

- Matlab tiene extensión .m
 - Poner edit en barra comando para abrir editor
 - Para ejecutar(run) puedes seleccionar por partes o todo PULSAR F9
 - INSTRUCCIONES
 - clear; borra las variables existentes
 - clc; borra los comandos usados de la consola
 - % para añadir comentarios
 - CREAR MATRICES :
 - >> a = [123(primera fila; 317 (segunda fila)] > creará la matriz 2x3
 - ; sirve para cambiar de fila
 - >> [n m] = size (a)
 - >> a=[1:1:1000] De 1 a 1000 de 1 en 1
 - Primer número es el comienzo, el de en medio de cuánto en cuánto y el último es el final
 - b=[0:2:1000] serían los pares de 0 hasta 1000
 - c=[99:-1:0] se puede utilizar también a la inversa
 - >> Se pueden concatenar matrices
 - x1=[7 8]
 - x2=[9 5]
 - x=[x1 x2] (crearía x=[7895])
 - >> Hay matrices predefinidas
 - ones(2,3) crearía una matriz 2 filas 3 columnas de unos
 - >>eye(4) matriz identidad con tantos 1 en la diagonal como se indique
 - REFERENCIAR ELEMENTOS DE LA MATRIZ
 - Si tienes una matriz a=[] , el comando a(2,3) coge el elemento de la fila 2 y columna 3
 - a(2,3:4)coge los elementos de la fila 2 y columna 3 4
 - a(3,[1:end])
 - CAMBIAR ELEMENTOS DE LA MATRIZ
 - >> A(A>8)= 10 – pone 10 a los valores de la matriz mayores de 8
 - >> A(3,:)+2 → al tener la matriz 3 filas sería lo mismo poner A(end,:)+2
 - suma 2 a la 3 fila(última)

- OPERAR CON MATRICES

>> $x((x>2) \& (x<5)) = -x((x>2) \& (x<5))$ → hace el inverso de los números entre 2 y 5

>> SUMA/ RESTA → $c = a + b$ → debe ser igual tamaño

>> MULTIPLICACIÓN → Columnas A = Filas B → $c = a * b$

$C = a.*b$ → al poner el punto no multiplicara las matrices normal hará simplemente la multiplicación de los elementos que estén en la misma posición

-EL PUNTO HACE QUE LAS OPERACIONES SEAN ELEMENTO A ELEMENTO

>> POTENCIA → $a^3 = A^*A^*A$ → A tiene que ser cuadrada (filas=columnas)

>> DIVISIÓN → $A/B = A^* \text{inv}(B)$ → B cuadrada e invertible

→ $A\backslash B = \text{inv}(A)^* B$ → A cuadrada e invertible

→ $1./a$ = pondria todos los elementos de la matriz como fracciones

>> $T = A'$ → inversa de A

APUNTES PARA COMPUTACIONAL

$$\cosh(1) \equiv \frac{\cosh(1+h) - 2\cosh(1) + \cosh(1-h)}{h^2}$$

Se pide:

- Construir un vector n con valores 1, 2, 3,..., 8.
- A partir de n construir un vector h con valores 0.1, 0.01, 0.001,..., 0.00000001.
- Construir un vector con los resultados de evaluar la expresión de la derecha para el vector h.
- Calcular el error relativo de los resultados de la expresión de la derecha con respecto a la de la izquierda.
- Dibujar, en la escala adecuada, la relación entre n y el error relativo (los valores de n deben ir en el eje horizontal y los del error relativo en el vertical).
- Calcular el número de cifras decimales correctas entre ambas expresiones.
- Dibujar, en la escala adecuada, la relación entre h y las cifras (los valores de h deben ir en el eje horizontal y los de las cifras en el vertical).
- Para qué valor de h se consiguen 8 cifras correctas.

Ejercicio 2.

Ejercicio 1

```
h=10.^-[1:8]
n=[1:8]
h=10.^-n
der = (cosh(1+h)-2*cosh(1)+cosh(1-h))./(h.^2)
izq = cosh(1)
er=abs(der-izq)/abs(izq)
semilogy(n,er)
figure(1),semilogy(n,er)
cif = -log10(er)
figure(2),semilogx(h,cif)
% No se consiguen 8 cifras. Lo más cercano (casi 8) es para h = 10^-4 que es
```

Para indicar las filas de una matriz:

A(1,2) -> cogería numero 1 fila y 2 columna

A(2,3:4)--> cogería la fila 2 y elementos DESDE columna 3 HASTA 4

a(2,[3 4])--> igual caso arriba

A(3,:) → al usar los dos puntos seleccionas todo en este caso fila 2 y todas las columnas, es decir la fila 2 entera

Ej. Asignar 10 a los valores mayores o iguales que 8 de la matriz A:

>> A(A>=8)=10

Ej. Sumar 2 a los elementos de la última fila de la matriz A:

>> A(end,:)+2

Extraer los elementos de la primera fila de A en un vector llamado B:

>> B=A(1, :)

Eliminar la última fila de la matriz A:

>> A(end,:)=[]

Guardar primera y tercera fila de A en una matriz C

>> C=A([1 3],:)

Dimensión de A y guardar em las variables m y n:

>>[n m]=size(A)

Extraer elementos mayores que 4 en y

>> y=x(x>4)

Poner a 0 los elementos menores o iguales a 4

>> x(x<=4)=0

3. Cambiar el signo a los elementos con valores mayores o iguales que 2 y menores que 5.

>> selected=(x>=2&x<5)

>> x(selected)=-x(selected)

• abs(v) → pone v en valor absoluto

• sum(v) suma los elementos de un vector

• prod(v) producto de los elementos de un vector

• mean(v) Media de los elementos del vector v

• max(v) devuelve valor máximo de las componentes de un vector. Si se quiere guardar la posición k en la que se alcanza el máximo y el valor máximo m: [y,k]=max(v)

. Análogamente min(v)

• Las funciones anteriores aplicadas a matrices realizan dichas operaciones por columnas.

FUNCION PLOT

plot(xdata, ydata, "estilo de la línea")

>> x=-5:0.1:5; %vector eje x

>> y=x.^2; %vector eje y

>> plot(x, y, 'r:s');

Crear un vector de 100 elementos equiespaciados en $[0, 2\pi]$:

```
>>x=linspace(0,2*pi,100); // x= [0:0.01:2*pi]
```

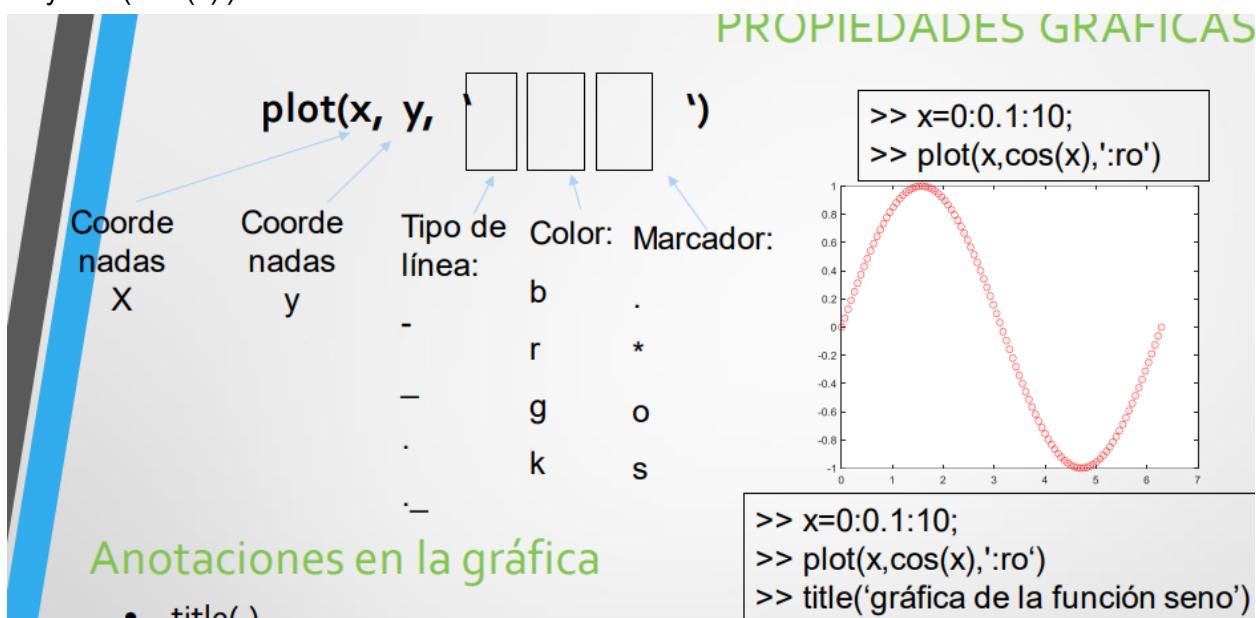
Para dibujarlo → >>plot(x,sin(x))

Anotaciones en la gráfica

```
>>title('gráfica de la función seno')
```

```
>>xlabel('Eje x')
```

```
>>ylabel('sen(x)')
```



FORMATO FPRINTF

```
>> fprintf('formato', var1,var2,var3,...)
```

%d → indica separación columnas → %4d, %16d

\n poner siempre al final para salto de línea

EJEMPLO

Calcular los valores aproximados del número e (tomamos $\exp(1)$ como valor exacto) que proporcionan los términos de la siguiente sucesión con $n=10.^[1:15]$.

Script creado en editor con nombre apru.m:

```
clear %Es recomendable iniciar el script con clear
```

```
% Valores de n para los que calculamos los valores de la sucesión  
k=1:15;  
n=10.^[1:15];
```

```
%Valor exacto en el límite:  
vex=exp(1);
```

```
%Vector de valores aproximados para los n considerados:  
vap=(1+(1./n)).^n;
```

```
% Vectores de errores relativos y cifras dec. significativas.  
erel=abs(vex-vap)/vex;  
ncif=floor(-log10(erel));
```

```
% Gráficas  
subplot(1,2,1), semilogy(k,erel, '*r');  
subplot(1,2,2), plot(k,ncif,'*g');
```

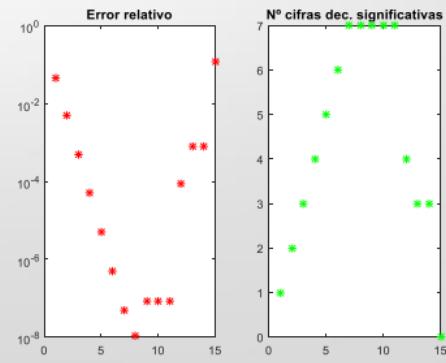
```
% ¿Cuál es el menor nº de cifras dec. signif. y en qué posición se alcanza?  
[c,p]=min(ncif);
```

```
44
```

$$e = \lim_{n \rightarrow \infty} \left(1 + \frac{1}{n}\right)^n$$

Valor exacto ($\exp(1)$ de Matlab)

valor aproximado para cada n



Calcular las 50 primeras iteraciones, y dibujar la gráfica de la sucesión $x_0=0$, $x_{n+1}=\cos(x_n)$

```
x=zeros(51,1);  
for k=1:50,  
    x(k+1)=cos(x(k));  
end  
plot(x,'.')  
x(end)
```

Usar bucles anidados asignando a cada elemento de A el valor 0.2 si el valor original era menor o igual a 0.4 y 0.7 si era mayor que 0.4:

```
>> A=rand(5,7);  
>> for k=1:5, for j=1:7, if A(k,j)<=0.4, A(k,j)=0.2; else A(k,j)=0.7; end  
end; end  
>> A
```

Repetir usando indexado lógico:
 $A(A<=0.4)=0.2$; $A(A>0.4)=0.7$;

PROBLEMA

Calcular valor aproximado de e sumando los primeros 30 sumandos de la serie
 $e=1+1/1!+1/2!+1/3!+\dots$

- a) Utilizando los comandos .*, ./, .^

```
>> n=[0:29]; fact=factorial(n); s=sum(1./fact)
s = 2.7183
>> error=abs(exp(1)-s)
error =0
```

Observación: Poco eficiente, recalcula factorial para cada valor de n.

- b) Mediante un bucle **for** ($s_0=1, s_k=s_{k-1}+1/k!$)

```
>> s=1; for k=1:29, s=s+1/factorial(k); end
```

¿Obtenemos la misma estimación que en el apartado anterior?

Observación: Poco eficiente, recalcula factorial.

- c) Modificar bucle anterior para sea más eficiente (calculo factorial iterativ. $1/k!=(1/(k-1)!)/k$)

```
>> s=1; temp=1; for k=1:29, temp=temp/k; s=s+temp; end
```

- d) Repetir bucle anterior de forma que termine (break) si el término a sumar es menor que 10^{-18} . ¿Hay diferencias entre las soluciones?

```
>>s=1; temp=1; for k=1:29, temp=temp/k; s=s+temp; if temp<1e-18; break; end; end
```

¿Cuántas iteraciones del bucle se hacen? ¿Por qué motivo se impone que el bucle termine si el término a sumar es menor que 10^{-18} ?

TERMINO FUN:

Ejemplo 1: Función que calcula el coseno de kx:

EDITOR:

```
% En el editor creamos el siguiente script
y1
function p=fun1(x,k)
p=cos(k*x);
end
```

VENTANA DE COMANDOS:

```
% Ejecutamos el script para los argumentos de entrada o
>>p=fun1(0,1)
p=1
```

Arguments de entrada
Arguments de salida:

Ejemplo 3: Función que admite como entrada un vector x , y devuelve la suma de los elementos del vector y el producto de los elementos del vector:

EDITOR:

```
function[s,p]=sumprod(x)
s=sum(x);
p=prod(x);
end
```

Arguments de entrada

Arguments de salida:

VENTANA COMANDOS:

```
>> [s,p]=sumprod(1:2:10)
s = 25
p = 945
```

· ¿Nº máquina que representa al nº real 0.4 ?:

- Lo llevamos al intervalo $[1,2)$: $0.4 \times 2^2 = 1.6 \rightarrow 0.4 = 1.6 \times 2^{-2}$
- Representamos la mantisa en base 2:

$$\begin{array}{llll} 1.6 > 1 & \rightarrow 1. & \text{resto} = 0.6 & (2^0) \\ 0.6 \times 2 = 1.2 > 1 & \rightarrow 1 & \text{resto} = 0.2 & (2^{-1}) \\ 0.2 \times 2 = 0.4 < 1 & \rightarrow 0 & \text{resto} = 0.4 & (2^{-2}) \\ 0.4 \times 2 = 0.8 < 1 & \rightarrow 0 & \text{resto} = 0.8 & (2^{-3}) \\ 0.8 \times 2 = 1.6 > 1 & \rightarrow 1 & \text{resto} = 0.6 & (2^{-4}) \\ \dots\dots\dots & \text{Se repite la cadena de bits} & & \end{array}$$

$$\rightarrow 0.4 = (1.1001001\dots)_2 \times 2^{-2}$$

No es nº máquina en esta representación

Posibles nos. máquina que aproximen a 0.4 $\approx \begin{cases} (1.100)_2 \times 2^{-2} = 0.375 & (\text{truncamiento}) \\ (1.101)_2 \times 2^{-2} = 0.40625 & (\text{redondeo más próximo}) \end{cases}$

NÚMEROS MÁQUINA $\hat{x} = \pm(1.b_1b_2b_3)_2 \times 2^e$ exponentes posibles: 0, -1, -2

- b) Calcular $\text{eps}(1)$ y una cota del error relativo de esta representación:

$$1 = (1.000)_2 \times 2^0 \quad \left. \begin{array}{l} \\ \text{Nº máquina siguiente a } 1 = (1.001)_2 \times 2^0 \end{array} \right\} \rightarrow \text{eps}(1) = (1.001)_2 \times 2^0 - (1.000)_2 \times 2^0 = 2^{-3} \times 2^0 = 2^{-3}$$

$$\text{Error relativo de esta representación} \leq \begin{cases} \frac{1}{2} 2^{-3} & \text{si aproximación por redondeo más próximo} \\ 2^{-3} & \text{si aproximación por truncamiento} \end{cases}$$

Posición del bit menos representativo de la mantisa

- N° cifras binarias significativas que garantiza esta representación si redondeo al más próximo: $-\log_2(\text{erel}) \geq 4$
- N° cifras decimales significativas que garantiza esta representación si redondeo al más próximo? $-\log_{10}(\text{erel})$

- c) ¿Cuál es el error relativo al representar el número real 0.4? N° cifras decimales significativas que se consiguen?

$$0.4 = (1.1001001\dots)_2 \times 2^{-2}$$

$$0.4 \approx \begin{cases} (1.100)_2 \times 2^{-2} = 0.375 \quad (\text{truncamiento}) \rightarrow E_{\text{rel}} = \frac{|0.4 - 0.375|}{0.4} = 0.0625 \\ \rightarrow N^{\circ} \text{ cifras decimales signif: } -\log_{10}(0.0625) = 0.8062 \approx 1 \\ (1.101)_2 \times 2^{-2} = 0.40625 \quad (\text{redondeo más próximo}) \rightarrow E_{\text{rel}} = \frac{|0.4 - 0.40625|}{0.4} = 0.015625 \approx 10^{-1} \\ \rightarrow N^{\circ} \text{ cifras decimales signif: } -\log_{10}(E_{\text{rel}}) = 1.8062. \end{cases}$$

1. [2 puntos] Sea la función

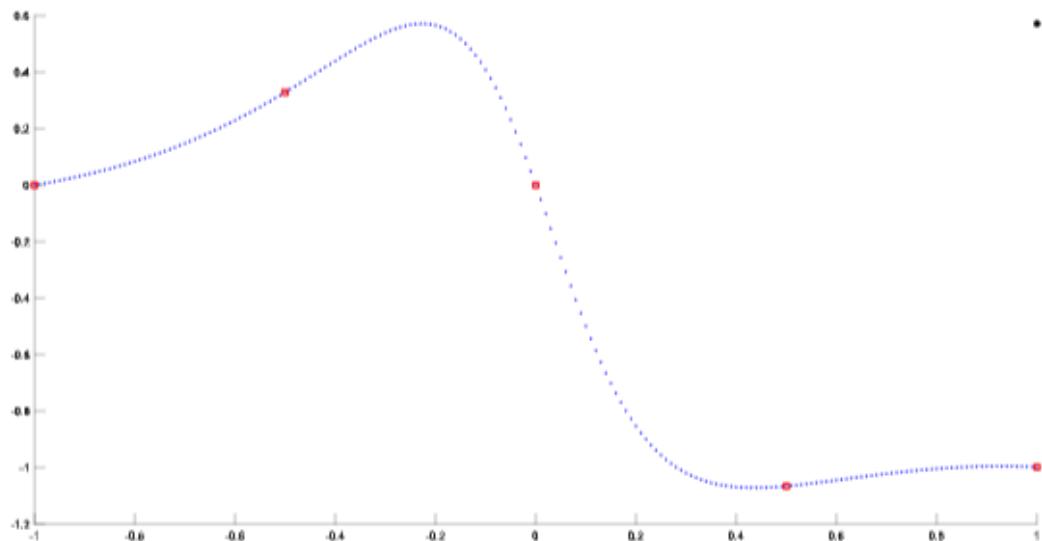
$$f(x) = \frac{10 \log(x^2 + x + 1)}{10x^3 - 20x^2 + x - 2}$$

- Dibujar la gráfica de $f(x)$ para x en el intervalo $[-1, 1]$ con un salto de 0.01 (con el atributo 'b').
- Sean los 5 puntos $xi=-1:0.5:1$. Calcular los valores que toma la función $f(x)$ en los puntos $xi: f(xi)$.
- En la gráfica anterior añadir los 5 puntos $(xi, f(xi))$ (con el atributo 'sr').
- En la gráfica anterior añadir el valor máximo de la función (con el atributo '*k').

Ejercicio_1

```
clear;clc;
x=-1:0.01:1;
fx=(10*log(x.^2+x+1))./(10.*x.^3)-20.*x.^2+x-2;
xi=-1:0.5:1;
fxi=(10*log(xi.^2+xi+1))./(10.*xi.^3)-20.*xi.^2+xi-2;

hold on
plot (x,fx, 'b.')
plot(xi,fxi, 'sr')
plot(max(fx), '*k')
hold off
```



2. [4 puntos] Sea

$$s = \sqrt[3]{5}$$

Utilizar el comando `^` para calcular el valor de s.

Para calcular s utilizamos la siguiente sucesión (método iterativo):

$$\begin{cases} x_0 = 1 \\ x_{n+1} = \sqrt[3]{5/x_n} \end{cases}$$

Implementar un script, con un bucle for que calcule las 50 primeras iteraciones. Almacenar el resultado en una tabla x.

A partir de la tabla x, con operaciones punto a punto, calcular las tablas del error relativo E_rel y del número de cifras de precisión N_cifras.

$$E_{rel} = |s - x_n| / |s|$$

Dibujar una figura con 3 gráficas:

- Con el comando subplot(131), la tabla x, con el atributo 'ob'.
- Con subplot(132) el Error relativo, con el atributo 'og'
- Con subplot(133) el Número de cifras, con el atributo 'or'.

Calcular el vector $d(n) = x_{n+1} - x_n$ sin utilizar bucles. ¿Para qué valor de n se verifica $x(n+1)-x(n) = 0$?

Calcular los índices donde el vector d se anula, utilizando indexación lógica.

Interpretación de los resultados:

- ¿La sucesión es convergente? ¿Porqué? ¿Pará que valor de n ha convergido?
- ¿Cuántas iteraciones son necesarias para alcanzar 5 cifras de precisión? ¿y 10 cifras?

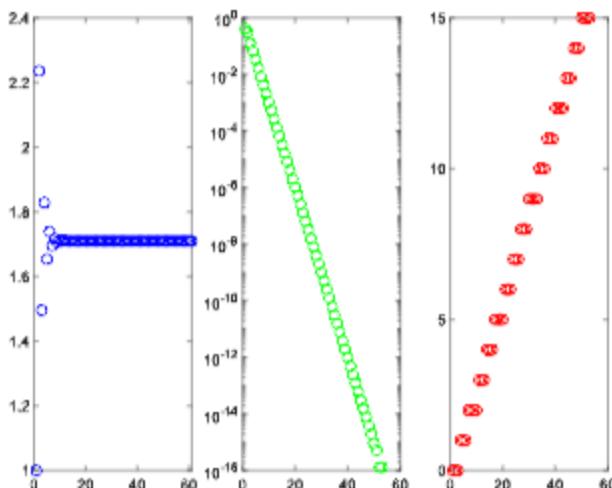
WUOLAI

- ¿Cuál es el mayor número de cifras de precisión que se obtiene? ¿Cuál es la precisión de la máquina (cuántas cifras de precisión alcanza Matlab)? ¿En qué iteración se alcanza la precisión de la máquina?
- ¿Están dibujados en la gráfica todos los valores de Erel y Ncifras? ¿Por qué?.
- ¿Cuál es la pendiente (aproximada) de la gráfica de Erel respecto de las iteraciones?
- Completar la siguiente frase:
'En cada 'x' iteraciones aumentan/disminuyen 'y' cifras de precisión.'

```

Ejercicio_2
clear;clc;
%Recordemos que una raiz cubica de un numero es como si lo elevasemos a 1/3
s=5.^((1/3));
x(1)=1;
for k=1:60
    x(k+1)=(5./x(k)).^(1/2);
end
%Calculamos ahora el error relativo y el nº de cifras
Erel=abs(s-x)./abs(s);
NumCifras=floor(-log10(Erel));
%Creamos una figura con tres graficas y las representamos adecuadamente
figure;
%Creamos el vector con todas las iteraciones de k
subplot(1,3,1); plot(x,'ob')
subplot(1,3,2); semilogy(Erel,'og')
subplot(1,3,3); plot(NumCifras,'or')
%Calculamos el vector dn definiendo previamente un vector igual que el de k
v=1:60;
d(v)=x(v+1)-x(v);
find(d==0)

```



La sucesión converge a la raíz cubica de 3 (el vector s, definido previamente) entre los valores de $n=[54,60]$; Para alcanzar 5 cifras de precisión, son necesarias 17 iteraciones y para alcanzar 10 cifras de precisión, son necesarias 34 iteraciones; En la iteración 54 se alcanza el nº maximo de cifras de precisión (15 cifras) y a partir de ahí, Matlab ya no alcanza mas cifras de precisión; No están dibujados todos los valores de Erel y de Ncifras, ya que hay algunos valores que no los alcanza Matlab a representar; La pendiente aproximada de Erel es $15/53=0.28$, pero como desciende negativamente, -0.28; "En cada 3 iteraciones aumenta una cifra de precisión".

Subplot(1,3,1), loglog(x,erel)

Creas un subplot de tamaño 1 - 3. Este empezaría en el 1

Subplot(1,3,2),semilogx(x,ncif)

. Este empezaría en el 2

Subplot(1,3,3)

. Este empezaría en el 3

3. [4 puntos] Queremos calcular el número real s que verifica la relación:

$$s = \cos(s)$$

Para obtener el número s, utilizamos el siguiente método iterativo:

$$\begin{cases} x = 1 \\ x = \cos(x) \end{cases}$$

Realizar el número de iteraciones necesarias para que $x - \cos(x) = 0$. Sea $s=x$.

¿Cuántas iteraciones son necesarias para que converja el método? ¿Cuál es el valor de s?

Volver a realizar las iteraciones. En cada iteración hacer lo siguiente:

- Calcular el Error relativo y el Número de cifras.
- Volcar en pantalla, con el comando `fprintf`, el número de iteración, el error relativo y el número de cifras

¿Cuál es la velocidad de convergencia?

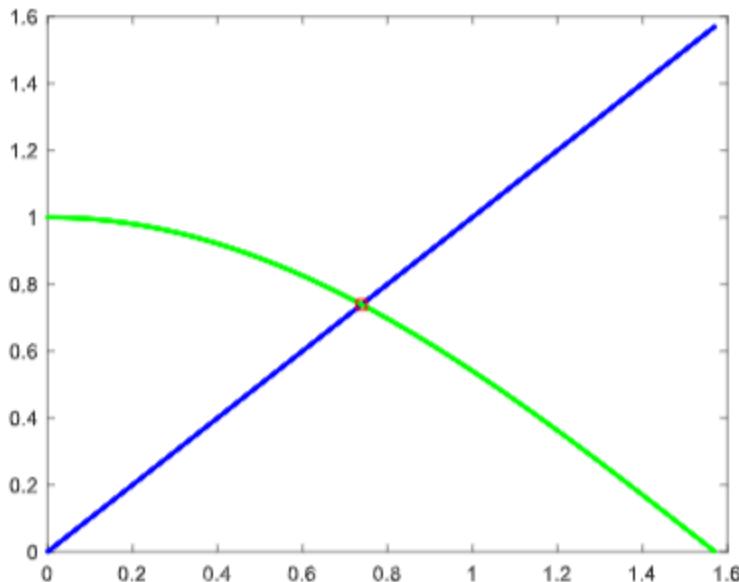
En una figura dibujar en la misma gráfica las funciones $f(x)=x$, $g(x)=\cos(x)$ y el valor (s, s) . ¿Qué relación tienen las gráficas de las funciones y el valor s?

```

Ejercicio_3
clear;clc
x=1;
for k=1:90
    x=cos(x);
end
s=x;
%Se comprueba si es 0
s-cos(s)
%Volvemos a iterar..
x=1;
for k=1:90
    x=cos(x);
    Erel=abs(x-s)/s;
    Ncifras=floor(-log10(Erel));
    fprintf(['En la iteración %d el Erel es %.10e y ' ...
        'el numero de cifras es %d\n'],k,Erel,Ncifras);
end
%Dibujamos en una figura la siguiente grafica
figure;
x=0:0.001:pi/2;
plot(x,x,'b.',x,cos(x),'g.',s,s,'sr')

```

Observamos pues que son necesarias 90 iteraciones para hacer que el método converja; s vale 0.7391; cada 6 iteraciones aumenta una cifra de precisión;



Si observamos la grafica vemos que el punto rojo representa el punto de cobvergencia, el valor s, que es igual a la solución de la ecuación planteada," $s=\cos(s)$ ".

FORMATO FPRINTF:

`fprintf('Formato fprintf es el siguiente EL VALOR EXACTO %.10e EL VALOR APROXIMADO %.10e Y EL VALOR V3 %.10e %d/n',vex2,vap2,v3)`

Ejemplo (Examen Computac. 2019). Comparar los resultados numéricos que proporcionan las siguientes fórmulas matemáticas equivalentes para valores grandes de x:

$$\frac{4}{x+\sqrt{x^2-4}} = x - \sqrt{x^2-4} \rightarrow \begin{array}{l} \text{Fórmula 1} \\ \text{Fórmula 2} \end{array}$$

$x \approx \sqrt{x^2-4}$ si x grande

%1. Consideramos $x=[10, 10^2, \dots, 10^{10}]$

$k=1:10;$

$x=10.^k;$

%2. Evaluamos la fórmula 1 para los valores de x y lo guardamos en el vector vex.

$vex=4./(\sqrt{x.^2-4});$

%3. Evaluamos la fórmula 2 para los valores de x y lo guardamos en el vector vap.

$vap=x-sqrt(x.^2-4);$

%4. Calculamos el error relativo y el nº de cifras dec. significativas. Pintamos sus

%gráficas respecto de x (en escala adecuada).

$erel=abs(vex-vap)/abs(vex);$

$ncif=floor(-log10(erel));$

$subplot(1,2,1), loglog(x,erel,'r*'), title('Error relativo')$

$subplot(1,2,2), semilogy(x,ncif,'b*'), title('Nº cifras decimales significativas')$

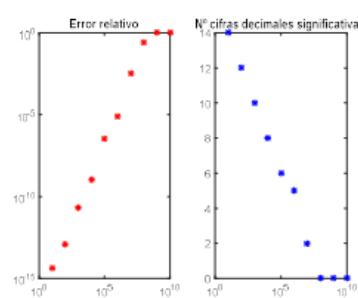
%5. Máximo de nº cifras dec. significativas y para qué valor de x se obtiene? ¿Y el mínimo?

$[mx,p]=max(ncif)$

$mx=14, p=1 \rightarrow x=10^{14}$

$[mn,l]=min(ncif)$

$mn=0 \quad l=8 \rightarrow a \text{ partir de } x=10^8$



k= 1 error rel:4.26e-15 nº cifras:14
 k= 2 error rel:1.22e-13 nº cifras:12
 k= 3 error rel:2.06e-11 nº cifras:10
 k= 4 error rel:1.12e-09 nº cifras: 8
 k= 5 error rel:3.38e-07 nº cifras: 6
 k= 6 error rel:7.61e-06 nº cifras: 5
 k= 7 error rel:3.48e-03 nº cifras: 2
 k= 8 error rel:2.55e-01 nº cifras: 0
 k= 9 error rel:1.00e+00 nº cifras: 0
 k=10 error rel:1.00e+00 nº cifras: 0

57

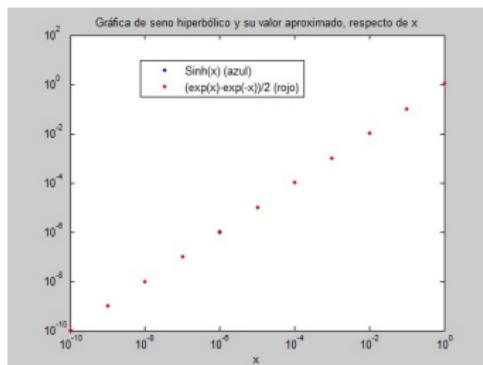
FIJARSE EN COMO HACER EL MAXIMO Y MINIMO DE CIFRAS SIGNIFICATIVAS

Valores "exactos y aproximados". Error relativo. Visualización

```

x = 10.^(-[0:10])
V_exacto=sinh(x);V_aprox=(exp(x)-exp(-x))/2;
Erel=abs(V_exacto-V_aprox)./abs(V_exacto);
Cota=eps(1)./sinh(x);
loglog(x,V_exacto,'b.',x,V_aprox,'r.');
title('Gráfica de sinh(x): V-exacto (azul), V-aprox (rojo), respecto de x')
loglog(x,Erel,'bo',x,Cota,'ro');
title('Gráfica del error relativo del sinh(x)(azul) y su cota (rojo), respecto de x')
  
```

COTAS
Ej.



Gráfica del seno hiperbólico y su aproximación, respecto de la variable x

62

Examen Julio 20

Un ordenador con una representación en coma flotante (base 2) usa números de la forma:

$$\begin{cases} x = (0.b_1 b_2)_2 \times 2^{-1} \text{ si } e=0 \text{ (nº denormaliz.)} \\ x = (1.b_1 b_2)_2 \times 2^{e-2} \text{ si } e=1,2,3 \text{ (nº normalizados)} \end{cases} \rightarrow \text{Exponente admisible (e=3)}$$

1. En dicho ordenador se ejecuta $a=2.2$; $b=0.15$; $c=a+b$; Rellenar la tabla con los datos pedidos:

• $a = 2.2 = 1.1 \times 2^1 = (1.0001..)_2 2^{1-2} \in [(1.00)_2 2^{3-2}, (1.01)_2 2^{3-2}] \rightarrow a \approx (1.00)_2 2^{3-2} = 2$

Representamos 1.1 en base 2:

$$\begin{array}{l|l|l} 1.1 > 1 & \rightarrow 1. (2^0) & \text{resto}=0.1 \\ 0.1 \times 2 = 0.2 < 1 & \rightarrow 0 (2^0) & \text{resto}=0.2 \\ 0.2 \times 2 = 0.4 < 1 & \rightarrow 0 (2^0) & \text{resto}=0.4 \\ 0.4 \times 2 = 0.8 < 1 & \rightarrow 0 (2^0) & \text{resto}=0.8 \\ 0.8 \times 2 = 1.6 > 1 & \rightarrow 1 (2^0) & \text{resto}=0.6 \\ \dots & & \end{array}$$

Exponente no admisible Exponente admisible (e=0)

Más próximo: $(1.00)_2 2^{3-2} = 2$

\rightarrow Codificación: $b_1 = b_2 = 0$, $e=3$

• $b = 0.15$;

$0.15 \times 2^3 = 1.2 \Rightarrow b = 0.15 = 1.2 \times 2^{-3} = (1.00..)_2 2^{3-3} = (0.0100..)_2 2^{-1} \approx (0.01)_2 2^{-1} = 0.125$

$$\begin{array}{l|l|l} 1.2 > 1 & \rightarrow 1. (2^0) & \text{resto}=0.2 \\ 0.2 \times 2 = 0.4 < 1 & \rightarrow 0 (2^0) & \text{resto}=0.4 \\ 0.4 \times 2 = 0.8 < 1 & \rightarrow 0 (2^0) & \text{resto}=0.8 \\ \dots & & \end{array}$$

\rightarrow Codificación: $b_1 = 0$, $b_2 = 1$, $e=0$

• $c = a + b \approx 2 + 0.125 = 2 + 2^{-3} = (1.0001)_2 2^1 \approx (1.00)_2 2^{3-2} = 2 \rightarrow$ Codificación: $b_1 = b_2 = 0$, $e=3$

Números coma flotante de la forma:

$$\begin{cases} x = (0.b_1b_2)_2 \times 2^{-1} & \text{si } e=0 \text{ (nº denormaliz.)} \\ x = (1.b_1b_2)_2 \times 2^{e-2} & \text{si } e=1, 2, 3 \text{ (nº normalizados)} \end{cases}$$

2. ¿Cuál es la distancia máxima y mínima entre números máquina consecutivos en esta representación?

Previo: $\text{eps}(1) = 2^{-2}$ → Observación: distancia entre mantisas consecutivas es 2^{-2}

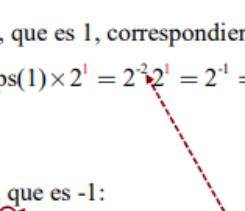
· Distancia máxima:

Nos fijamos en los números máquina con exponente máximo, que es 1, correspondiente a $e=3$.

Distancia entre dos números consecutivos de exponente 1 = $\text{eps}(1) \times 2^1 = 2^{-2} \times 2^1 = 2^{-1} = 0.5$

· Distancia mínima:

Nos fijamos en los números máquina con exponente mínimo, que es -1:

Distancia entre dos números consecutivos de exponente -1 = $2^{-2} \times 2^{-1} = 2^{-3} = 0.125$  posición bit menos representativo de la mantisa

NÚMEROS MÁQUINA $\hat{x} = \pm(1.b_1b_2b_3)_2 \times 2^e$ exponentes posibles: 0, -1, -2

a) Valor del menor número máquina no nulo positivo $v_{\min} = (1.000)_2 \times 2^{-2} = 0.25$

- Valor del mayor número máquina $v_{\max} = (1.111)_2 \times 2^0 = 1.875$
- ¿Nº máquina que representa al nº real 1 ?: $1=(1.000)_2 \times 2^0$
- ¿Nº máquina que representa al nº real 0.4 ?:

- Lo llevamos al intervalo $[1,2)$: $0.4 \times 2^2 = 1.6 \rightarrow 0.4 = 1.6 \times 2^{-2}$
- Representamos la mantisa en base 2:

1.6 > 1	$\rightarrow 1.$	resto = 0.6	(2^0)
$0.6 \times 2 = 1.2 > 1$	$\rightarrow 1$	resto = 0.2	(2^{-1})
$0.2 \times 2 = 0.4 < 1$	$\rightarrow 0$	resto = 0.4	(2^{-2})
$0.4 \times 2 = 0.8 < 1$	$\rightarrow 0$	resto = 0.8	(2^{-3})
$0.8 \times 2 = 1.6 > 1$	$\rightarrow 1$	resto = 0.6	(2^{-4})
..... Se repite la cadena de bits			

$\rightarrow 0.4 = (1.1001001\dots)_2 \times 2^{-2}$
No es nº máquina en esta representación

Posibles nos. máquina que aproximen a 0.4 $\approx \begin{cases} (1.100)_2 \times 2^{-2} = 0.375 & \text{(truncamiento)} \\ (1.101)_2 \times 2^{-2} = 0.40625 & \text{(redondeo más próximo)} \end{cases}$

NÚMEROS MÁQUINA $\hat{x} = \pm(1.b_1b_2b_3)_2 \times 2^e$ exponentes posibles: 0, -1, -2

b) Calcular $\text{eps}(1)$ y una cota del error relativo de esta representación:

- $1=(1.000)_2 \times 2^0$ $\rightarrow \text{eps}(1)=(1.001)_2 \times 2^0 - (1.000)_2 \times 2^0 = 2^{-3} \times 2^0 = 2^{-3}$
Nº máquina siguiente a 1 $= (1.001)_2 \times 2^0$
- Error relativo de esta representación $\leq \begin{cases} \frac{1}{2}2^{-3} & \text{si aproximación por redondeo más próximo} \\ 2^{-3} & \text{si aproximación por truncamiento} \end{cases}$ Posición del bit menos representativo de la mantisa
- Nº cifras binarias significativas que garantiza esta representación si redondeo al más próximo: $-\log_2(\text{erel}) \geq 4$
- Nº cifras decimales significativas que garantiza esta representación si redondeo al más próximo? $-\log_{10}(\text{erel})$

c) ¿Cuál es el error relativo al representar el número real 0.4? ¿Nº cifras decimales significativas que se consiguen?

$$0.4 = (1.1001001\dots)_2 \times 2^{-2}$$

$$0.4 \approx \begin{cases} (1.100)_2 \times 2^{-2} = 0.375 & \text{(truncamiento)} \\ (1.101)_2 \times 2^{-2} = 0.40625 & \text{(redondeo más próximo)} \end{cases}$$

$$\rightarrow \text{E_rel} = \frac{|0.4 - 0.375|}{0.4} = 0.0625$$

$$\rightarrow \text{Nº cifras decimales signif: } -\log_{10}(0.0625) = 0.8062 \approx 1$$

$$\rightarrow \text{E_rel} = \frac{|0.4 - 0.40625|}{0.4} = 0.015625 \approx 10^{-4}$$

$$\rightarrow \text{Nº cifras decimales signif: } -\log_{10}(0.015625) = 4.8062 \approx 5$$

CUANDO EL NUMERO ES INFINITO

- TRUNCAMIENTO= cogen solo 3 primero decimales
- REDONDEO MAS PRÓXIMO = coger los 3 decimales que nos hagan acercarnos al N° inicial

REPRESENTACIÓN COMA FLOTANTE

PASO 1= Convertir a binario

PASO 2= Escribir en notación científica

PASO 3 = Seguir el estándar IEEE 754 32 bit

→ 1 bit signo - = 1 / signo + = 0

→ 8 bit exponente +127

→ 23 bit mantisa

Ejemplo Escribir (-134.3135) en coma flotante

10

PRIMERO cogenes parte entera 134

134 / 2 = 67.0 si el decimal<0.5=0, si no =+1 a la parte entera

67/2 = 33.5 1

33.5/2=16.75 1

16.75/2= 8.375 0

8.375/2= 4.1875 0 se ordena de abajo a arriba →(-10000110.

4.18= 2.09 0

2.09/2 = 1.04 0

1.04/2= 0.52 1 (se para al ser la parte entera 0 →(-10000110.0101)2

SEGUNDO cogenes parte Fraccionaria

0.31*2=0.625 → 0

0.625*2= 1.25 1 De arriba a abajo→ .0101

0.25*2=0.5 0

0.5*2= 1.0 1 (terminas decimal=0)

TERCERO pasas notación científica

(-10000110.0101)2 → (-1.00001100101)* 2^7 cuentas veces movida la coma y multiplicado por la base

CUARTO

1 digito→ al ser negativo primer dígito 1

8 digitos→ n°exponente + 127 = 127+7 = 134 10 → pasas a binario(hecho antes)--> 10000110

23 mantisa→ parte derecha de la coma de notación científica→00001100101(al no ser 23 rellenas con ceros)

RESULTADO FINAL → 1 10000110 000011001010000000000000

Ejemplo

En una representación IEEE-754 simple precisión ¿cuál es el valor decimal representado como [s: 1] [e: 01111100] [m:1100000...0]?

s=1 → -

Exponente= $(01111100)_2 - 127 = 124 - 127 = -3$

Mantisa= $(1.11000000 \dots 0)_2 = 1.75$

$$\left. \begin{array}{l} \\ \\ \end{array} \right\} -1.75 \times 2^{-3} = -0.21875$$

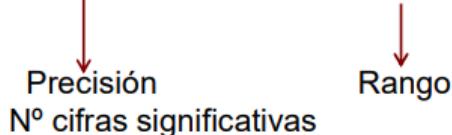
Mantisa= $2^0 + 2^{-1} + 2^{-2}$ (ELEVAR LOS 1 SEGUN SU POSICIÓN),
EXPONENTE IGUAL PERO ELEVANDO A POSITIVOS= $2^6 + 2^5 + 2^4 + 2^3 + 2^2$

REPRESENTACIÓN COMA FLOTANTE. ESTÁNDAR IEEE 754

Formatos:

Simple Precisión: 4 bytes (32 bits)= 24 mantisa con signo + 8 exponente (signo incl.)

Doble Precisión: 8 bytes (64 bits)= 53 mantisa con signo +11exponente (signo incl.)



Ejercicio (con Matlab)

Matlab por defecto trabaja con doble precisión.

1. - Creamos la variable: a=0.1;

- Crear una variable as convirtiendo a en precisión simple:
as=single(a);

- Con el comando whos ver el nº de bytes que emplea Matlab en cada caso:

Name	Size	Bytes	Class	Attributes
a	1x1	8	double	
as	1x1	4	single	

2. Con comando fprintf volcar ambas variables en dos líneas con 15 decimales (formato %.15f) ¿Coinciden ambos valores? ¿Cuántas cifras significativas coincidentes tienen los números a y as con 0.1?

```
fprintf("%.15f ",a,as)
0.100000000000000
0.100000001490116
```

3. Repetir el apartado anterior mostrando 20 decimales ¿El nº 0.1 es un nº máquina en la representación empleada por Matlab?

```
fprintf("%.20f ",a,as )
0.1000000000000001000
0.10000000149011612000
```

4. Calcular el error relativo que produce el número as (simple precisión) respecto del valor 0.1 Calcular el nº de cifras significativas que proporciona ¿Coinciden con las cifras obtenidas en el apartado 1?

```
er=abs(0.1-as)/0.1
er = 1.4901e-08<10^-7
```

Ejercicio (Matlab)

```

k=1;
% overflow (doble precisión):
while(2^k<inf),
    k=k+1;
end
k-1          (1023)

(a=1; k=1; while(a<inf),  a=2
k=k+1;end)

```

% underflow

```
while(2^(-k)~=0),  
      k=k+1;  
end  
k-1      (1074)
```

% epsilon (=2^k)

```

while(1+(2^(-k))~=1),
  k=k+1;
end
k-1          (52)
eps          (2.2204e-16)

```

NOTA: Confrontar los valores de los parámetros obtenidos con los señalados en diapositiva o doble prec.

NOTA: Confrontar los valores de los parámetros obtenidos con los señalados en diapositiva con tabla doble prec.

Ejercicio

- ¿0.1 es un nº máquina en simple o doble precisión?

mantis

onente

1

↑

1

- Con Matlab, hacemos un pequeño programa para sumar a $x=0$ iterativamente 0.1 hasta que sea igual a 1:

```
x=0;  
iter=0;  
while(x~>1),  
x=x+0.1;  
iter=iter+1;  
end
```

- ¿En qué iteración se debería parar el bucle?
 - ¿En qué iteración se para el bucle?

Ejercicio

Se considera el siguiente sistema de representación en coma flotante en base 2 de números positivos con dos bits para la mantisa y dos bits para el exponente:

NÚMEROS MÁQUINA representados de la forma:

$$\begin{cases} \hat{x} = (0.b_1b_2)_2 \times 2^0 \text{ si } e = (e_1e_2)_2 = (00)_2 = 0 \\ \hat{x} = (1.b_1b_2)_2 \times 2^{e-1} \text{ si } e = (e_1e_2)_2 \neq 0 \end{cases}$$

MANTISA EXPONENTE

Se ALMACENAN EN MEMORIA los siguientes 4 bits: b_1 b_2 e_1 e_2

· 2 bits (b_1, b_2) para la mantisa $\rightarrow m = (1.b_1b_2)_2$

· 2 bits para $e (e_1, e_2) \rightarrow e = (e_1e_2)_2$

- Calcular todos los números máquina de esta representación y representar en la recta real.
- Rango de nº. Calcular v_{\max} y v_{\min} los valores de los números máquina mayor y menor no nulo, respectivamente.
- Precisión:
 - Calcular $\text{eps}(1)$, $\text{eps}(3)$ y $\text{eps}(6)$.
 - Valor de x positivo a partir del cual $1+x \neq 1$
 - Cota del error relativo para representación de nos. normalizados.
 - Aproximar 0.2 por el nº máquina más próximo y calcular el error relativo.
- Mayor distancia entre números máquina consecutivos.
- Sea $x=5.6$, calcular su número máquina y los 4 bits a almacenar en memoria.
- ¿Qué valor decimal representa el nº codificado como $(b_1b_2)(e_1e_2) = (10)(01)$?

a) RANGO DE VALORES DE ESTA REPRESENTACIÓN → VALORES (BITS) EXPONENTE

Menor número máquina (denormalizado) no nulo $v_{\min} = (0.01)_2 \times 2^0 = 0.25$

Mayor número máquina denormalizado: $(0.11)_2 \times 2^0 = 2^0 - 2^{-2} = 0.75$

Menor número máquina normalizado = $(1.00)_2 \times 2^{1-1} = 1$

Mayor número máquina (normalizado) $v_{\max} = (1.11)_2 \times 2^{3-1} = 7$

b) PRECISIÓN. ERROR RELATIVO. CIFRAS SIGNIFICATIVAS → BITS MANTISA

- $\text{eps}(1) = 2^{\text{(posición último bit mantisa)}} = 2^{-2}$
- $1 = (1.00)_2 \times 2^0$
- N^o máquina siguiente a 1 = $(1.01)_2 \times 2^0$
- $\text{eps}(3) = \text{eps}(1) \cdot 2^{\text{(exponente de 3)}} = 2^{-2} \cdot 2^1 = 2^{-1}$
- $3 = (1.10)_2 \times 2^1$
- N^o máquina siguiente a 3 = $(1.11)_2 \times 2^1$
- $\text{eps}(6) = \text{eps}(1) \cdot 2^{\text{(exponente de 6)}} = 2^{-2} \cdot 2^2 = 1$

Posición del bit menos representativo de la mantisa

Observación: Estos valores se pueden ver directamente en la Tabla anterior donde se han listado los números máquina.

· Valor de x positivo a partir del cual:

- $1+x \neq 1 \quad x > \text{eps}(1)/2 = 2^{-3} = 0.175$
- $3+x \neq 3 \quad x > \text{eps}(3)/2 = 2^{-2} = 0.25$

· Si $a=0.2$ qué números máquina resultan de las siguientes operaciones:

- $1+a \approx 1 + \text{eps}(1) = 1.25 \quad \text{porque } \text{eps}(1)/2 < a < \text{eps}(1)$

Otra forma: $1+a \approx (1.00)_2 2^0 + (0.01)_2 2^0 = (1.01)_2 2^0 = 1.25$

- $3+a \approx 3 \quad \text{porque } 0 < a < \text{eps}(3)/2$

Otra forma: $3+a \approx (1.10)_2 2^1 + (0.01)_2 2^0 = [(1.10)_2 + (0.001)_2] 2^1 = \underbrace{(1.101)_2 2^1}_{\text{no es máquina}} \approx \underbrace{(1.10)_2 2^1}_{\text{es máquina más cercana}} = 3$

- $\underbrace{(1+a)}_{1.25} + \underbrace{(3+a)}_{3} \approx 1.25 + 3 = 4.25 \approx 4$

Observación: 0.2 no es n^o máquina en esta representación, se approxima por n^o máquina más próximo:

$$a=0.2 \approx 0.25 = (0.01)_2$$

- Error relativo (E_{rel}) de esta representación para números con $e \neq 0$ si aproximación por redondeo más próximo

$$E_{\text{rel}} = \frac{|x - \bar{x}|}{|x|} \leq \frac{|x_i - x_d|}{|x|} = \frac{1}{2} \frac{|(1.b_1b_2)_2 \times 2^{\exp(x)} - (1.b_1(b_2+1))_2 \times 2^{\exp(x)}|}{|m \times 2^{\exp(x)}|} \leq \frac{1}{2} \frac{2^{-2} 2^{\exp(x)}}{1 \cdot 2^{\exp(x)}} = \frac{1}{2} 2^{-2} = \frac{1}{2} \text{eps}(1) = 2^{-3}$$

Posición del bit menos representativo de la mantisa

dónde x es un n^o real entre el mayor y menor n^o normalizado y x_i y x_d son los n^o máquina más próximos a x por la izquierda y por la derecha respectivamente.

($|x| = m 2^e \times 2^{\exp(x)} \geq 2^{\exp(x)}$, por ser $m \geq 1$)

Observación: $E_{\text{rel}} \leq 2^{-2}$ si la aproximación es por truncamiento.

- N^o cifras binarias significativas que garantiza esta representación si redondeo al más próximo: $-\log_2(\text{erel}) \geq -\log_2(2^{-3}) = 3$ (al menos 3 cifras binarias)
- N^o cifras decimales significativas que garantiza esta representación si redondeo al más próximo: $-\log_{10}(\text{erel}) \geq 9.03 \times 10^{-1}$ (entre ninguna y 1 cifra dec.)

- ¿Cuántos bits deberían tener la mantisa y el exponente para obtener 3 cifras decimales significativas de precisión?

Precisión → Bits MANTISA

Si la mantisa es $(1.b_1\dots b_p)_2$ una cota del error relativo es $E_{\text{rel}} \leq \frac{1}{2} 2^{-p}$

si acotamos este valor por 10^{-3} tendremos garantizadas 3 cifras decimales significativas de precisión:

$$E_{\text{rel}} \leq \frac{1}{2} 2^{-p} < 10^{-3} \rightarrow -(p+1) < -3 \log_2(10) \rightarrow p > 3 \log_2(10) - 1 = 8.9657$$

Se concluye que una representación con 9 bits para la mantisa mas uno para el signo garantizará 3 cifras dec. significativas de precisión.

- Cota del error relativo si se aproxima 0.2 por el n^o máquina más próximo:

$$0.2 \approx 0.25 \quad (\text{n}^o \text{ máquina más próximo}) \rightarrow E_{\text{rel}} = \frac{|0.2 - 0.25|}{0.2} = 0.25 \rightarrow 25\%$$

c) REPRESENTACIÓN DE ALGUNOS NÚMEROS.

- ¿Nº máquina más próximo que representa al nº real 5.6?: $6 = (110)_2 = (1.\textcolor{red}{10})_2 \cdot 2^2 = (1.10)_2 \cdot 2^{3-1}$
 \rightarrow Codificación: $b_1 = 1, b_2 = 0, e = (1\textcolor{red}{1})_2$

$$5.6 = m \times 2^e \rightarrow \begin{cases} \text{exponente} = 2 = e - 1 \rightarrow e = 3 = (\textcolor{red}{11})_2 \rightarrow \text{Codificación: } e_1 = 1, e_2 = 1 \\ m = 5.6 / 2^2 = 1.4 \end{cases}$$

Representamos la mantisa en base 2:

$$\begin{array}{l} 1.4 > 1 \rightarrow 1. \quad (2^0) \quad \text{resto} = 0.4 \\ 0.4 \times 2 = 0.8 < 1 \rightarrow 0 \quad (2^{-1}) \quad \text{resto} = 0.8 \\ 0.8 \times 2 = 1.6 > 1 \rightarrow 1 \quad (2^{-2}) \quad \text{resto} = 0.6 \\ 0.6 \times 2 = 1.2 > 1 \rightarrow 1 \quad (2^{-3}) \quad \text{resto} = 0.2 \\ 0.2 \times 2 = 0.4 < 1 \rightarrow 0 \quad (2^{-4}) \quad \text{resto} = 0.4 \\ \dots \dots \text{Se repite la cadena de bits (011)} \end{array} \rightarrow 1.4 = (1.01100110\dots)_2 \rightarrow (1.01)_2 < 1.4 < (1.10)_2$$

Posibles nos. máquina que aproximan a 5.6 en esta representación $\approx \begin{cases} (1.01)_2 \times 2^2 = 5 \text{ (truncamiento)} \\ (\textcolor{red}{1.10})_2 \times 2^2 = 6 \text{ (redondeo más próximo)} \end{cases}$

- Error relativo $= \frac{|5.6 - 6|}{5.6} = 0.07142 < 10^{-1} \rightarrow$ garantiza una cifra decimal significativa.

- ¿Qué valor representa el nº codificado como $(b_1 b_2) (e_1 e_2) = (10) (01)$?

$$\left. \begin{array}{l} m = (1.10)_2 = 1.5 \\ e = (01)_2 = 1 \end{array} \right\} \rightarrow (1.10)_2 \times 2^{(01)_2 - 1} = 1.5 \times 2^{1-1} = 1.5$$

1. ¿Qué nº (en decimal) está codificado por el byte 1 101 1100?

$$\begin{array}{ccccccc} \boxed{1} & & \boxed{1} \boxed{0} \boxed{1} & & \boxed{1} \boxed{1} \boxed{0} \boxed{0} & & \rightarrow \hat{x} = (-1)^e (1.m_1 m_2 m_3 m_4)_2 \times 2^{e-4} \text{ si } e = (e_1 e_2 e_3)_2 \neq 0 \\ \downarrow & & \downarrow & & \downarrow & & \\ - & & e = (\textcolor{red}{101})_2 = 5 \neq 0 & & m = (\textcolor{red}{1.1100})_2 & & \rightarrow \hat{x} = -(1.1100)_2 \times 2^{5-4} = -(1 + 1/2 + 1/4) \times 2 = -3.5 \end{array}$$

2. ¿Cuántos números máquina distintos tiene esta representación? 2⁴ mantisas, 2³ exponentes, 2 signos $\rightarrow 2^8 - 1$

3. Mayor nº positivo representable:

$$\begin{array}{ccccccc} \boxed{0} & \boxed{\boxed{1}}\boxed{\boxed{1}} & \boxed{\boxed{1}}\boxed{\boxed{1}}\boxed{1} & \rightarrow & \hat{x} = (-1)^s(1.m_1m_2m_3m_4)_2 \times 2^{e-4} \text{ si } e=(e_1e_2e_3)_2 \neq 0 \\ \downarrow & \downarrow & \downarrow & & & & \\ + & e = (\underline{\underline{11}})_2 = 7 \neq 0 & m = (\underline{\underline{1111}})_2 & \rightarrow & \hat{x} = (1.1111)_2 \times 2^{7-4} = (1+1/2+1/4+1/8+1/16) \times 2^3 = 15.5 \end{array}$$

Menor nº positivo representable:

$$\begin{array}{ccccccc} \boxed{0} & \boxed{\boxed{0}}\boxed{\boxed{0}} & \boxed{\boxed{0}}\boxed{\boxed{0}}\boxed{1} & \rightarrow & \hat{x} = (-1)^s(0.m_1m_2m_3m_4)_2 \times 2^{-3} \text{ si } e=(e_1e_2e_3)_2 = 0 \\ \downarrow & \downarrow & \downarrow & & & & \\ + & e = (\underline{\underline{000}})_2 = 0 & m = (\underline{\underline{00001}})_2 & \rightarrow & \hat{x} = (0.0001)_2 \times 2^{-3} = 2^{-7} = 0.0078... \end{array}$$

Nota: Valores exponentes posibles: $\begin{cases} e=0 \\ e \neq 0 \rightarrow \text{exponente: } e-4=[1:7]-4=-3:3 \end{cases}$

4. ¿Cuál es el valor de x positivo a partir del cual $1+x \neq 1$ en esta representación?

Previo: $\text{eps}(1)=2^{-4}$ (distancia entre mantisas distintas)

$$x > \frac{1}{2} \text{eps}(1) = \frac{1}{2} 2^{-4} = 2^{-5} = 0.0313 \text{ si redondeo al más próximo.}$$

Obs.: Si estrategia de truncamiento, $\text{eps}(1)=2^{-4}=0.0625$

· ¿Cuál es el mayor valor de n (entero) para el que $1+2^{-n} \neq 1$ si approxim. por redondeo al más próximo?

$$n \leq 4$$

· ¿Cuál es el valor de x positivo a partir del cual $2+x \neq 2$ en esta representación?

$$\text{eps}(2)=2^{-4}2=2^{-3} \rightarrow x > \frac{1}{2} \text{eps}(2)=2^{-4}=0.0625$$

$$\sinh(x) = \frac{e^x - e^{-x}}{2}$$

Valores “exactos y aproximados”. Error relativo. Visualización

$$x = 10.^{(-[0:10])}$$

$$V_{\text{exacto}} = \sinh(x); V_{\text{aprox}} = (\exp(x) - \exp(-x)) / 2;$$

$$E_{\text{rel}} = \text{abs}(V_{\text{exacto}} - V_{\text{aprox}}) / \text{abs}(V_{\text{exacto}});$$

$$\text{Cota} = \text{eps}(1) / \sinh(x);$$

$$\loglog(x, V_{\text{exacto}}, 'b', x, V_{\text{aprox}}, 'r'),$$

title('Gráfica de sinh(x): V-exacto (azul), V-aprox (rojo), respecto de x')

$$\loglog(x, E_{\text{rel}}, 'bo', x, Cota, 'ro'),$$

title('Gráfica del error relativo del sinh(x)(azul) y su cota (rojo), respecto de x')

CREACIÓN COTA = $\text{eps}(1) \cdot \text{vex}$

EJERCICIO. Suponemos representación en coma flotante normalizada $x = \pm m2^e$, $m \in [1,2)$

- Calcular la mantisa y el exponente (en esta representación) de $y = \frac{1}{x}$

$$x = \pm m2^e \rightarrow y = \frac{1}{x} = \pm \frac{1}{m} 2^{-e}$$

Sabemos: $m \in [1, 2) \Rightarrow \frac{1}{m} \in \left(\frac{1}{2}, 1\right]$ → No es mantisa normalizada, no está en $[1,2)$, salvo $m=1$

$$y = \frac{1}{x} = \pm \frac{1}{m} 2^{-e} = \begin{cases} \pm \frac{2}{m} 2^{e-1} & \text{si } m \neq 1 \rightarrow \text{MANTISA: } \frac{2}{m} \text{ EXPONENTE: } -e+1 \\ \pm \frac{1}{m} 2^{-e} & \text{si } m=1 \rightarrow \text{MANTISA: } 1 \text{ EXPONENTE: } -e \end{cases}$$

- Calcular la mantisa y el exponente (en esta representación) de $y = \sqrt{x}$

$$x = m2^e \rightarrow y = \sqrt{m} 2^{\frac{e}{2}}$$

¿EXPOENTE ADMISIBLE, N° ENTERO?

- Suponemos e es múltiplo de 2 → $\frac{e}{2}$ n° entero, exponente admisible

Además si $m \in [1, 2) \Rightarrow \sqrt{m} \in [1, \sqrt{2}) \subset [1, 2)$ es la mantisa

- Suponemos e no es múltiplo de 2 → $\frac{e-1}{2}$ n° entero, exponente admisible

Además $2m \in [2, 4) \Rightarrow \sqrt{2m} \in [\sqrt{2}, 2) \subset [1, 2)$ es la mantisa

$$(y = (2m2^{e-1})^{1/2} = (2m)^{1/2} 2^{(e-1)/2})$$

- Número más pequeño con mantisa normalizada:
 $1.00000 \cdot 2^{\text{exp_minimo}} = 1.00000 \cdot 2^{-3} = 2^{-3} = 1/8$
- La separación (eps) entre el 1 y el siguiente número máquina.
 $1.00001 - 1.00000 = 0.00001 = 2^{-5} = 1/32$
- La máxima separación entre dos números máquina consecutivos.
 Se dará en los números con exponente máximo =>
 $(\text{salto mínimo mantisa}) \cdot 2^{\text{max_exp}} = (2^{-5}) \cdot (2^3) = 2^{-2} = 1/4$
- El número (no nulo) más pequeño de la representación.
 Será el más pequeño de los desnormalizados:
 $0.00001 \cdot 2^{-3} = 2^{-5} \cdot 2^{-3} = 2^{-8} = 1/256$

b) Sea el número real $x=0.3$. Obtener el exponente e y mantisa m del número máquina más cercano (para la mantisa dar su expresión en decimal y en binario). Indicar el error relativo de su representación máquina.

$$x = 0.3 = 1.2 / 4 = 1.2 \cdot 2^{-2} \rightarrow e = 2, \text{mantisa } m = 1.2$$

resto
1.2 → 0.2
0.4 → 0.4
0.8 → 0.8
1.6 → 0.6
1.2 → ... y a partir de aquí se repite 1001 $m = 1.001\ 1001\ 1001\dots$

Redondeando a 5 "decimales" binarios: $m = 1.00110 = 1+1/8+1/16 = 1.1875$

El número máquina guardado es $1.1875 \cdot 2^{-2} = 0.2969$

Problema 1. Se considera una representación en coma flotante en base 2. Cada palabra utiliza en memoria los siguientes 8 bits: 3 bits para el exponente, $e = (e_1 e_2 e_3)_2$, y 5 bits para la mantisa, $m = (b_1 b_2 b_3 b_4 b_5)_2$.

Los números máquina \hat{x} representados son los siguientes:

$$\hat{x} = (1.b_1 b_2 b_3 b_4 b_5)_2 \times 2^{e-4}$$

En esta representación:

- ¿Cuántos números máquina hay?
- Calcular los números máquina (en formato decimal) y el contenido de los 8 bits a almacenar en memoria para los siguientes números: v_{\min} valor mínimo, v_{\max} valor máximo y los números reales 1 y 5.5.
- Calcular el $\text{eps}(1)$ y $\text{eps}(5.5)$.
- Sea $\alpha = 0.02$. Calcular los números máquina que se obtienen al realizar las siguientes operaciones: $1 + \alpha$, $5.5 + \alpha$, $(1 + \alpha) + (5.5 + \alpha)$. Justificar.

Problema 1.

En esta representación hay $2^8 = 256$ números máquina.

$$v_{\min} = (1.00000)_2 \times 2^{(000)_2 - 4} = 1 \times 2^{-4} = 2^{-4} = 0.0625$$

$$v_{\max} = (1.11111)_2 \times 2^{(111)_2 - 4} = (2 - 2^{-5}) \times 2^3 = 2^4 - 2^{-2} = 15.75$$

$$1 = (1.00000)_2 \times 2^{(100)_2 - 4} = 1 \times 2^0 = 1$$

$$\text{eps}(1) = 2^{-(n \text{ bits mantisa})} = 2^{-5} = 0.03125$$

$$5.5 = (1.01100)_2 \times 2^{(110)_2 - 4} = (1 + 2^{-2} + 2^{-3}) \times 2^2 = 5.5$$

El siguiente número máquina es:

$$v_{5.5} = (1.01101)_2 \times 2^{(110)_2 - 4} = (1 + 2^{-2} + 2^{-3} + 2^{-5}) \times 2^2 = 5.5 + 2^{-3}$$

$$\text{luego } \text{eps}(5.5) = 2^{-3} = 0.125$$

Sea $\alpha = 0.02$.

Se verifica $\alpha = 0.02 < v_{\min} = 0.0625$, luego:

El número máquina de α es v_{\min} .

El número máquina de $1 + \alpha$ es

$$1 + v_{\min} = (1.00000)_2 2^0 + (1.00000)_2 2^{-4} = (1.00000)_2 2^0 + (0.00010)_2 2^0 = (1.00010)_2 2^0 = 1 + 2^{-4} = \frac{17}{16} = 1.0625.$$

El número máquina de $5.5 + \alpha$ es

$$5.5 + v_{\min} = (1.01100)_2 2^2 + (1.00000)_2 2^{-4} = (1.01100)_2 2^2 + (0.00000)_2 2^2 = 5.5.$$

El número máquina de $(1 + \alpha) + (5.5 + \alpha)$ es

$$(1 + v_{\min}) + (5.5) = (1.00010)_2 2^0 + (1.01100)_2 2^2 = (0.01000)_2 \times 2^2 + (1.01100)_2 \times 2^2 = (1.10100)_2 \times 2^2 = 6.5$$

Problema 1. (5/10 puntos)

Se considera un sistema de números máquina positivos en coma flotante en base 2. Cada número máquina se codifica en una palabra de 6 bits: 2 bits $(e_1 e_2)$ para almacenar la información del exponente y 4 bits para almacenar los dígitos $(b_1 b_2 b_3 b_4)$ de la mantisa, atendiendo a las siguientes representaciones:

$$\text{Si } e = (e_1 e_2)_2 = (0 0)_2 = 0, \quad \hat{x} = (0.b_1 b_2 b_3 b_4)_2 \times 2^0 \quad \text{Número denormalizado}$$

$$\text{Si } e = (e_1 e_2)_2 \neq 0, \quad \hat{x} = (1.b_1 b_2 b_3 b_4)_2 \times 2^{e-1} \quad \text{Número normalizado}$$

- .1. Calcular los números máquina (en formato decimal) y el contenido de los 6 bits que codifican los siguientes números: v_{\min} valor del número máquina mínimo no nulo, v_{\max} valor del número máquina máximo y los números reales 0.5, 1 y 3.5.

¿Qué valor decimal representa el número máquina codificado con $(e_1 e_2) = (10)$ y $(b_1 b_2 b_3 b_4) = (1110)$?

Completar la siguiente tabla:

Nº en formato decimal	Nº máquina	$e_1 \ e_2$	$b_1 \ b_2 \ b_3 \ b_4$
$v_{\min}=$			
$v_{\max}=$			
0.5			
1			
3.5			
		10	1110

- .2. Dar $\text{eps}(1)$, $\text{eps}(3.5)$ y $\text{eps}(6.5)$. Justificar la respuesta.

- .3. Dar una cota del error relativo que se comete al aproximar un número real, en el rango de los números normalizados, por el número máquina normalizado más próximo en el sistema dado.

¿Cuántas cifras decimales significativas se garantizan con este sistema de números normalizados?

1.1.

Menor número máquina no nulo: $e = (00)_2 = 0$; $(b_1 b_2 b_3 b_4) = (0000) \rightarrow v_{\min} = (0.0001)_2 \times 2^0 = 2^{-4} = 1/16 = 0.0$

Mayor número máquina: $e = (11)_2 = 3$; $(b_1 b_2 b_3 b_4) = (1111) \rightarrow v_{\max} = (1.1111)_2 \times 2^{3-1} = 7.75$

$0.5 = (0.1)_2 = (0.100)_2 2^0 \rightarrow e = 0 = (00)_2$, $(b_1 b_2 b_3 b_4) = (1000)$

$1 = 2^0 = (1.0000)_2 2^0 \rightarrow e = 1 = (01)_2$, $(b_1 b_2 b_3 b_4) = (0000)$

$3.5 = (11.1)_2 = (1.1100)_2 2^1 \rightarrow e = 2 = (10)_2$, $(b_1 b_2 b_3 b_4) = (1100)$

$e = (10)_2 = 2$ y $(b_1 b_2 b_3 b_4) = (1110) \rightarrow \hat{x} = (1.1110)_2 \times 2^{2-1} = 3.75$

1.2.

Epsilon de un número máquina x es la distancia entre x y el número máquina x_d inmediatamente mayor que x .

Sabemos que $\text{eps}(x) = x_d - x = \text{eps}(1)2^{\text{exponente}(x)}$.

- $\text{eps}(1) = (1.0001)_2 - (1.0000)_2 = 2^{-4} = 0.0625$.
- $\text{eps}(3.5) = (1.1100)_2 2^1 - (1.1101)_2 2^1 = 2^{-4} 2^1 = 2^{-3} = 0.125$.
- $\text{eps}(6) = (1.1000)_2 2^2 - (1.1001)_2 2^2 = 2^{-4} 2^2 = 2^{-2} = 0.25$.

1.3. Dar una cota del error relativo que se comete al aproximar un número real x en el rango de los números normalizados por el número máquina normalizado \hat{x} más próximo:

$$\text{Error relativo} = \frac{|x - \hat{x}|}{|x|} \leq \frac{|\hat{x}_i - \hat{x}_d|}{2|x|} \leq \frac{1}{2} \text{eps}(1) = 2^{-4} / 2 = 2^{-5} = 0.03125$$

donde \hat{x} es el número máquina más próximo a x , y \hat{x}_i y \hat{x}_d son los números máquina inmediatamente menor y mayor que x , respectivamente.

El número N de cifras decimales significativas garantizadas atendiendo a esta cota del error relativo es:

$$N = \text{floor}(-\log_{10}(\text{Error relativo})) \geq \text{floor}(-\log_{10}(2^{-5})) = \text{floor}(1.5051) = 1$$

Por tanto, está garantizada al menos 1 cifras decimal significativa.

Ejercicio 1: Dado un sistema de representación en coma flotante donde los números máquina \hat{x} se almacenan de la siguiente forma:

$$\hat{x} = (\pm 1.a_1a_2\dots a_m)2^e$$

$$e \in \{0, -1, -2, \dots, -n\} \quad a_i \in \{0, 1\} \forall i \quad n, m > 0$$

Suponiendo que $m=3$ y $n=2$,

- a)** Escribir la representación $(\pm 1.a_1a_2\dots a_m) \cdot 2^e$ y el valor decimal de los siguientes números máquina:
- Valor mínimo mayor que cero (V_{\min})
 - Valor máximo (V_{\max})
 - Números reales 1 y 0.4

$$V_{\min} = +1.000 \cdot 2^{-2} = 0.25$$

$$V_{\max} = +1.111 \cdot 2^0 = 1.875$$

$$1 = +1.000 \cdot 2^0 = 1.0$$

$$0.4 = (0.0110\dots)_2 \sim +1.100 \cdot 2^{-2} = 0.375 \quad (\text{si truncamos})$$

$$0.4 = (0.0110\dots)_2 \sim +1.101 \cdot 2^{-2} = 0.40625 \quad (\text{si redondeamos})$$

- b)** Calcular $\text{eps}(1)$ y dar una cota del error relativo.

El siguiente valor representable al 1 es $+1.001 \cdot 2^0$, luego $\text{eps}(1) = +1.001 \cdot 2^0 - +1.000 \cdot 2^0 = 0.125$. Además $\text{eps}(1) = 2^{-(M-1)}$ y en este caso $M = m+1 = 3+1 = 4$ y $\text{eps}(1) = 2^{-(3)} = 0.125$

$$\text{La cota del error relativo es } \frac{\text{eps}(1)}{2} = 0.0625$$

- c)** ¿Cuál es el error relativo al representar el número real 0.4? ¿Cuántas cifras decimales significativas se consiguen?

Para el número máquina obtenido truncando (0.375) el error relativo es:

$$\text{Error_relativo} = \left| \frac{0.4 - 0.375}{0.4} \right| = 0.0625$$

$$\text{Número_cifras_significativas} = -\log_{10}(0.0625) \approx 1$$

- d)** Para cualquier m y n , ¿Cuántos números máquina se pueden representar?

Signos posibles: 2
 Mantas posibles: 2^m Total: $2 \times 2^m \times (n+1)$
 Exponentes posibles: $n+1$

Ejercicio 1: Una representación en coma flotante (en base 2) dedica 6 bits a la mantisa ($m = 1.m_1m_2m_3m_4m_5m_6$). La representación no usa números desnormalizados y el rango de los exponentes posibles es entre -8 y 8.

a) Número mínimo de la representación:

$$x_{\min} = (\text{mantisa mínima}) \cdot 2^{\exp_{\min}} = 1.000000 \cdot 2^{-8} = 1/256 \sim 0.0039$$

Separación entre el 1 y el siguiente número máquina:

$$\text{Espaciado entre } 1.000000 \text{ y } 1.000001 = 0.000001 = 2^{-6} = 1/64 = 0.0156$$

$$\text{Error relativo de la representación: } E_{\text{rel}} \leq \text{eps}/2 = 1/128 \sim 0.0078$$

Máxima separación entre números máquina: se dará en los números con exponente máximo => (espaciado mantisa) · $2^{\max_{\exp}} = (2^{-6}) \cdot (2^8) = 4$

b) Sea el $x=3.2$. Expresarlo como $m \cdot 2^e$ dando su exponente e y su mantisa m . Error absoluto y relativo entre el número real original y su representación máquina.

$$x = 3.2 = 1.6 \cdot 2 \rightarrow \text{exponente} = 1, \text{mantisa (decimal)} m = 1.6$$

resto

$$1.6 \rightarrow 0.6 \quad m = 1.100110011001\dots$$

$$1.2 \rightarrow 0.2$$

$$0.4 \rightarrow 0.4 \quad \text{guardado con 6 "decimales" binarios: } m = 1.100110$$

$$0.8 \rightarrow 0.8$$

1.6 → ... y a partir de aquí se repite 1001

$$\text{El número máquina es } 2 \cdot (1 + 1/2 + 1/16 + 1/32) = 2 + 1 + 1/8 + 1/16 = 3.1875$$

$$\text{Error absoluto} = |3.2 - 3.1875| = 0.0125. \quad E_{\text{relativo}} = 0.0125/3.2 \sim 0.0039$$

c) Sea $a=1/256$ (un número válido en esta representación). Si la representación redondea, indicad (justificando) el resultado de las siguientes operaciones:

$$(1+a) \quad 1+(a+a+a) \quad (1+a+a+a+a)$$

Fijaros que $a = 1/256 = \text{mínimo número representable} = \text{eps}/4$ ($\text{eps} = 1/64$).

- $(1+a) = 1$ ($1+\text{eps}/4$ no llega a $1+\text{eps}/2$ para "saltar" al siguiente nº)
- $1+(a+a+a) = 1+\text{eps}$ ($3a=3\text{eps}/4$, $1+0.75\text{eps}$ "salta" al siguiente nº = $1+\text{eps}$)
- $(1+a+a+a+a) = 1$. Ante esta expresión el ordenador hace la primera operación $(1+a)$ y el resultado se queda en 1 como en el primer caso. Luego añade el siguiente a pero vuelve a pasar lo mismo, etc.

En este último caso sería mucho mejor hacer $1+(a+a+a+a)$, sumando primero los términos pequeños entre si ($4a$) antes de añadirlos a la unidad, evitando en lo posible sumar números de magnitudes muy distintas.

MODELO A

1. Hacer un programa para calcular $\text{eps}(2^{-1})$ escrito en formato 2^{p} . Dar el valor de p.

Usar el comando `fprintf` para mostrar los valores $\frac{1}{2}$ y el número máquina siguiente a $\frac{1}{2}$ con formato `%0.17f`, del tipo: `1/2: 0.50000000000000000`, siguiente:

¿Cuál es el número máquina inmediatamente anterior a 1?

2. Se van a calcular valores aproximados de $\cos(\pi/4)$ sumando términos de la serie

$$\cos(x) = \sum_{n=0}^{\infty} (-1)^n x^{2n} / (2n)! \quad (\text{con } x=\pi/4)$$

hasta obtener un valor aproximado que produzca 14 cifras decimales significativas. Para ello, para cada k desde 0 hasta llegar a la precisión pedida el código debe:

- calcular el valor aproximado (`vap`) resultante de sumar los $k+1$ primeros términos de la serie;

$$vap = \sum_{n=0}^k (-1)^n (\pi/4)^{2n} / (2n)!$$

- calcular el correspondiente error relativo y el número de cifras decimales significativas que produce `vap`;
- imprimir estos resultados usando `fprintf` con los formatos indicados: el valor de `k` (`%d`), el valor aproximado `vap` (`%0.17f`), el correspondiente error relativo (`%0.2e`) y nº de cifras decimales significativas (`%2d`) en líneas separadas para cada valor de `k`, del tipo:

`k:0, vap:1.0000000000000000, erel:4.14e-01, cifras:0`

`k:1, vap:0.69157486246595756, erel: ..., cifras:...`

¿Cuántos sumandos son necesarios para alcanzar la precisión pedida? Indicar aproximadamente cuántos dígitos decimales significativos se ganan al aumentar un sumando en el cálculo de la aproximación.

```

1. p=0;
while ((1/2)+2^(-p) <= (1/2))
    p=p+1;
end
p_pedido=p-1
fprintf('1/2: %.0.17f, siguiente: %.0.17f\n', 1/2, 1/2+2^(-p-1))

```

```

2.
x=pi./4;k=0;vap=1;erel=abs((vap-cos(x))./cos(x));cif=floor(-log10(erel));
while(cif<14)
    vap=sum((-1).^(0:k)).*((x).^2*(0:k))./factorial(2*(0:k));
    erel=abs((vap-cos(x))./cos(x)); cif=floor(-log10(erel));
    fprintf('k:%d,vap:.17f,erel:.0.2e, cifras:%d\n',k,vap,erel,cif)
    k=k+1;
end

```

Otras opciones son posibles, por ejemplo:

```

x=pi/4;k=0;vap=1;erel=abs((vap-cos(x))/cos(x));cif=floor(-log10(erel));
while(cif<14)
k=k+1;
vap=vap+((-1)^k)*((x)^(2*k))/factorial(2*k); erel=abs((vap-cos(x))/cos(x)); cif=floor(-log10(erel));
fprintf('k:%d,vap: %.17f,erel: %0.2e, cifras: %d\n',k,vap,erel,cif)
end

```

```

k:0,vap:1.0000000000000000,erel:4.14e-01,cifras:0
k:1,vap:0.69157486246595756,erel:2.20e-02,cifras:1
k:2,vap:0.70742920670977305,erel:4.56e-04,cifras:3
k:3,vap:0.70710321482284566,erel:5.04e-06,cifras:5
k:4,vap:0.70710680568329420,erel:3.46e-08,cifras:7
k:5,vap:0.70710678107192471,erel:1.62e-10,cifras:9
k:6,vap:0.70710678118693626,erel:5.50e-13,cifras:12
k:7,vap:0.70710678118654646,erel:1.57e-15,cifras:14

```

Con 8 sumandos se calcula $\cos(\pi/4)$ con la precisión pedida. En los datos impresos se observa que un sumando en el cálculo del valor aproximado se ganan dos cifras decimales significativas (aprox.

```

3.
k=1:54; n=2.^-k;x=(pi/4)+n;g=2*((cos(x)).^2)-1;f=cos(2*x); f(end)
erel=abs((f-g)./f); ncif=floor(-log10(erel)); ncif(2)
[max_arel,p_maxerel]=max(erel)
x_maxerel=x(p_maxerel)
[maxcif,p_maxcif]=max(ncif)
x_maxcif=x(p_maxcif)
subplot(1,2,1), semilogy(erel, "r");
subplot(1,2,2), plot(ncif,"g");

```

```
f(end)= 6.1232e-17, ncif(2)=15  
max_eref = 2.6263; p_maxeref = 54; x_maxeref = 0.7854  
maxcif = Inf; p_maxcif = 1;x_maxcif = 1.2854;
```

$$X_{\min} = (\text{mantisa mínima}) \cdot 2^{\text{exp mínimo}} = 1.000000 \cdot 2^{-8} = 1/256 \approx 0.0039$$

$$\text{eps} = \text{espaciado entre } 1.000000 \text{ y } 1.000001 = 0.000001 = 2^{-6} \approx 0.0156$$

$$E_{\text{rel}} \leq \text{eps}/2 = 1/128 \approx 0.0078$$

$$\text{Máx separación} = (\text{eps}) \cdot 2^{\text{exp máximo}} = 2^{-6} \cdot 2^8 = 4$$

EREL = $2^{N^{\circ} \text{ MAS PEQUEÑO}} * 2$ // eps(=espaciado mínimo entre $2n^{\circ}$ con bit positivo)/2
(= $2^{\text{- ultimo número mantisa}}$)

$$\text{EREL} = \text{EPS}/2$$