Post P3, pre P4

Bruno Juliá-Díaz (brunojulia@ub.edu)

Dpto. Física Quàntica i Astrofísica

Facultat de Física

Universitat de Barcelona

Curso 2016/2017

Post Práctica 3, Consideraciones globales

+ Problemas variados de estructura

+ Problemas con subrutinas, functions, programa principal

+ IMPLICIT NONE

+ Problemas en la forma de encarar la práctica a partir de la prepráctica

+ Problemas, menores, en la programación de los métodos

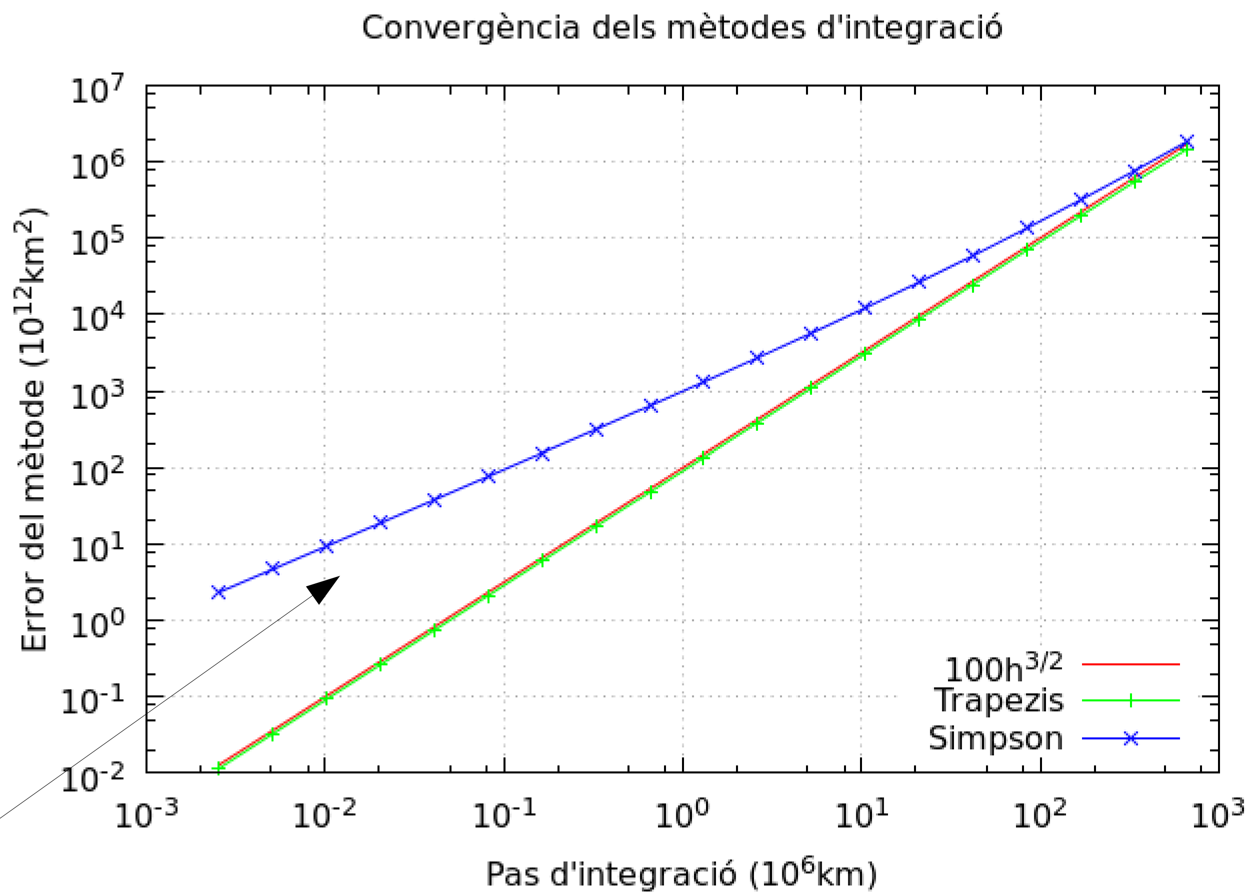
+ Problemas al trabajar con unidades dadas

+ Problemas con la notación E, 5*10**6 = 5.D6

+ Pocos utilizais las opciones de compilación "mejoradas"

Fallos interesantes (1)

```
XI = Z - L/2.D0
XF = Z + L/2.D0
NVAL=2**M
H= (XF-XI)/DBLE(NVAL)

X=XI
FINT=0.D0

FP=0.D0
FS=0.D0

DO K=2,NVAL-1

X=X+H !PUNTS INTERNS
```

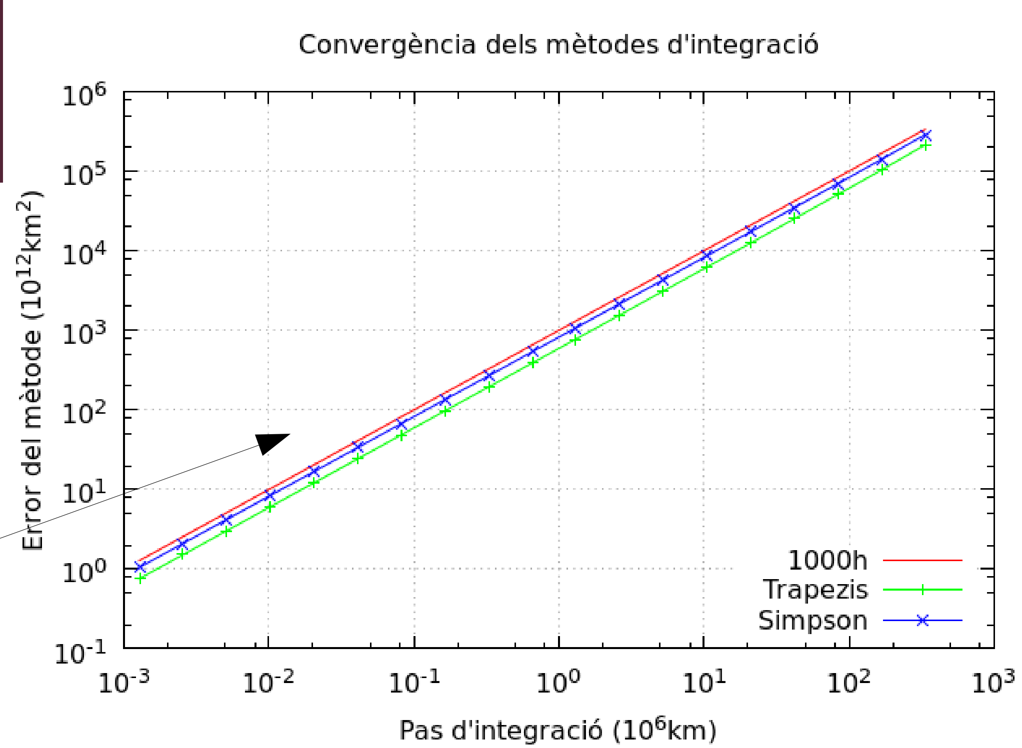


Simpson más lento
que trapecios

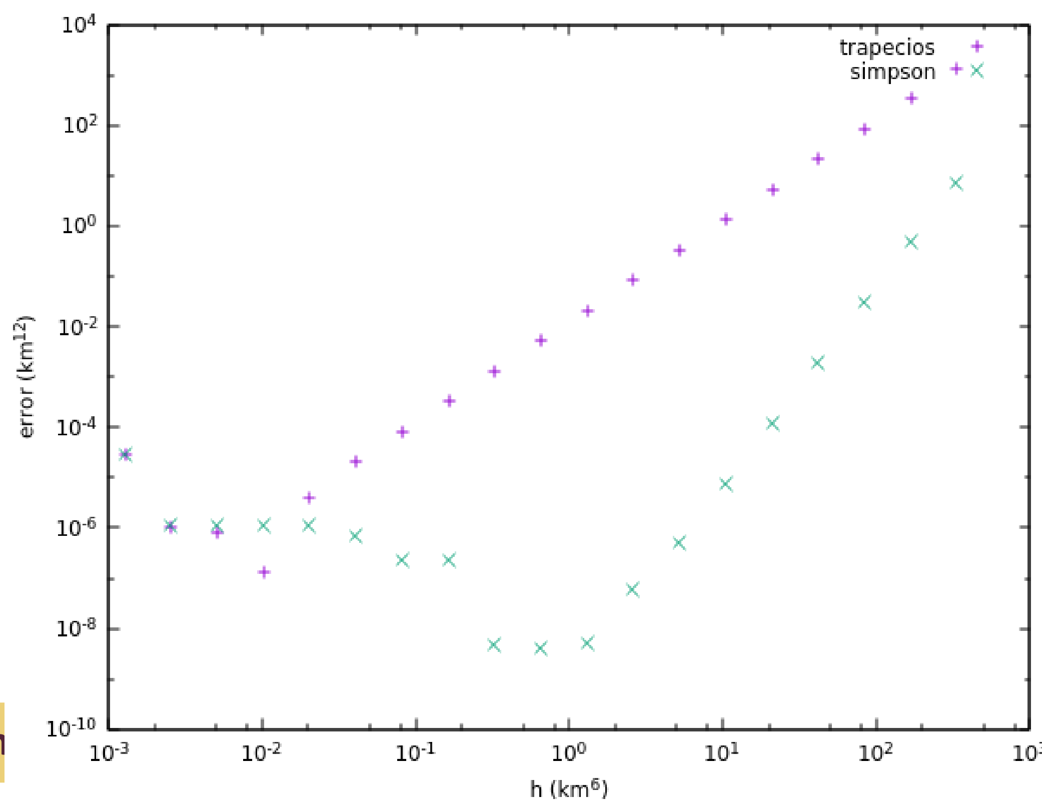
Fallos interesantes (1)

```
XI = Z - L/2.D0
XF = Z + L/2.D0
NVAL=2**M
H= (XF-XI)/DBLE(NVAL)

X=XI
FINT=0.D0

FP=0.D0
FS=0.D0

DO K=1,NVAL-1

X=X+H !PUNTS INTERNS
```

Simpson igual de rápido que trapecios en la segunda integral

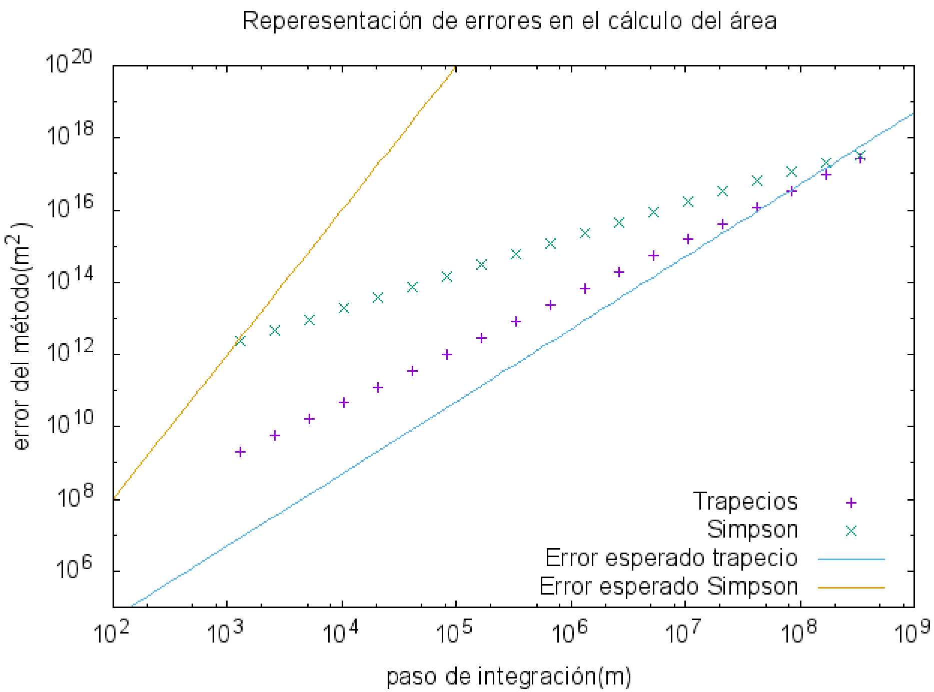


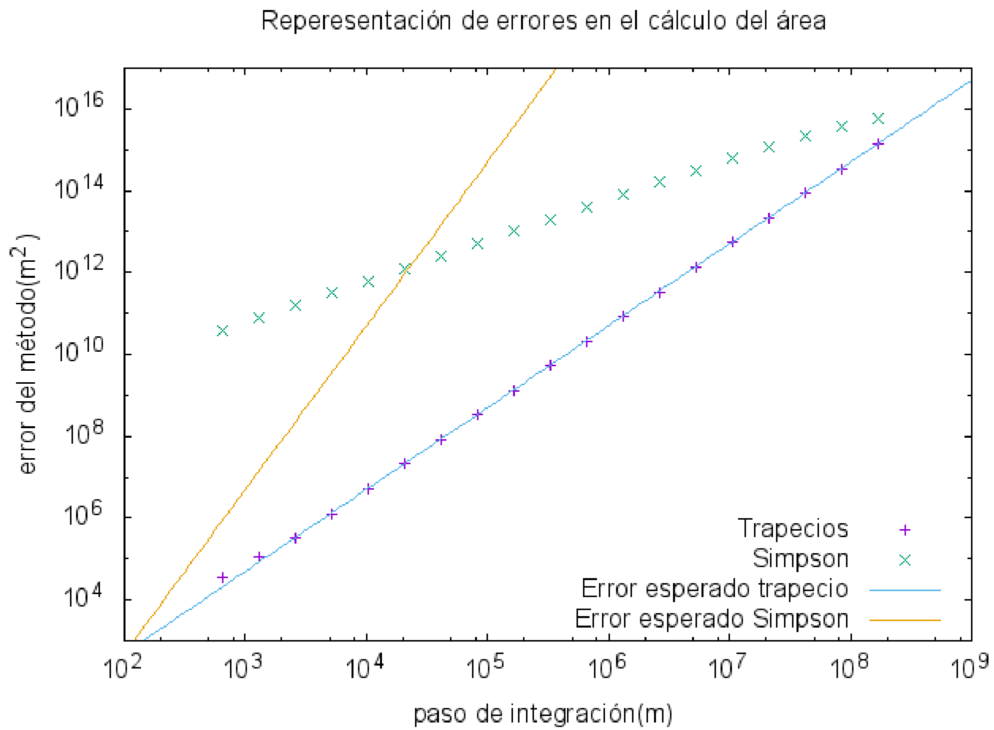
Corregido →



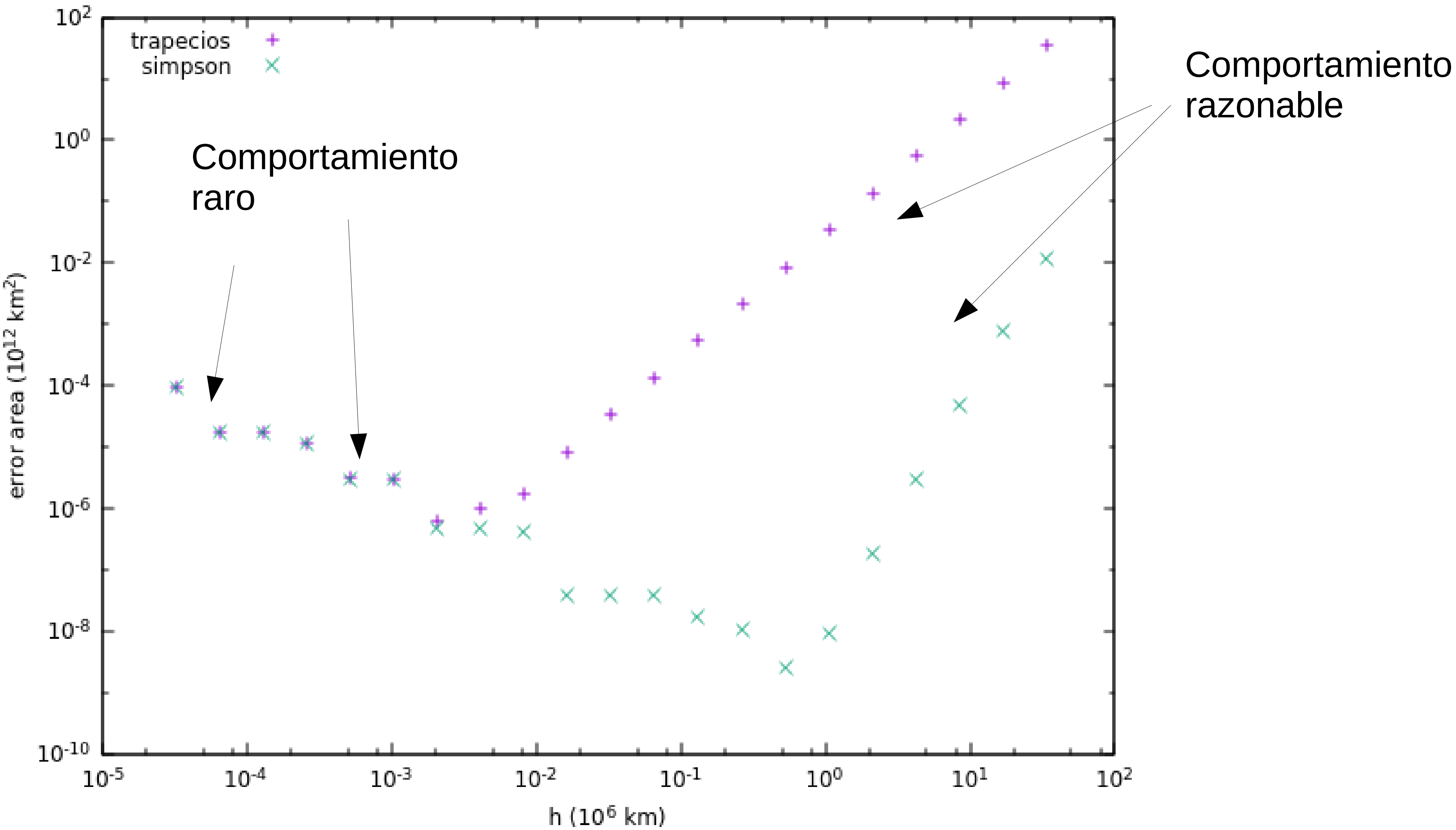
Fallos interesantes (1)

```
xi=z-L/2.d0
x1=z-L/2.d0 + h
xf=z+L/2.d0
if (im.eq.2) then
    pares=0
    impares=0
    do i=2,k-1
       resto=mod(i,2)
       xi=z- L/2.d0 +i*hXI = Z - L/2.D0
```





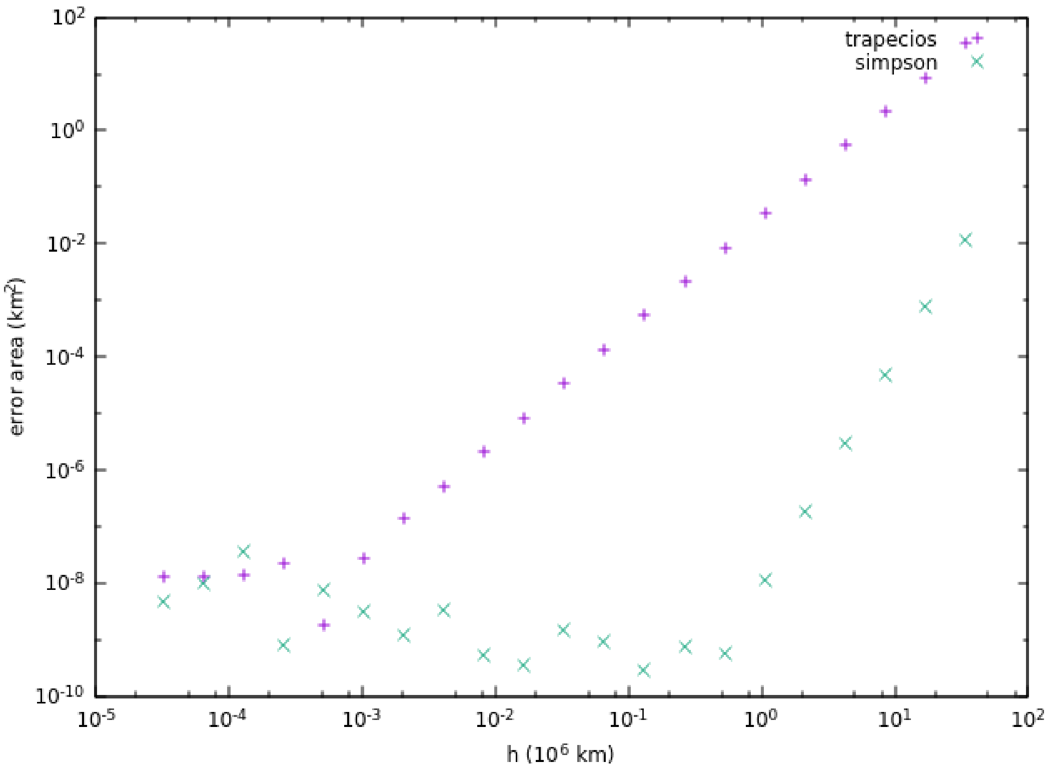
Fallo sutil y habitual (1)



Fallo sutil y habitual (1)

```
   DO I=1,NI-1
          X = X + H
          VAL = VAL + 2 * FCN(X)
    ENDDO
        VAL = VAL * (H / 2.d0)
```

```
   DO I=1,NI-1
          X = A + I*H
          VAL = VAL + 2 * FCN(X)
    ENDDO
        VAL = VAL * (H / 2.d0)
```



Error notación E

```
a = 1376.3d0*(10.d0**6.d0)
b = 542.617d0*(10.d0**6.d0)
```

→

```
a = 1376.3d6
b = 542.617d6
```

Errores de estructura (1)

Argumentos GENERALES para cualquier integral de cualquier función

```
c23456789012345678901234567890123456789012345678901234567890123456789012
      SUBROUTINE MYINTEGRATOR(Z, b, m, im, val, fcn)
          IMPLICIT NONE
!Definim les variables d'entrada, sortida i del mètode
          DOUBLE PRECISION Z,h,Fpar,Fimp,d,c,Ftrap,fcn,val,x,b
          INTEGER n,im,i,j,m,k
!Donem un valor a les variables que utilitzem al mètdode
          n=2.d0**m
          c=-3.d0*b
          d=-2.d0*b
```

Definiciones del problema en particular!!

Errores de estructura (2)

```
program practica
implicit none
external elipse
real*8 elipse, YCrommelin, a
a = YCrommelin(1.d0)
end program
```

Programa principal de difícil comprensión

```
C      funcion que describe la elipse. x ha de pertenecer al intervalo
C      [-4b, 0] para que el resultado sea real
        real*8 function elipse(x)
          implicit none

          real*8 x, a, b

          a = 1.3763d3
          b = 5.42617d2

          if ((x.LT.(-4*b)).OR.(x.GT.0.d0)) then
            elipse = 0.d0
          else
            elipse = a*(1.d0-((x+2.d0*b)/b)**2.d0)**0.5d0
          end if

          return
        end function
```

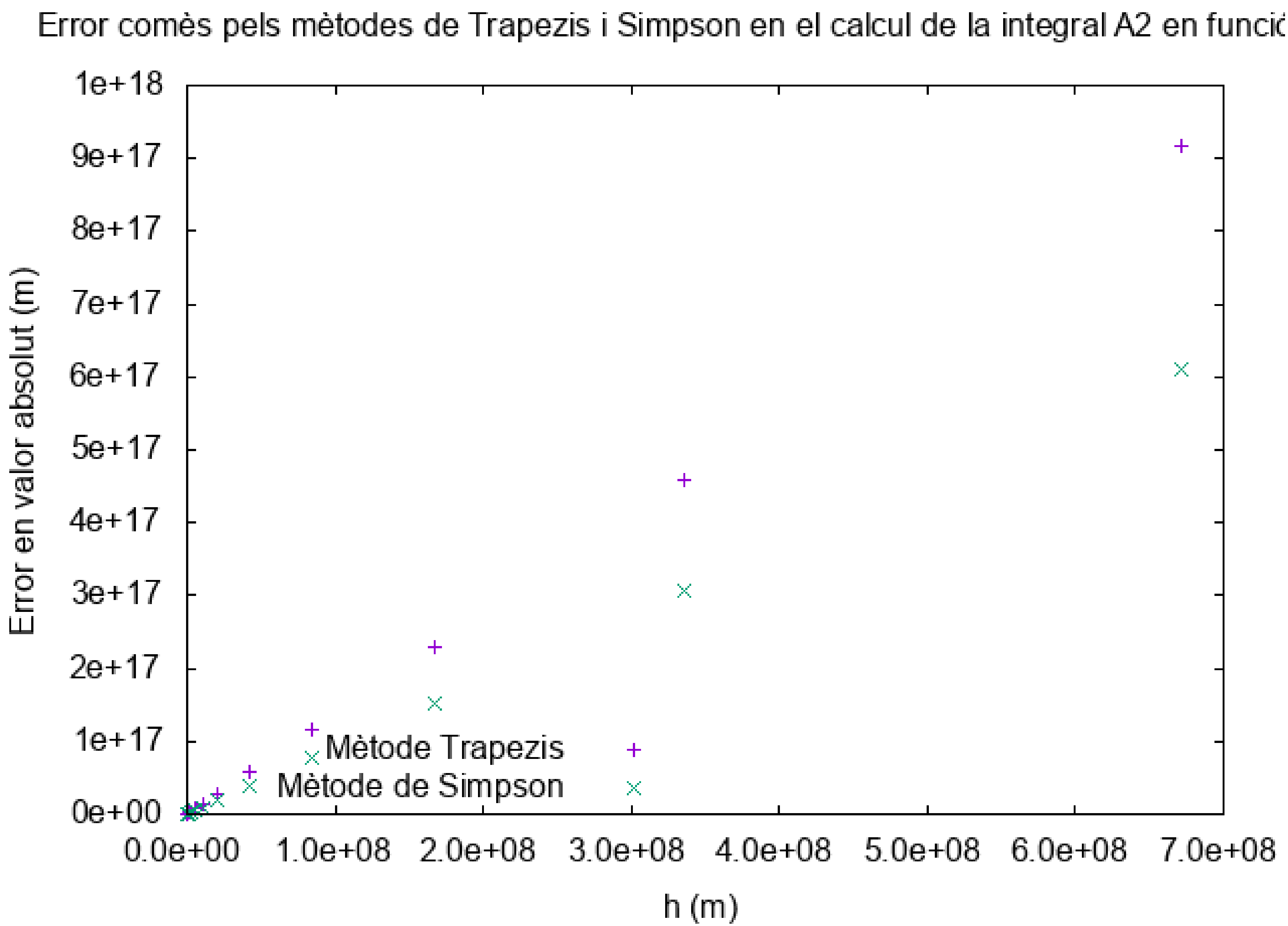
Programa principal de difícil comprensión

Errores de estructura (2)

```
   real*8 function YCrommelin(x)
          implicit none
          external elipse
          real*8 x, h, at, asi, centro, long, b, a, elipse
          integer m

          a = 1.3763d3
          b = 5.42617d2
C     2.a, abrimos el fichero y escribimos lo pedido
C     Para la integral pedida z seria -5/2*b y
C     L seria b

          centro = -5.d0*b/2.d0
          long = b
          open(1, file="P3-2016-c2-res1.dat")
          centro = -5.d0*b/2.d0
          do m = 4, 22, 1

            h = long/dble(2**m)

            call myintegrator(centro,long,m,1,at,elipse)
            call myintegrator(centro,long,m,2,asi,elipse)
            write(1,100)h,4*at,4*asi
          end do
          close(1)
C     2.c, calculamos el area pedida
          open(2, file="P3-2016-c2-res2.dat")

          centro = -2.25d0*b
          long = 0.25d0*b
          do m = 4, 22, 1

            h = long/dble(2**m)
            call myintegrator(centro,long,m,1,at,elipse)
            call myintegrator(centro,long,m,2,asi,elipse)

            write(2,100)h,at,asi
          end do
          close(2)

   100    format(e20.14, "    ", e20.14, "    ", e30.14)
          YCrommelin = 5.0d0
          return
        end function
```

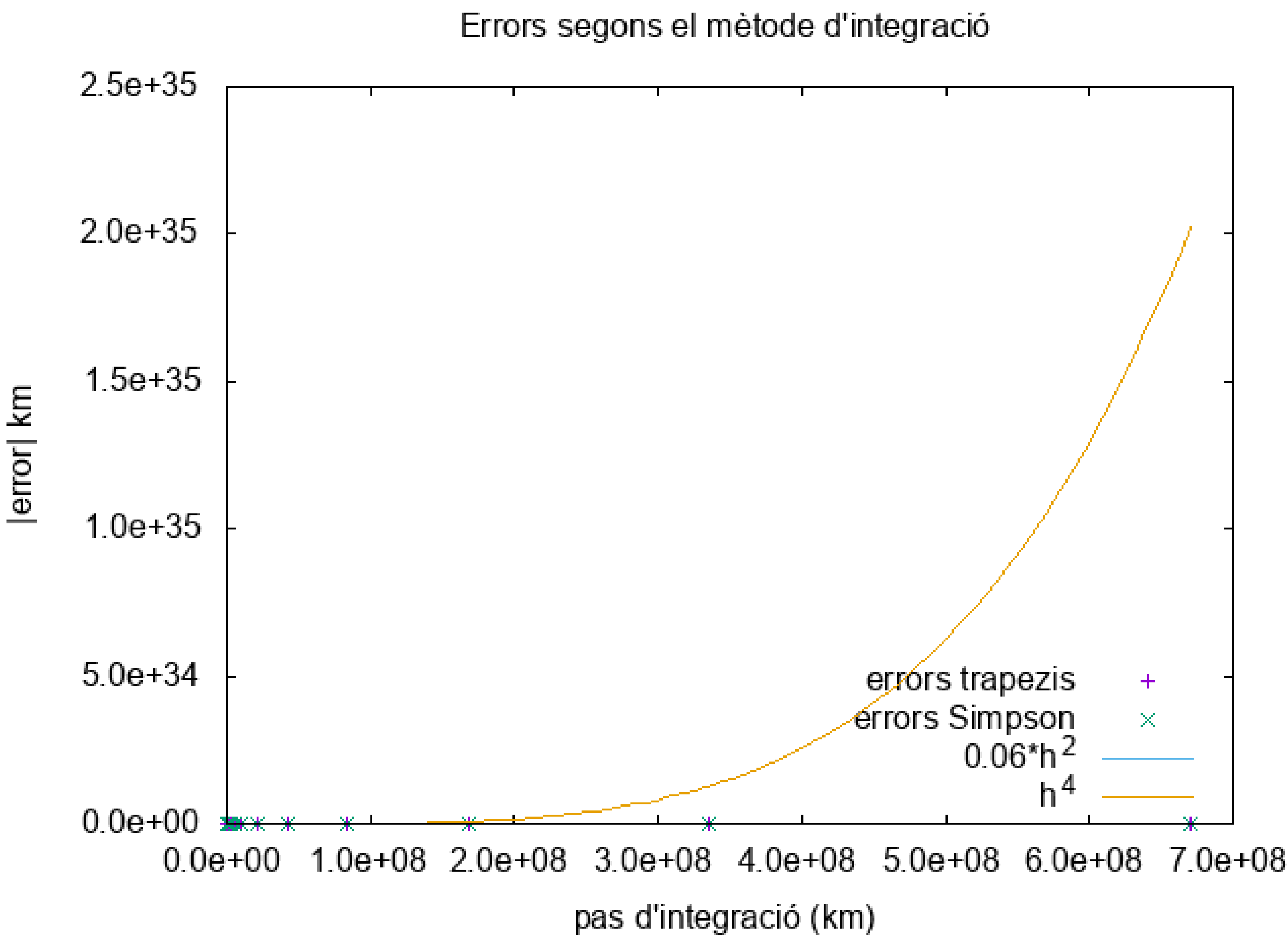
Programa principal en una function (resultados correctos)

Figuras (3)



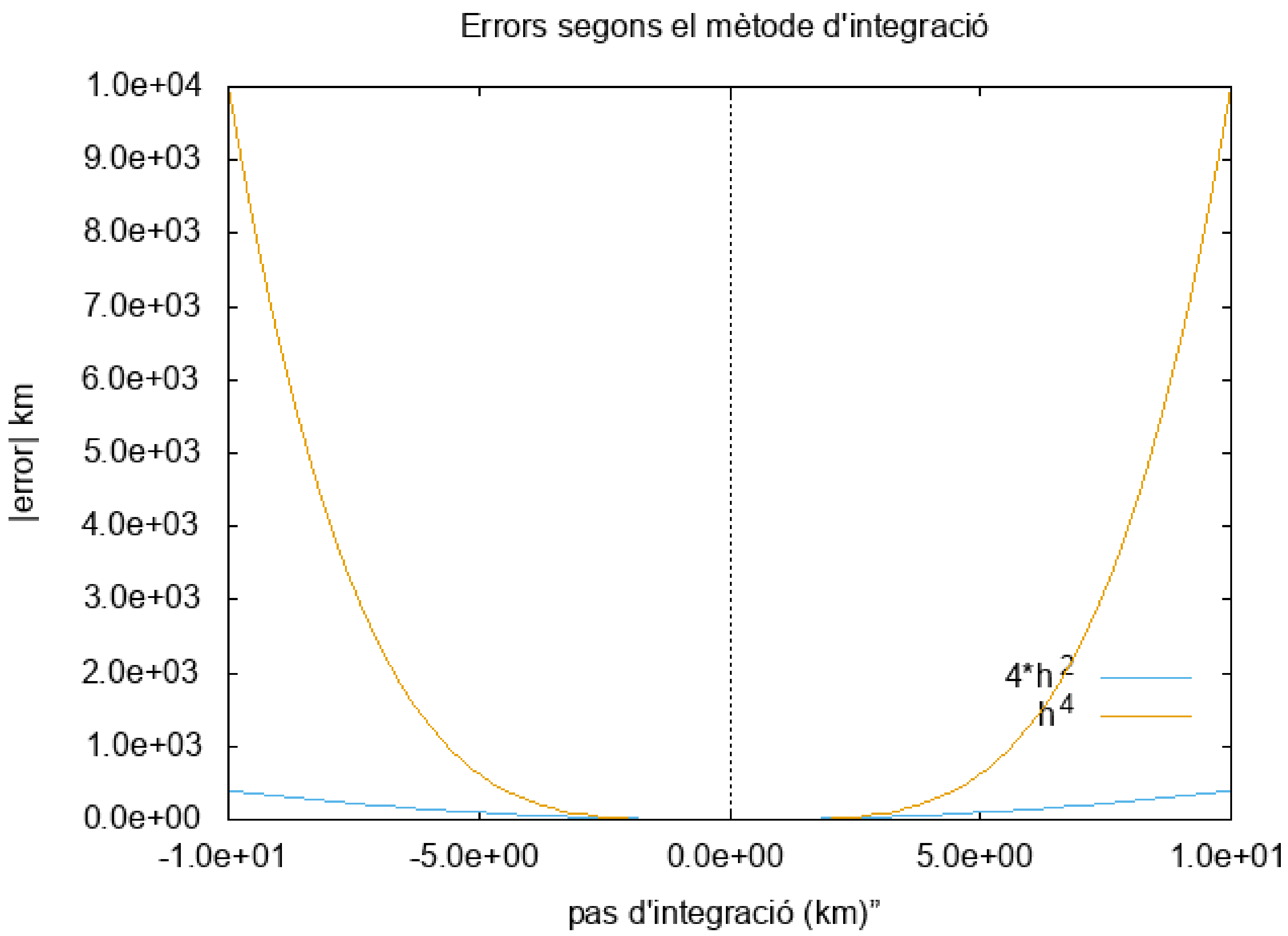
Error comès pels mètodes de Trapezis i Simpson en el calcul de la integral A2 en funció
1e+18
9e+17
8e+17
7e+17
6e+17
5e+17
4e+17
3e+17
2e+17
1e+17
0e+00
Error en valor absolut (m)
0.0e+00
1.0e+08
2.0e+08
3.0e+08
4.0e+08
5.0e+08
6.0e+08
7.0e+08
h (m)
Mètode Trapezis
Mètode de Simpson

Figuras (3)



B. Juliá-Díaz, Física Computacional 2016/2017

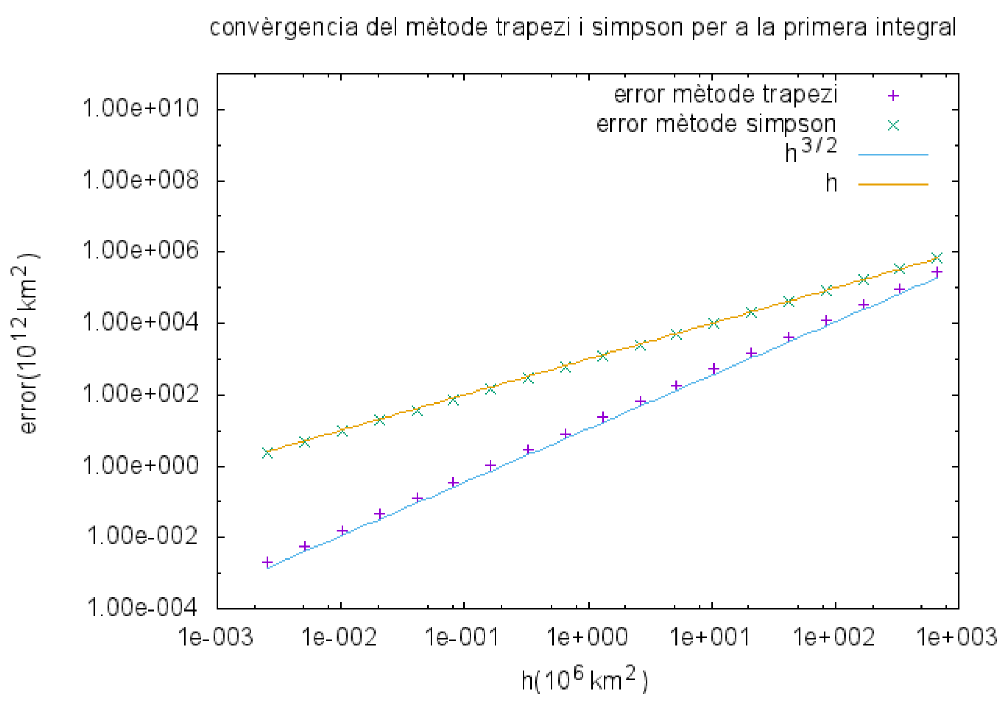
Figuras (3)

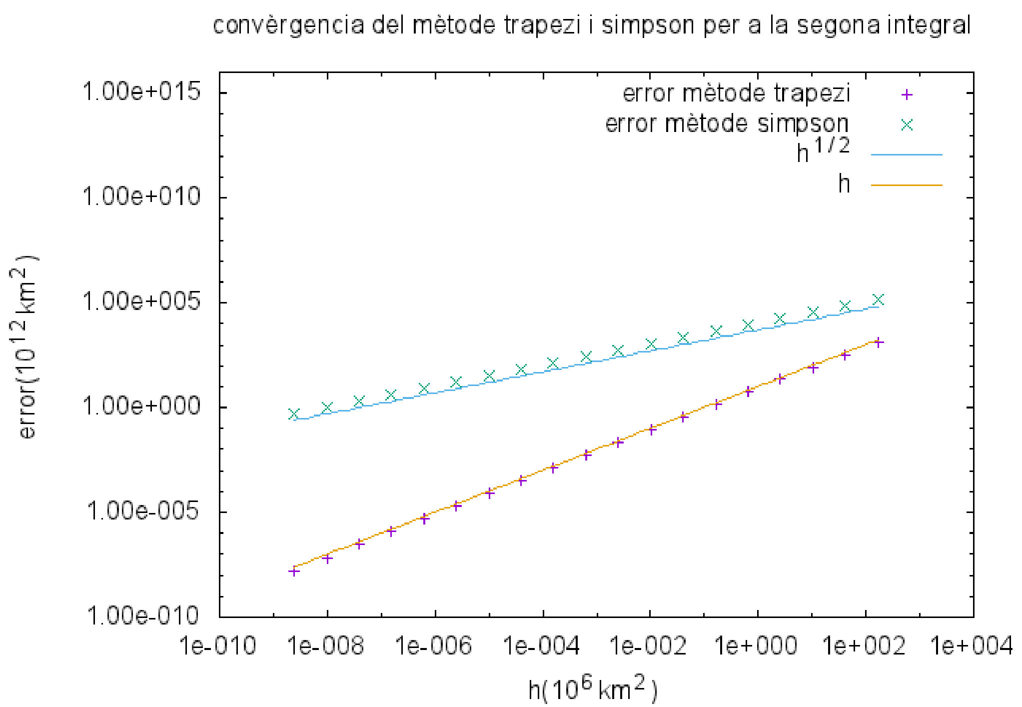


Errors segons el mètode d'integració
1.0e+04
9.0e+03
8.0e+03
7.0e+03
6.0e+03
5.0e+03
4.0e+03
3.0e+03
2.0e+03
1.0e+03
0.0e+00
|error| km
-1.0e+01
-5.0e+00
0.0e+00
5.0e+00
1.0e+01
pas d'integració (km)"
4*h^2
h^4

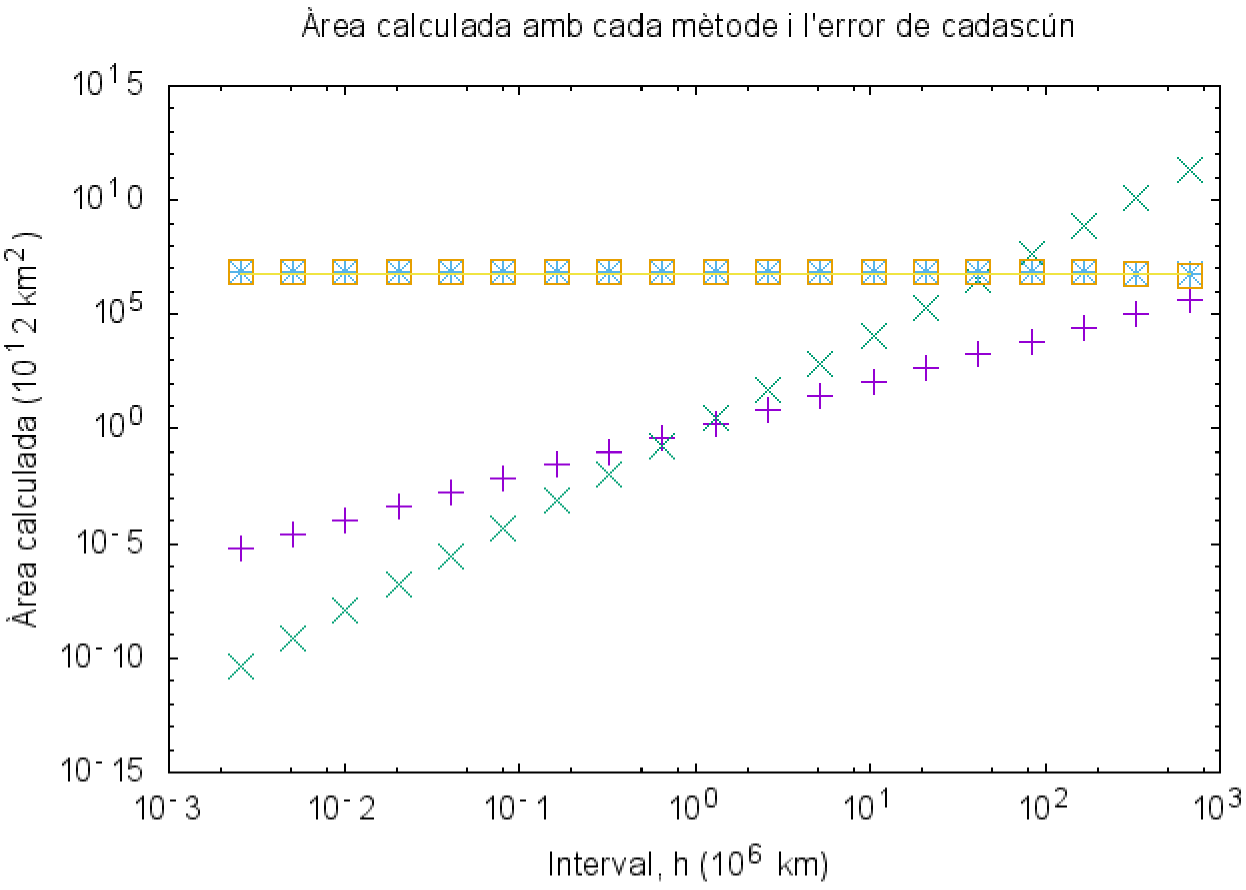
>

Figuras (1)


convèrgencia del mètode trapezi i simpson per a la primera integral
error mètode trapezi
error mètode simpson
h3/2
h
error(10¹² km²)
h(10⁶ km²)


convèrgencia del mètode trapezi i simpson per a la segona integral
error mètode trapezi
error mètode simpson
h1/2
h
error(10¹² km²)
h(10⁶ km²)

Figuras (1)


Àrea calculada amb cada mètode i l'error de cadascún
Àrea calculada (10^{12} km^2)
Interval, h (10^6 km)
10^{15}
10^{10}
10^5
10^0
10^{-5}
10^{-10}
10^{-15}
10^{-3}
10^{-2}
10^{-1}
10^0
10^1
10^2
10^3

Práctica 4

— Nom del programa principal **P4-2016.f**.
Precisió de reals: **double precision**.
Tots els outputs amb 12 xifres significatives, p.ex. format(e20.12)

La pràctica consistirà a estudiar aspectes de la transició líquid-gas amb l'equació de Van der Waals. El dia de la pràctica haureu de fer servir parts del codi que desenvolupeu a continuació.

+ En la práctica consideraremos un gas descrito por la ecuación de Van der Waals que describe la transición

Líquido gas:

$$\left(p + \frac{n^2 a}{V^2}\right)(V - nb) = nRT$$

En unidades reducidas:

$$P = \frac{8T}{3V - 1} - \frac{3}{V^2}\,.$$

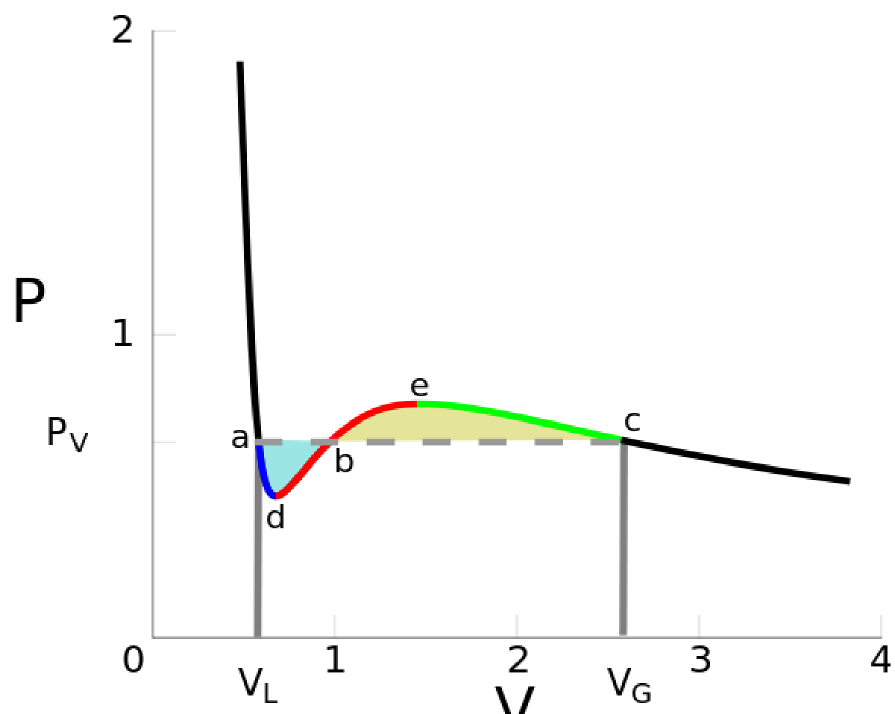


Figura de wikipedia

Práctica 4, apartado 1

1) Escriu dues subrutines, **newtonraphson(x0,eps,fun,dfun,niter,ierr,xarrel)** i **biseccio(A,B,eps,fun,niter,ierr,xarrel)** que trobin una arrel de la funció **fun(x)**.

Els inputs són:

- **x0**, punt inicial de Newton Raphson.
- **A**, **B** punts inicials de bisecció.
- **eps**, precisió desitjada.
- **fun(x)** i **dfun(x)**, funció considerada i la seva derivada, respectivament.

Els outputs,

- **ierr**, la subrutina ha convergit bé (**ierr**=0), o no ha convergit o hi ha algun problema **ierr**=1.
- **niter**, nombre d'iteracions per aconseguir la precisió.
- **xarrel**, valor de l'arrel

Práctica 4, apartado 1

+ En clase hemos visto explícitamente como programar ambos métodos
+ Un problema que os encontrareis:

```
IMPLICIT NONE
 INTEGER A,B,C
 CALL SUMA(1,2,C)
 PRINT*,C
 END

 SUBROUTINE SUMA(A,B,C)
 IMPLICIT NONE
 INTEGER A,B,C

  C=A+B
  A=1
 end
```

```
Program received signal SIGBUS:
Access
to an undefined portion of a memory
object.

Backtrace for this error:
#0  0x10609db99
#1  0x10609cf65
#2  0x7fff85df4f19
#3  0x106094d1a
#4  0x106094d8f
#5  0x106094dcc
Bus error: 10
```

```
IMPLICIT NONE
 INTEGER A,B,C
  A=1
  B=2
 CALL SUMA(A,B,C)
 PRINT*,C
 END

 SUBROUTINE SUMA(A,B,C)
 IMPLICIT NONE
 INTEGER A,B,C

  C=A+B          b
  A=1
 end
```

Correcto

```
Program received signal SIGSEGV: Segmentation fault
- invalid memory reference.

Backtrace for this error:
#0  0x7F347AEA6E08
#1  0x7F347AEA5F90
#2  0x7F347AAF749F
#3  0x4007AE in suma_
#4  0x4007D7 in MAIN__ at tast.f:?
Segmentation fault (core dumped)
```

Práctica 4

2) Per a testejar les subrutines bisecció i newtonraphson:

   a) Considera el polinomi de grau 3 amb $v \in [0, 4]$.

$$\mathrm{Po}(v) = -\frac{21793}{1000} + \frac{2439}{100}v - \frac{87}{10}v^2 + v^3 \tag{0.15}$$

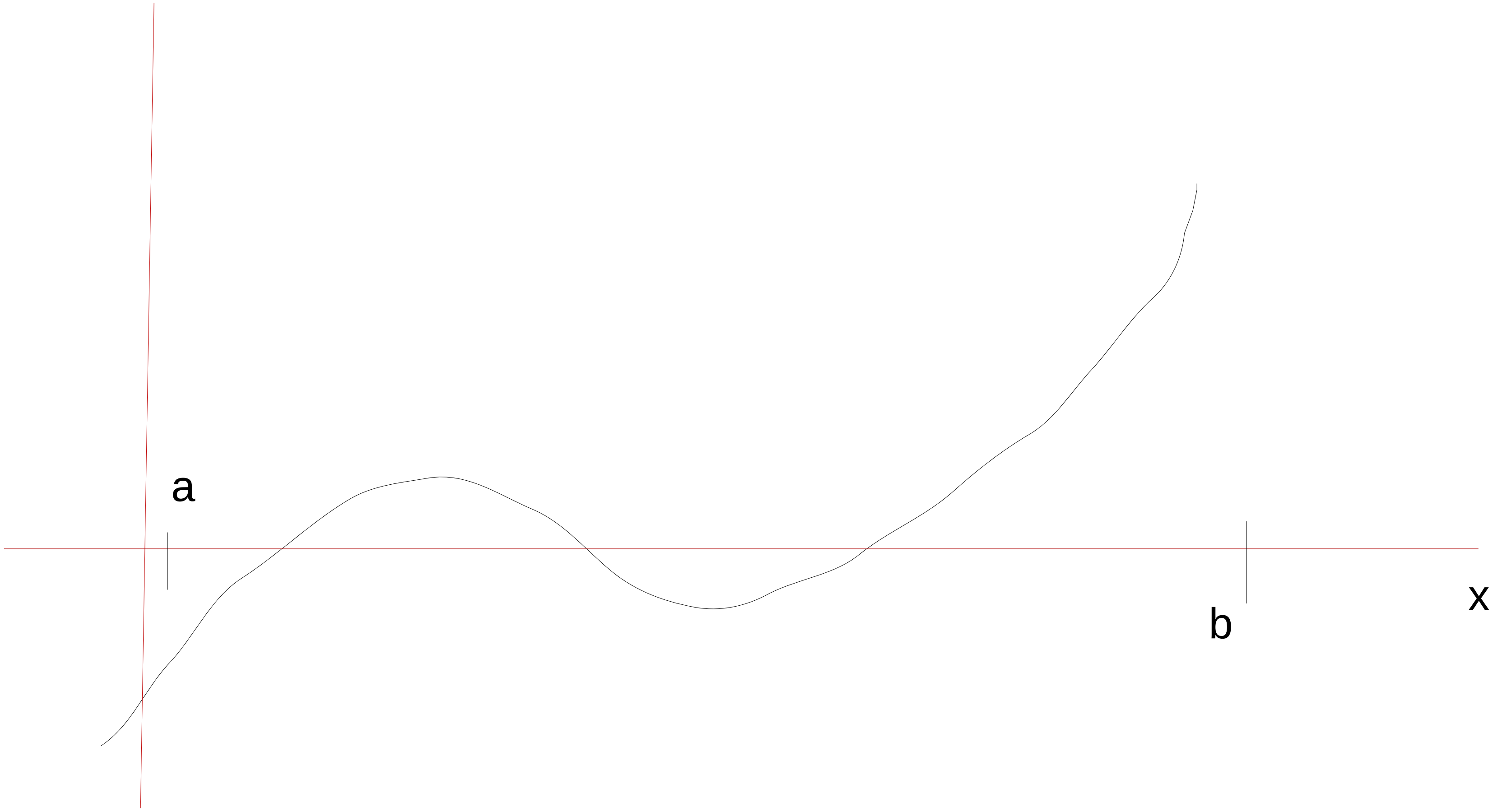
   Representa gràficament la funció $P(v) = \cosh(v)\mathrm{Po}(v)$ i la seva derivada a l'interval considerat, **P4-2016-fig1.png**.

   b) Mitjançant la subrutina de bisecció troba les tres arrels d'$P(v)$ (amb $v \in [0, 4]$), fent servir la informació visual de la representació gràfica, amb una precisió de **1.d-12**.

   c) A continuació estudia la convergència del mètode de Newton-Raphson per trobar les arrels reals començant des dels 10 punts diferents,
   $v_0 = 0.1, 1.0, 1.5, 2.5, 2.51, 2.52, 2.54, 2.55, 2.7$ i 3.0 amb una precisió **1d-12**. Escriu en un fitxer **P4-2016-res2.dat** el valor v_0 i el nombre d'iteracions necessàries per assolir la precisió. Fes una gràfica que il.lustri la convergència del mètode pels valors $v_0 = 2.54, 2.55, 2.7$, p.ex. mostra con varia el valor aproximat de l'arrel per a cada iteració del mètode, **P4-2016-fig2.png**.

Práctica 4, apartado 2

+ Utilizad para Newton Raphson la derivada calculada analíticamente



Práctica 4

3) Considera la següent fórmula per a calcular la derivada primera d'una funció dins de l'interval $[a, b]$,

$$
\begin{aligned}
\text{si } x \in (a, b) \qquad & f'(x) = \frac{f(x+h) - f(x-h)}{2h} \\
\text{si } x = a \qquad & f'(a) = \frac{f(a+h) - f(a)}{h} \\
\text{si } x = b \qquad & f'(b) = \frac{f(b) - f(b-h)}{h} \, .
\end{aligned}
\tag{0.16}
$$

Construeix una subrutina $\mathbf{derifun(ndat,fu,x,dfu)}$ que rebi dos vectors, un amb els valors de la variable equiespaiats $(x_{k+1} - x_k = h)$, x_k, $\mathbf{x(ndat)}$, i l'altre amb els valors corresponents de la funció $\mathbf{fu}(x_k)$, $\mathbf{fu(ndat)}$, i retorni un vector amb la derivada calculada numèricament $f'(x_k)$, $\mathbf{dfu(ndat)}$.

Práctica 4

4) Per a testejar la subrutina **derifun**.

Genera dues taules amb 10 i 200 punts de la funció $P(v)$ amb $v \in [0, 4]$, calcula numèricament la seva derivada amb la subrutina de l'apartat anterior, escriu en dos fitxers: **P4-2016-res3-n10.dat** i **P4-2016-res3-n200.dat**: v, $P(v)$, $P'_{\text{approx}}(v)$, $P'(v)$. Fes una gràfica **P4-2016-fig3.png** comparant les derivades aproximades amb 20 i 200 punts amb el resultat exacte.