Laboratorio 1

Gonzalo Tapia | 201073566-4 | gonras@gmail.com
Gabriel Camilo | 2904216-0 | gabriel.camilo@alumnos.usm.cl
Universidad Técnica Federico Santa María
Computación Cientifica 1
ILI-285

30 de marzo de 2016

1. Introducción

El objetivo principal del presente informe es dar a conocer los resultados obtenidos al realizar diversos cálculos matemáticos, cuyo resultado se encuentra sujeto a variaciones debido a la precisión del computador al utilizar notación de punto flotante IEEE.
Por ultimo agregar que para realizar el laboratorio asociado al informe se utilizó como lenguaje Python y tres librerías adicionales que son *Numpy*, *Decimal*, *Math*

2. Desarrollo y análisis de resultados.

2.1. Punto flotante

2.1.1. a

En esta parte se importa la librería a utilizar

```
import numpy
```

Se define la función que retornará el épsilon machine

```
def machineEpsilon(precision):
```

Si la precisión es simple se utiliza numpy.float32

```
        if(precision=="simple"):
                        funcion=numpy.float32
```

En este caso si la precisión en double se usa numpy.float64

```
        else:
                        funcion=numpy.float64
```

Se define el número 1 en la precisión necesaria

```
        e_mach = funcion(1)
```

Ciclo que se encarga de funcionar mientras no encuentre el número mas pequeño representable para la precisión necesaria

```
        while funcion(1)+funcion(e_mach) != funcion(1):
```

Se define un nuevo valor que almacenara cada valor para luego retornarlo al terminar el ciclo

```
                        e_mach2 = e_mach
```

Se define nuevamente épsilon machine como el anterior épsilon machine dividido por 2, ambos representados en la precisión seleccionada

```
                        e_mach = funcion(e_mach) / funcion(2)
```

Se retorna el épsilon machine encontrado dentro del ciclo, que corresponde al mas pequeño representable por la precisión

```
        return e_mach2
```

Se imprime el valor encontrado en la función para la precisión simple

```
print "epsilon machine single presicion: "+str(machineEpsilon("simple"))
```

Se imprime el valor encontrado en la función para la precisión double

```
print "epsilon machine doble presicion: "+str(machineEpsilon("double"))
```

2.1.2. b

Precisión simple:
Distancia 1:$d = 75290{,}2771402$
Distancia 2:$d = 77837{,}6752064$
Distancia 3:$d = 302{,}488965782$
Distancia 4:$d = 59060{,}2136106$

Precisión double:
Distancia 1:$d = 75290{,}2808467$
Distancia 2:$d = 77837{,}6752193$
Distancia 3:$d = 302{,}488971549$
Distancia 4:$d = 59060{,}2136106$

Diferencias entre las representaciones para cada distancia:
Diferencia 1:$diff = |75290{,}2771402 - 75290{,}2808467| = 0{,}0037$
Diferencia 2:$diff = |77837{,}6752064 - 77837{,}6752193| = 0{,}00001$
Diferencia 3:$diff = |302{,}488965782 - 302{,}488971549| = 0{,}0000058$
Diferencia 4:$diff = |59060{,}2136106 - 59060{,}2136106| = 0$

Como se puede apreciar existe una diferencia entre las distancias dependiendo de que precisión se utilice.

Valor real de las distancias:
Siendo los valores reales de las distancias:
$Real1 = 75290{,}28085$
$Real2 = 77837{,}67522$
$Real3 = 302{,}4889715$
$Real4 = 59060{,}21361$

Precisión simple:
Diferencia 1:$diff = |75290{,}2771402 - 75290{,}28085| = 0{,}00371$
Diferencia 2:$diff = |77837{,}6752064 - 77837{,}67522| = 0{,}00002$
Diferencia 3:$diff = |302{,}488965782 - 302{,}4889715| = 0{,}0000058$

Diferencia 4:$diff = |59060{,}2136106 - 59060{,}21361| = 0$

Precisión double:
Diferencia 1:$diff = |75290{,}2808467 - 75290{,}28085| = 0{,}00001$
Diferencia 2:$diff = |77837{,}6752193 - 77837{,}67522| = 0{,}00001$
Diferencia 3:$diff = |302{,}488971549 - 302{,}4889715| = 0$
Diferencia 4:$diff = |59060{,}2136106 - 59060{,}21361| = 0$

Estas diferencias se deben a que para el cálculo de las distancias se utilizó una calculadora científica que es más precisa que la precisión simple pero menos precisa que la precisión double, esto se puede apreciar al ver que los valores de las diferencias entre el valor real y la precisión respectiva son menores para el caso de la precisión double.

2.1.3. c

$$\mathrm{i})(3-2) + 2^{-54} = 1 \tag{1}$$

$$\mathrm{ii})3 - (2 - 2^{-54}) = 1{,}00000000000000055511151231 \tag{2}$$

En la parte (i) del problema podemos notar que 2^{-54} es menor que el ϵ_{mach} por lo que al sumar $(3-2) + 2^{-54}$, la cifra significativa de la parte 2^{-54} queda ubicada en una posición fuera de la mantisa del resultado. Debido a esto el valor se traduce en sumar algo tan pequeño que no es representable por el computador y entregando 1.0 como resultado. Por otro lado en la parte (ii) del problema el primer paréntesis que se debe resolver da como resultado $2 - \epsilon_{mach}$, esto al realizar la sustracción de forma manual [1]. Y luego al realizar la diferencia entre 3 y el resultado obtenido se debiese obtener $1 + \epsilon_{mach}$.

$$\mathrm{i})\sin(2^{-54}) = 5{,}5511151231257827021181583404541015625E - 17 \tag{3}$$

$$\mathrm{ii})2\sin(2^{-53})cos(2^{-53}) = 2{,}220446049250313080847263336E - 16 \tag{4}$$

El resultado no con coincide debido a que no son iguales aritméticamente. Además la representación de números tan pequeños está sujeta a aproximaciones debido al tamaño limitado de la mantisa.

3. Conclusiones

Podemos dar cuenta que al realizar un cálculo aritmético mediante el computador no siempre obtenemos resultado exacto.En especial, esta diferencia, se hace notar cuando los números son pequeños y se necesita una gran cantidad de exactitud en los cálculos. Pese a esto el mejor modo para encontrar un buen resultado es comprendiendo la razón de la diferencia y encontrando una solución en base a esto

[1] $resultado = 1\ 01111111111\ 1111111111111111111111111111111111111111111111111111$

4. Anexo

A continuación se presentan los códigos usados:
Código a:

```
import numpy
def machineEpsilon(precision):
    if(precision=="simple"):
        funcion=numpy.float32
    else:
        funcion=numpy.float64

    e_mach = funcion(1)
    while funcion(1)+funcion(e_mach) != funcion(1):
        e_mach2 = e_mach
        e_mach = funcion(e_mach) / funcion(2)
    return e_mach2

print "epsilon machine single presicion: "+str(machineEpsilon("simple"))
print "epsilon machine doble presicion: "+str(machineEpsilon("double"))
```

Código b:

```
import numpy

def distancia_double(x,y,z):
    aux=numpy.float64(0)
    return ((x-aux)*(x-aux)+(y-aux)*(y-aux)+(z-aux)*(z-aux))**(0.5)
def distancia_simple(x,y,z):
    aux=numpy.float32(0)
    return ((x-aux)*(x-aux)+(y-aux)*(y-aux)+(z-aux)*(z-aux))**(0.5)

funcion64=numpy.float64
funcion32=numpy.float32
print "distancia precision simple: "+str(distancia_double(funcion32(9.4**5),funcio
print "distancia precision simple: "+str(distancia_double(funcion32(0.2**5),funcio
print "distancia precision simple: "+str(distancia_double(funcion32(0.6**5),funcio
print "distancia precision simple: "+str(distancia_double(funcion32(9**5),funcion3
```

Código c:

```
from decimal import Decimal
import math

x=(3-2)
print ('resultado 1.(i)')
```

```
print Decimal(x+2**-54)


x=Decimal(2**-54)
y=Decimal(2-x)
z=Decimal(3)
print ('Resultado 1.(ii)')
print Decimal(z-y)

#---------------------------------------
print ('-----------o-----------')
x=Decimal(2**-54)
i=Decimal(math.sin(x))
print ('Resultado 2.(i)')
print i
y=Decimal(2**-53)
sin=Decimal(math.sin(y))
cos=Decimal(math.cos(y))
print ('Resultado 2.(ii)')
ii=Decimal(Decimal(2)*sin*cos)
print ii
```

5. Bibliografía

1. Numerical analysis 2nd edition timothy sauer
2. http://profesores.elo.utfsm.cl/ tarredondo/info/comp-architecture/apuntes-lsb/cap9a.pdf "UNIVERSIDAD TECNICA FEDERICO SANTA MARIA, DEPARTAMENTO DE ELECTRONICA, ELO311 Estructuras de Computadores"