UNIVERSIDAD DE MURCIA

Departamento de Matemáticas
Cálculo Numérico en una Variable
Curso 2021/22

Prof. S. Sánchez-Pedreño

Práctica 1

19 de septiembre de 2021

Conviene leer con atención el documento `CN1V_OctaveIntroduccion.pdf` incluido en los recursos del Aula Virtual antes de abordar la realización de esta práctica.

1. La función de Octave proporcionada `redondeo.m` (incluida en `/Biblioteca`) devuelve el valor de su argumento redondeado al número de cifras decimales indicado por la variable global `ndig`. Estudia el código de la función:

   a) `mat2str(x,n)` convierte el valor de x en una cadena de caracteres, redondeando x a n dígitos de precisión.

   b) `eval(cad)` ejecuta la cadena `cad` como si fuera código de Octave. El resultado en el ejemplo `redondeo.m` es un valor numérico en coma flotante.

   Crea un archivo script en el que se calculen los resultados de las operaciones que siguen con cuatro cifras decimales (redondeo el que utiliza Octave por defecto):

   a) $0.6688 \oplus 0.3334$;

   b) $1000 \ominus 0.05001$;

   c) $2.000 \otimes 0.6667$;

   d) $25.00 \oslash 16.00$.

   Calcula en cada caso los errores absolutos y relativos cometidos.

2. Consideremos la ecuación de segundo grado

   $$1.002\, x^2 - 11.01\, x + 0.01265 = 0$$

   Trabajando con la función `redondeo`, calcula las soluciones de la ecuación realizando todos los cálculos con cuatro cifras significativas de las dos formas detalladas a continuación. Para una ecuación $ax^2 + bx + c = 0$ las raíces se pueden calcular:

   a) mediante las expresiones bien conocidas $\frac{-b\pm\sqrt{b^2-4ac}}{2a}$, y

   b) calculando una raíz x_1 de forma que se evite la pérdida de cifras significativas y teniendo en cuenta entonces que las raíces verifican $x_1 x_2 = \frac{c}{a}$.

   Para hacernos una idea de la precisión de ambos cálculos vamos a reconstruir el polinomio original utilizando las dos funciones siguientes.

   - La función `poly[x1,x2,...,xn]` de Octave devuelve un vector compuesto por los coeficientes del polinomio cuyas raíces son `x1,x2,...,xn`, los coeficientes están ordenados de forma decreciente, es decir, desde el coeficiente del término de mayor grado, hasta el término independiente.

- La función `polyout(v,'x')` de Octave, devuelve una cadena igual a la expresión del polinomio en la variable `x` que tiene los coeficientes incluidos en el vector `v`.

Realiza los mismos cálculos con 2 y 8 cifras significativas.

3. De acuerdo con el ejercicio 1.37 (con las precauciones sugeridas por el apartado *c*) del mismo ejercicio) parece que será más preciso realizar una suma de términos positivos de forma que el tamaño de los sumandos sea creciente. Vamos a realizar la experiencia con una suma bien conocida:

$$\sum_{k=1}^{\infty} \frac{1}{k^2} = \frac{\pi^2}{6}$$

El objetivo es calcular las sumas

$$\sum_{k=1}^{n} \frac{1}{k^2}$$

para distintos valores de n en orden creciente y decreciente y comparar los resultados. Entonces:

a) Calcula la suma en orden creciente y decreciente de sumandos para n igual a 50, 100, 500, 1000, 10000, 100000. Haz una tabla con los valores de n, los dos valores de la suma y los respectivos errores relativos (usando el valor exacto $\pi^2/6$).

b) Haz lo mismo en precisión doble.

c) Repite el resultado utilizando la función `redondeo` con cinco cifras decimales (evita en principio el caso $n = 100000$ en este apartado, porque es muy lento).

d) Una vez hayas depurado todos los apartados anteriores introduce los comandos `tic` y `toc` que permiten controlar el tiempo de ejecución: `tic` inicia el cronómetro, `toc` lo detiene. Puedes asignar el valor de `toc` a una variable (por ejemplo, `t=toc`), y así mostrar el tiempo de ejecución de un proceso en el formato que desees.

Observaciones.

a) Los bucles en Octave los puedes escribir de varias formas

```
for k=1:n        for k=n:-2:1    for k=[n1,n2,...,np]
...              ...             ...
endfor           endfor          endfor
```

b) Una idea para obtener tablas en la salida de Octave es utilizar especificaciones de formato en la orden de escritura. Por ejemplo:

`printf(form,x1,x2,x3)`

escribirá las variables `x1`, `x2` y `x3` con el formato indicado por la especificación `form`, que es una variable de tipo cadena y que debemos haber definido previamente, por ejemplo, como:

`form='%12.7e,%16.8f,%6u';`

lo que significa que: la primera variable se imprimirá ocupando el ancho de 12 caracteres con 7 cifras decimales en notación exponencial, la segunda ocupará el ancho de 16 caracteres con 8 decimales y la tercera será un entero sin signo ocupando el ancho de 6 caracteres.

4. De acuerdo con el ejercicio 1.22 la sucesión

$$x_n = \int_0^1 x^n e^x \, dx$$

converge a 0, y se puede obtener con la recurrencia: $x_0 = e - 1$, $x_n = e - n x_{n-1}$. Implementa un programa que calcule los 25 primeros valores de la sucesión. Estudia los resultados. Comprueba que es un proceso inestable e intenta explicar el porqué.

5. **Para pensar individualmente en casa (si no hay tiempo en la sesión de prácticas).**

   Después del ejercicio 2 de esta práctica es sencillo realizar la siguiente tarea: implementa una función

   `function [r1,r2]=solve2grado([a,b,c],prec)`

   que reciba como dato el vector `[a,b,c]` con los coeficientes de la ecuación $ax^2 + bx + c = 0$ y devuelva las dos raíces de dicha ecuación. El argumento `prec` será el número de dígitos con los que se realizarán todos los cálculos (recurriendo a la función `redondeo`) y adoptamos el criterio de que si `prec` es 0 entonces los cálculos se realizan en doble precisión.

   El programa debe chequear que se pasen efectivamente tres parámetros (a, b y c) y que si a es nulo emita un mensaje de error, devolviendo la solución de la ecuación lineal que queda, si existe.

   Además debe realizar los cálculos de forma que evite la pérdida de cifras significativas.

   Una vez implementado, chequéalo con los datos del ejercicio 2 y con otros polinomios para los que conozcas con antelación las raíces.

Departamento de Matemáticas. Campus de Espinardo. 30100 Espinardo Murcia
e-mail: pedreno@um.es