

# ANÁLISIS NUMÉRICO I – 2017

## Trabajo de Laboratorio N<sup>o</sup> 1

1. Dados dos vectores  $v \in \mathbb{R}^2$  y  $u \in \mathbb{R}^4$  y las matrices  $A \in \mathbb{R}^{2 \times 2}$  y  $B \in \mathbb{R}^{2 \times 3}$  realizar las siguientes operaciones en Octave:

- |                             |                                |
|-----------------------------|--------------------------------|
| (a) <code>pi*v</code>       | (f) <code>A*v</code>           |
| (b) <code>v'*v</code>       | (g) <code>A^2</code>           |
| (c) <code>sqrt(v'*v)</code> | (h) <code>A.*A</code>          |
| (d) <code>v*u'</code>       | (i) <code>A*B</code>           |
| (e) <code>v.*u(2:3)</code>  | (j) <code>A.*B(1:2,2:3)</code> |

Dadas 3 variables numéricas  $\mathbf{x}$ ,  $\mathbf{y}$ ,  $\mathbf{z}$ , notar las diferencias entre  $\mathbf{x}/\mathbf{y} + \mathbf{z}$  y  $\mathbf{x}/(\mathbf{y} + \mathbf{z})$ , y entre  $\mathbf{x}/\mathbf{y}*\mathbf{z}$  y  $\mathbf{x}/(\mathbf{y}*\mathbf{z})$ .

2. Comprobar que el épsilon-máquina es  $2^{-52} = 2.2204 \times 10^{-16}$ , escribiendo en la línea de comandos:

```
>> a = 1 + 2^(-53); b = a-1
```

y comparando con

```
>> a = 1 + 2^(-52); b = a-1
```

3. Obtener el mayor y menor número positivo en punto flotante (*overflow* y *underflow*). Para obtener el mayor número de overflow escribir un ciclo que vaya calculando las sucesivas potencias de 2 y que finalice cuando se produce overflow. Se recomienda utilizar el comando `isinf` para detectar cuando se produce el overflow (escribir `help isinf` para obtener información sobre este comando). Otra instrucción que puede resultar útil es `break` para interrumpir el ciclo cuando se produce el overflow. El número de underflow se puede obtener dividiendo por 2 repetidamente hasta obtener un número indistinguible del cero en punto flotante.
4. Escribir la siguiente secuencia de comandos en un archivo con extensión `.m` y ejecutarlo en Octave.

```
x = 0;
while x~=10
    x = x + 0.1
end
```

Para interrumpir la ejecución, pulsar CTRL-C. ¿Qué ocurre si en lugar de incrementarse la variable en 0.1 lo hace en 0.5? ¿Por qué?

5.
  - Escriba un programa (**fac.m**) que calcule el factorial de 6.
  - Utilice el comando **lookfor** para buscar comandos relacionados con la expresión **fact**.
  - Escriba una función (**facf.m**) que calcule el factorial de un número  $n$  dado.
6. Notar que Octave es eficiente operando con matrices y vectores, y no elemento a elemento. Para ello tomar un valor grande de  $N$  y comparar:

```
>tic, S=0; for k=1:N, S=S+1/k; end, S, toc
>tic, S=sum(1./(1:N)), toc
```

7. Escribir un programa que pida dos números reales e imprima en la pantalla el mayor de ellos. El programa debe indicar si los números son iguales.
8. Escribir una función que calcule la potencia  $n$ -ésima de un número, es decir que devuelva  $x^n$  para  $x$  real y  $n$  entero. Realice un programa que utilice la función e imprima en pantalla las primeras 5 potencias naturales de un número ingresado.
9. Escribir dos funciones en Octave para la resolución de ecuaciones de segundo grado  $ax^2 + bx + c = 0$ . Una de ellas, que llamaremos de **mala.m**, implementando la tradicional fórmula de Baskhara y la otra, que llamaremos de **buena.m**, usando una manera eficiente para evitar cancelación de dígitos significativos. La sintaxis de la llamada a las funciones debería ser:

```
[x_1, x_2] = buena(a,b,c)
```

y análogamente para **mala.m**.

10. Escribir una función que implemente el algoritmo de Horner (**horn.m**) para la evaluación de polinomios. La sintaxis de llamada a la rutina debería ser:

```
> p = horn(coefs,x)
```

donde **p** es el valor del polinomio, **coefs** es un vector con los coeficientes del polinomio, de mayor a menor grado y **x** es el valor de la variable independiente. Es decir que si, por ejemplo, hacemos:

```
> p = horn([1 -5 6 2], 2)
```

entonces la variable **p** almacenará el valor  $p(2)$  donde  $p(x) = x^3 - 5x^2 + 6x + 2$ .