Cálculo numérico Trabajo práctico Nro 1

Los ejercicios que se listan a continuación no requieren conocimiento de ningún método numérico en particular. Muchos de ellos solo requieren conocer una definición y un rato con una calculadora científica. En algunos casos se muestran algunas líneas de programa (en línea de comando) en *Matlab*. En muchos casos son ejemplos que ilustran resultados relativos al almacenamiento de números en la computadora y los errores asociados al operar con ellos. En otros se comentan algunos conceptos relacionados con la naturaleza de algunos problemas (ser mal condicionados) y de algunos algoritmos (que puedan ser inestables) para resolver problemas. Los ejercicios marcados con $\star$ son especialmente recomendados.

1. Representación de números en punto flotante

1.1. $\star$ La representación en punto flotante de doble precisión. La representación de números en base 2 habitual en una computadora es la de punto flotante en *doble precisión*. En este formato los números se almacenan en 64 bits de acuerdo al standard IEEE754.
Por ejemplo los 64 bits con los que se representa una aproximación racional al número π son:

`0100000000001001001000011111101101010100010001000010110100011000`

De estos 64 bits el primero es de signo (0= positivo, 1 = negativo). Los siguientes 11 bits son las representacion binaria del exponente `expo` de 2 pero incrementado en 1023. Los restantes 52 bits son la representación binaria de un numero de la forma $m = 0{,}1\ldots$ que se denomina *mantisa* que resulta, entonces un número entre 0 y 1 ($0 \leq m < 1$).
Como se puede modificar la potencia de 2 a la vez que se corre un lugar la coma (binaria) sin modificar el número (de allí lo de *punto flotante*) entonces siempre puede lograrse que el primer bit de la parte decimal pueda ser 1.
Lo que se establece en este formato IEEE754 es que el número (en cuanto a cifras significativas se refiere) es de la forma $1.\ldots$ y ese primer 1 no se almacena (es una cifra ganada que no se almacena). El equivalente en decimal de un número almacenado con este formato es:

$$(-1)^{\text{bit de signo}}\, 2^{\text{expo}-1023}\,(1+m)$$

El siguiente código de *Matlab* convierte a decimal la representación en el formato bit de π.

```
format long
x=[0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 1 0 0 0 0 1 1 1 1 1 1 0 1 1];
x=[x 0 1 0 1 0 1 0 0 0 1 0 0 0 1 0 0 0 0 1 0 1 1 0 1 0 0 0 1 1 0 0 0];
% el vector x guarda la secuencia de unos y ceros
% obtenida a partir de pi en formato bit
signo=(-1)^x(1); % vale 1 ó -1 dependiendo del valor del primer bit
expo=0;
for k=2:12
            if x(k)~=0
                        b=12-k;
                       expo=expo+2^(b);
```

```
            end
end
expo=expo-1023; % el exponente se almacena en exceso 1023
mant=1; % la mantisa es de la forma 1. (y el 1 no se almacena)
for k=13:64
            a=12-k;
            if x(k)~=0
                        mant=mant+2^(a);
            end
end
signo*2^expo*mant
```

1. Ejecutar el código mostrado y obtener la aproximación racional de π en este formato de doble precisión. Calcular el error relativo con que se representa π teniendo en cuenta que estos primeros decimales de π son correctos: 3.14159265358979323846264 3.
2. Dado que cada número se representa con un número finito de bits entonces existe un número máximo positivo posible de representar así como un menor número positivo. En la línea de comando de *Matlab* pruebe esta secuencia:

   ```
   format long
   realmax
   realmin
   2*realmax
   ```

   Interprete lo observado respecto de `realmax` y `2*realmax`. Anote los resultados obtenidos al ejecutar `Inf + 1`, `1/Inf` , `1/0` y `Inf - Inf`.
3. Otro parámetro de interés asociado a esta representación es `eps`. En *Matlab* es una constante predefinida:

   ```
   » eps

   eps =

       2.220446049250313e-016
   ```

   Este número mide la distancia entre 1 y el próximo número representable en el sistema de punto flotante en doble precisión. Se verifica $eps = 2^{-t}$ donde t es el número de bits de la mantisa que en doble precisión es 52. El valor de `eps` es una cota al error relativo de representación de un número en el sistema de representación de punto flotante.

   ```
   » 2^(-52)

   ans =

       2.220446049250313e-016
   ```

   Ensaye la ejecución de estas operaciones en la línea de comando de *Matlab*:

   ```
   a=1;b=1+eps;c=1+eps/2;
   ```

```
b-a
c-a
```

Verifique que `realmax` es igual $(2 - eps)\, 2^{1023}$.

4. Obtenga el valor de *eps* para su calculadora de mano. Para ello realice la operación $(1 + 2^{-t}) - 1$ para t tomando los valores $1, 2, \ldots$ hasta que el resultado sea cero. En una calculadora tipo Casio FX82 el valor de `eps` es 2^{-34}.

1.2. Los números menor positivo, mayor positivo y el ϵ en Matlab: Los siguientes códigos permiten obtener el menor número positivo, el mayor número positivo y el ϵ (la diferencia entre el primer número representable posterior a 1 y 1). Analizar los códigos, si es posible correrlos y así obtener para su entorno de programación esos tres números. No es complicado realizar programas de este estilo en cualquier otro lenguaje de programación.

```
format long
% el mas chico
y=1; while y>0
    maschico=y;
        y=y/2;
end
maschico

format long
% el mas grande
y=1; while y<inf
    masgrande=y;
    y=2*y;
end
masgrande

% epsilon de la maquina
x=1;ex=1; while ex>0
    epsilon=ex;
    x=x/2;
    ex=x*0.98+1;
    ex=ex-1;
end
epsilon
```

En Matlab *eps* es una constante del sistema cuyo valor es:
eps = 2.220446049250313e-016 = 2^{-52}, ya que la mantisa tiene 53 bits.

2. Aritmética en punto flotante

2.1. $\star$ Operaciones en punto flotante. Sea x un número real y $F(x)$ su representación en punto flotante con t dígitos representativos y suponiendo base 10. Por ejemplo si $x = \frac{2}{9}$ y $t = 4$ entonces $F(\frac{2}{9}) = 0.2222$.
Las operaciones de suma, diferencia, producto y cociente están definidas de

la siguiente manera:

$$\begin{aligned} x \oplus y = &\quad F(F(x) + F(y)) \\ x \ominus y = &\quad F(F(x) - F(y)) \\ x \otimes y = &\quad F(F(x).F(y)) \\ x \oslash y = &\quad F(F(x)/F(y)) \end{aligned}$$

1. Calcular los errores absoluto y relativo para las cuatro operaciones indicadas si $x = \frac{1}{3}$ e $y = \frac{4}{7}$. Suponer redondeo por corte o truncado.

operación	respuesta	valor exacto	error absoluto	error relativo
$x \oplus y$				
$x \ominus y$				
$x \otimes y$				
$x \oslash y$				

   Cuando dos números de punto flotante son sumados , el número de la mantisa con el exponente menor se modifica de manera tal que los exponentes sean iguales. Si queremos sumar $0.1557\,10^1 + 0.4381\,10^{-1}$ hay que transformar el segundo sumando en $0.004381\,10^1$. Al sumar y redondear resulta $0.1600\,10^1$.Es interesante observar la información perdida del segundo sumando.
2. Si $a = 0.5711, \quad b = 4271, \quad c = 0.1001\,10^{-3}$ e $y = \frac{4}{7}$ calcular $a \oplus b$ y $(a \ominus y) \oslash c$ y obtener los errores absoluto y relativo.
3. Comparar los resultados de $(121 - 0.3271) - 119$ y $(121 - 119) - 0.3271$. Se supone que se opera resolviendo lo indicado entre paréntesis primero. Considerar truncado y $t = 4$.

2.2. El error de redondeo al resolver un sistema de ecuaciones lineales.

1. El sistema lineal de 2 por 2

$$\begin{aligned} ax + by &= e \\ cx + dy &= f \end{aligned}$$

   se puede resolver para x, y de la siguiente manera:
   a) Sea $m = \frac{c}{a} \quad a \neq 0$
   b) $d_1 = d - mb$
   c) $f_1 = f - me$
   d) $y = \frac{f_1}{d_1}$
   e) $x = \frac{e - by}{a}$

   Usar este procedimiento y la aritmética de redondeo de cuatro dígitos para resolver los sistemas lineales:

$$\begin{aligned} 1.130\,x - 6.990\,y &= 14.200 \\ 1.013\,x - 6.099\,y &= 14.220 \end{aligned}$$

$$\begin{aligned} 8.110\,x + 12.200\,y &= \text{-0.1370} \\ \text{-18.110}\,x + 112.200\,y &= \text{-0.1376} \end{aligned}$$

En cada caso resolverlo también con aritmética exacta y calcular el error relativo.

2. Usando aritmética de punto flotante con tres dígitos significativos resolver:

$$\begin{aligned} x + 400\,y &= 801 \\ 200\,x + 200\,y &= 600 \end{aligned}$$

usando redondeo simétrico.

Sugerencia: Resolver primero eliminando x al multiplicar la primera ecuación por 200 y luego restarlas. Obtener $y = 1.99$ y luego de reemplazar en la primera resulta $x = 5$. Calcular el error relativo. Para obtener la solución del sistema con error relativo cero aún cuando se utilice la aritmética de tres dígitos entonces conviene hacer un escalamiento. Dividir ambas ecuaciones por 1000. Luego de este escalamiento dividir la segunda por 200 y eliminar x para obtener con aritmética de tres dígitos $y = 2.00$ y luego $x = 1.00$ (error relativo cero).

2.3. ¿Es conmutativa la suma en punto flotante ? Usando aritmética de tres dígitos y truncamiento obtener la suma de las recíprocas de los cuadrados de los primeros diez números naturales primero mediante:

$$\sum_{i=1}^{10} \frac{1}{i^2} = 1 + \frac{1}{4} + \frac{1}{9} + \ldots + \frac{1}{100}$$

y luego invirtiendo el orden

$$\frac{1}{100} + \frac{1}{81} + \ldots + 1$$

¿Cuál de los dos métodos es más exacto y por qué?. Observar que la suma se va haciendo tomando de a pares de sumandos y se trunca. Luego se acumula sobre el anterior usando el próximo término y así sucesivamente.

Suponga el siguiente código en Matlab (fácilmente trasladable a otro lenguaje de programación de alto nivel).

```
format long e
sum=1;
   for i=1:10000
    sum=sum+0.00001;
   end
sum
```

¿Qué operación realiza? ¿Cuál es el resultado exacto? Ejecútelo y compare.

2.4. ⋆ Evaluando un polinomio I. Una operación usual en muchos métodos numéricos para aproximar la solución de algún problema es la evaluación numérica de un polinomio.
La evaluación del polinomio p de grado 3 en z:

$$p(z) = a_0\,z^3 + a_1\,z^2 + a_2\,z + a_3$$

puede reformularse de la siguiente manera:

$$p(z) = ((a_0\, z + a_1)\, z + a_2)\, z + a_3$$

Para el cálculo a mano el siguiente esquema (regla de Horner) ilustra el algoritmo resultante de la reformulación indicada:

a_0	a_1	a_2	a_3
	$z\, b_0$	$z\, b_1$	$z\, b_2$
b_0	b_1	b_2	b_3

con lo que resulta $p(z) = b_3$.

1. Calcular $p(8)$ si $p(x) = 2\, x^3 + x + 7$
2. Implementar un programa que calcule el valor numérico de un polinomio (los datos de entrada son el grado del polinomio, sus coeficientes y el punto de evaluación). La regla de Horner se puede expresar en forma recurrente por:

   $$b_0 = a_0 \qquad b_i = a_i + z b_{i-1} \quad i = 1, 2, \ldots n \quad b_n = p(z)$$

3. Ensayar el programa con algún ejemplo.
4. Contar el número de productos y sumas requeridos para la evaluación del polinomio p de grado n con la regla de Horner. Comparar con las que son necesarias si las potencias sucesivas de x son calculadas como $x^k = x x^{k-1}$ y luego el producto con el coeficiente a_{n-k}.

2.5. $\star$ **Evaluando un polinomio II.** Usar aritmética exacta, redondeo por truncamiento y redondeo simétrico para evaluar

$$f(x) = x^3 - 4\, x^2 + 2\, x - 2{,}2$$

en $x = 2.41$.
Para el redondeo considerar aritmética de tres dígitos.
Por ejemplo $23.2324 \rightarrow 23.2$.

Para ordenar el cálculo completar la siguiente tabla:

	x	x^2	x^3	$4\, x^2$	$2\, x$
exacta					
red. truncando					
red. simétrico					

Rta: $f(2.41)$: -6.614879 (exacto), -6.68 (truncando con tres dígitos con un error relativo de 0.0098), -6.58 (redondeo simétrico con un error relativo de 0.0052).

Volver a realizar el cálculo si $f(x)$ se expresa de la siguiente manera (método de Horner):

$$f(x) = x(x(x - 4) + 2) - \ 2.2$$

que minimiza el número de operaciones a realizar.

Rta:$f(2.41)$: -6.614879 (exacto), -6.61 (truncando con tres dígitos con un error relativo de 0.00073), -6.61 (redondeo simétrico con un error relativo de 0.00073).

2.6. ⋆ Aproximando el valor de la derivada de una función en un punto.

1. Utilizar la conocida fórmula
$$f'(x) \approx \frac{f(x+h) - f(x)}{h}$$
para aproximar el valor de la derivada de $f(x) = x^2$ en $x = 1$. Considerar valores de h de la forma 10^{-k} con k tomando los valores $1, 2, \ldots, 20$. Generar una tabla y gráficos con los valores de estas aproximaciones y el error relativo para cada valor de h. ¿ Cuál es en su computadora el valor de h de esa lista para el cuál el error relativo es menor? Intente argumentar respecto de un rango de valores de h en el que el error de aproximación es mas importante que el error de redondeo y otro rango en el cual el error de redondeo predomina sobre el error de aproximación.
2. Repetir lo del item anterior pero para la aproximación de f' conocida como de diferencia centrada y comparar:
$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2\,h}$$
3. Repetir lo del item anterior pero para $f(x) = x^3$. Analizar el error de la aproximación y obtener por inspección en una tabla el *mejor valor* de h.

2.7. Las raíces de una ecuación de segundo grado. Calcular las raíces de
$$x^2 - 4\,x + 3.9999999\ldots = 0$$
con tantos nueves para el término independiente como permita su calculadora. Calcule el resultado exacto (el termino independiente $3.\hat{9}$) y compare con el obtenido con la calculadora. Observar que la ecuación puede escribirse en la forma
$$(x-2)^2 = 10^{-m}$$
para algún m adecuado (que depende de la cantidad de *nueves* que se pudieron introducir).

2.8. ⋆ Otra ecuación de segundo grado. Obtener la raíz de menor valor absoluto de la ecuación
$$x^2 + 0.4002\,x + 0.8\,10^{-4} = 0$$
Utilizar aritmética de 4 dígitos y comparar con el resultado obtenido utilizando aritmética más exacta (la de su calculadora). Observar la pérdida de cifras significativas atribuida a la resta de números *muy próximos*. Implementar un método que permita calcular con mayor precisión esa raíz (Sugerencia: *racionalizar el numerador*).

3. Problemas mal condicionados

Desde un punto de vista intuitivo (ésta no es una definición formal) se dice que un problema está *mal condicionado* cuando pequeños errores en los datos del problema producen grandes errores en la solución del mismo.

3.1. $\star$ **Evaluación de una función.** Supongamos que se desea evaluar

$$y = \sqrt{\frac{534 - 7\,x^2}{6}} \quad \text{si} \quad x = \frac{611}{70}$$

considerando que x e y se representan en formato de punto flotante en base diez y con cuatro cifras significativas (se representan como un número decimal con cuatro cifras siendo la primera de ellas distinta de cero y multiplicado por una potencia adecuada de 10). Los calculos intermedios se hacen con todos los decimales de la calculadora y luego se redondean con cuatro cifras significativas antes de pasar a otro cálculo.
Si x se guarda de esta manera resulta 8.729 (ó $0.8729\ \ 10^1$). Verificar que operando como se propone resulta $y = 0.3162$.
Realice el cálculo con su calculadora científica y verifique que (con cuatro cifras significativas) resulta $y = 0.3377$.
Observe que un error relativo en x de 0.00005 trae aparejado un error relativo en y de 0.064. El error se propagó en el resultado amplificado mas de 1000 veces.

El problema de evaluar esta función para ese valor de x es un problema mal condicionado.

3.2. Un sistema lineal mal condicionado. Obtener la solución a los siguientes tres sistemas de ecuaciones lineales $Ax = b$. El segundo es una versión del primero con una leve modificación en los elementos de A y el tercero con un pequeño cambio en las componentes de b. Se escriben A y b como en la línea de comando de *Matlab*.

```
A = [10 7 8 7;7 5 6 5;8 6 10 9;7 5 9 10]; b=[32;23;33;31];

A1 = [10 7 8.1 7.2;7.08 5.04 6 5;8 5.98 9.89 9;6.99 4.99 9 9.98];
b1=[32;23;33;31];

A2 = [10 7 8 7;7 5 6 5;8 6 10 9;7 5 9 10];
b2=[32.1;22.9;33.1;30.9];
```

La solución exacta del primer sistema (el no perturbado) es `x = [1;1;1;1]`. Para resolver el sistema de ecuaciones con *Matlab* usar `inv(A)*b`. Comentar el resultado.

4. Algoritmos inestables

Se dice que un proceso numérico (un algoritmo numérico para resolver un problema) es *inestable* si pequeños errores que se produzcan en una etapa del proceso son amplificados en etapas posteriores produciendo la pérdida de exactitud de todo el proceso.

4.1. $\star$ **Inestabilidad numérica I.** El siguiente ejemplo muestra como el error de redondeo puede *destruir* el resultado de un cálculo si se elige un algoritmo *malo* o inestable.

El ejemplo: Calcular para $n = 0, 1, 2 \ldots, 8$

$$I_n = \int_0^1 \frac{x^n}{x+5}\, dx$$

1. Verificar que se cumple la siguiente relación recursiva

   $$I_n + 5\, I_{n-1} = \frac{1}{n}.$$

   Calcular I_0 con tres decimales (por redondeo) y usando la relación recursiva calcular para $1, 2, 3, 4$ (y aquí ya aparece un resultado extraño).

   Observar que en la fórmula recursiva el valor calculado para I_{n-1} se multiplica por 5 en cada paso (inicialmente el error es 0.0005). El error se incrementa en cada paso hasta que se hace de orden mas grande que el valor de la integral. Poco se logra aumentando la precisión del valor de I_0.
2. Verificar que la fórmula recursiva se puede reescribir de la siguiente manera :

   $$I_{n-1} = \frac{0.2}{n} - 0.2\, I_n.$$
3. Analizar el ejemplo construyendo un algoritmo estable. Puede observarse que I_n es una función decreciente de n. Suponga $I_{10} \approx I_9$ y a partir de allí calcule I_n para $n : 8, 7, 6, \ldots 0$ usando la nueva fórmula recursiva *"hacia atrás"*. Analizar porque ahora el error de redondeo no influye en el resultado.

4.2. Inestabilidad numérica II. Sea la sucesión de números definida en forma recurrente por:

$$\begin{aligned} x_{n+1} &= \frac{13}{3}\, x_n - \frac{4}{3}\, x_{n-1} \quad \forall n \geq 1 \\ x_0 &= 1 \\ x_1 &= \frac{1}{3} \end{aligned}$$

1. Verificar que $x_n = (\frac{1}{3})^n$
2. Usar la relación de recurrencia para generar la sucesión utilizando 7 decimales. Verificar que a partir de x_1 se *"pierde"* un decimal correcto por cada iteración. El error relativo cometido en la determinación de x_{15} es de 10^8 (verificar esto).

Nota: El algoritmo es inestable. Cualquier error presente en x_n se multiplica por $\frac{13}{3}$ en la determinación de x_{n+1}. Así resulta que un error en x_1 se propaga en x_{10} con un factor de $(\frac{13}{3})^9$.