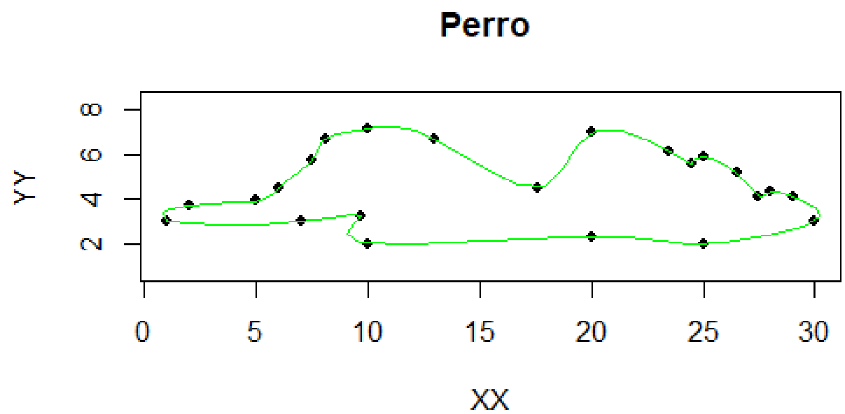
Solucion:
El resultado del polinomio es: 10 con un minimo de 8 operaciones de multiplicacion. Esto se realizo por medio de metodo de Horner. Algoritmo realizado en R.

```r
Horner<- function (func, g, x0){
  res<-func[1]
  n<-0
  for(i in 2:(g+1)){
    res<- res*x0 + func[i]
    n<-n+2
  }
  cat("El resultado del polinomio es: ", res, " en ",n,"operaciones")
}
func<-c(2,0,-3,3,-4)
Horner(func,4,-2)
```

6. Problema 6

Reconstruir la silueta del perrito utilizando la menor cantidad de puntos para reproducir el dibujo del contorno completo del perrito sin bigotes, con la información dada
Solucion:
Para la reconstruccion de la silueta del perro se utilizo una funcion de pintar en la cual se utilizo un spline que es una curva diferenciable definida en porciones mediante puntos que forman polinomios. Esto realizando tambien una interpolación de los datos en las parejas

(x,y) de cada punto que define la figura del perro. Para la realizacion de la interpolacion se añadieron 6 puntos mas de los que ya se tenian definidos.
Algoritmo realizado en R.




```
library(polynom)
library(PolynomF)

yy=c(3,3.7,3.9,4.5,5.7,6.69,7.12,6.7,4.45,7,6.1,5.6,5.87,5.15,4.1,4.3,4.1,3,2,2.3,2,3.2,3,3)
xx=c(1,2,5,6,7.5,8.1,10,13,17.6,20,23.5,24.5,25,26.5,27.5,28,29,30,25,20,10,9.7,7,1)
plot(xx,yy, pch=19, cex=0.9, col = "BLACK", asp=1,xlab="XX", ylab="YY", main="Perro ")
n=24
pint<-function(xx,yy){
  t = 1:length(xx)
  sx = spline(t,xx)
  sy = spline(t,yy)
  lines(sx[[2]],sy[[2]],type='l', col="GREEN")
}
pint(xx,yy)
```