1.4 Ejercicios

101. Resolver (con lápiz y papel) los siguientes sistemas de ecuaciones

a) $\begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & 3 & -1 \\ 0 & -2 & 1 \\ 0 & 0 & 3 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 2 \\ -1 \\ 1 \end{bmatrix}$,

b) $\begin{bmatrix} 2 & 0 & 0 \\ -1 & 1 & 0 \\ 3 & 2 & -1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} -1 \\ 3 \\ 0 \end{bmatrix}$

Resultados: a) $(-3, 3, 1)$, b) $(0.5, -4.5, 3.5)$

102. Obtener (con lápiz y papel) la descomposición LU de $A = \begin{bmatrix} 2 & 3 & -6 \\ 1 & -6 & 8 \\ 3 & -2 & 1 \end{bmatrix}$.

Resolver el sistema $A\boldsymbol{x} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$.

Resultados: $\ell_{33} = 0.4\widehat{6}$, $\boldsymbol{x} = (-1, -3, -2)$

103. El siguiente código corresponde al algoritmo para obtener la factorización LU de una matriz A, almacenándola sobre A. Completarlo rellenando los recuadros.

```
1  subroutine factor(a,n)
2  ! datos: matriz a de tamaño n
3  ! resultado: factorización LU de a, almacenada en a
4  implicit none
5  integer :: n,k,i,j,h
6  real*8 :: a(n,n),s
7  ! avanza por la diagonal
8  do k=1,n
9     ! calcula columna k de L
10    do i=[          ]
11       s=0
12       do h=1,k-1
13          s=s+[                ]
14       end do
15       a(i,k)=a(i,k)-s
16    end do
17    ! calcula fila k de U
18    do [    ]=k+1,n
19       s=0
```

```
20        do h=1,k-1
21           s=s+a(k,h)*a(h,j)
22        end do
23        a(k,j)=
24     end do
25  end do
26  return
27  end subroutine factor
```

104. Escribir en un fichero denominado `factor.f90` una subrutina `factor(A,n)` que factorice en LU la matriz A, real, de dimensión n y almacene los elementos de L y U sobre la matriz A. Utilizar `real*8`. Compilarlo y corregir los errores.

105. Escribir un programa que defina la matriz A del ejemplo 1 de la sección §1.1 y calcule su factorización. Compilarlo, ejecutarlo y comprobar el resultado.

106. El siguiente código corresponde a una subrutina para resolver $A\boldsymbol{x} = \boldsymbol{b}$ una vez que A está factorizada en LU. Los elementos a calcular de L están almacenados en la diagonal principal y subdiagonales de A, mientras que las superdiagonales contienen los elementos a calcular de U.

a) La primera parte corresponde a la resolución del sistema $L\boldsymbol{y} = \boldsymbol{b}$, con $\boldsymbol{y}$ almacenado en $\boldsymbol{b}$. Localizar y corregir los errores.

b) Completar el código con la resolución de $U\boldsymbol{x} = \boldsymbol{y}$, con $\boldsymbol{y}$ almacenado en $\boldsymbol{b}$ y almacenando $\boldsymbol{x}$ en $\boldsymbol{b}$.

```
1  subroutine solve(a,b,n)
2  ! datos: n, vector b, factorización LU  de A almacenada en A
3  ! resultado: vector solución almacenado en b
4  implicit none
5  integer::n,k,h
6  real*8 :: a(n,n),b(n),s
7  !resuelve el sistema Ax=b con la matriz A factorizada en LU
8  !resuelve Ly=b
9  do k=1,n
10    s=0
11    do h=1,k
12       s=s+a(h,k)*b(k)
13    end do
14    b(k)=(b(k)+s)/a(k,k)
15 end do
16 !resuelve Ux=y
```

```
17
18
19
20
21 return
22 end subroutine solve
```

107. Escribir en un fichero denominado `solve.f90` una subrutina `solve(A,b,n)` que resuelva el sistema $A\boldsymbol{x} = \boldsymbol{b}$ de tamaño n. En la llamada a la subrutina la matriz A contiene la factorización LU y $\boldsymbol{b}$ contiene el vector de términos independientes. En la salida $\boldsymbol{b}$ contiene el vector solución. Utilizar `real*8`. Compilarlo y corregir los errores.

108. Ampliar el programa del ejercicio 105 para resolver el sistema del ejemplo 3 de la sección §1.2. Compilarlo, ejecutarlo y comprobar el resultado.

109. Sean A y B dos matrices de tamaño n. A es una matriz dada, que ha sido factorizada en LU previamente mediante la subrutina `factor` (ejercicio 104). Utilizando la subrutina `solve` considerada en el ejercicio 107, escribir una subrutina `m_inversa(A,B,n)` para calcular la matriz inversa de A y almacenarla en B. Compilarla y corregir los errores.

110. Sea A una matriz dada de tamaño n, que ha sido factorizada en LU previamente mediante la subrutina `factor` (ejercicio 104). Escribir una función `det(A,n)` para calcular $\det(A)$. Compilarla y corregir los errores.

111. A partir del programa del ejercicio 105 escribir un programa para calcular la matriz inversa y el determinante de A. Ejecutarlo y comprobar el resultado.

Resultados: $\det(A) = -3$, $A^{-1}(3,:) = (0.\widehat{6}\ \ , 0, -1)$

112. Considérense los 4 sistemas de ecuaciones $A\boldsymbol{x} = \boldsymbol{b}_k$

$$\begin{aligned} 3x_1 + 2x_2 \qquad\quad + \ x_4 &= b_1 \\ 2x_1 + 5x_2 - \ x_3 + \ x_4 &= b_2 \\ 4x_1 - \ x_2 - 8x_3 + 2x_4 &= b_3 \\ 3x_1 + \ x_2 - \ x_3 + 6x_4 &= b_4 \end{aligned}$$

siendo $\boldsymbol{b}_1 = (1, 2, -1, 0)^\top$, $\boldsymbol{b}_2 = (9, 2, 4, 3)^\top$, $\boldsymbol{b}_3 = (3, -2, 0, 9)^\top$ y $\boldsymbol{b}_4 = (3, 2, 1, 6)^\top$.

Resultados: $\det(A) = -490$, $A^{-1}(3,:) = (0.2, 0.11, -0.11, 0.02)$, $\boldsymbol{x}_1(1) = 0.1$, $\boldsymbol{x}_2(2) = -0.\hat{6}$, $\boldsymbol{x}_3(3) = 1.02$, $\boldsymbol{x}_4(4) = 0.69$

Escribir un programa que realice las siguientes operaciones sobre la matriz A: calcular la factorización LU de la matriz A, calcular su inversa, calcular su determinante y resolver los 4 sistemas de ecuaciones. Ejecutarlo y comprobar el resultado.

113. Repetir el ejercicio anterior para los sistemas

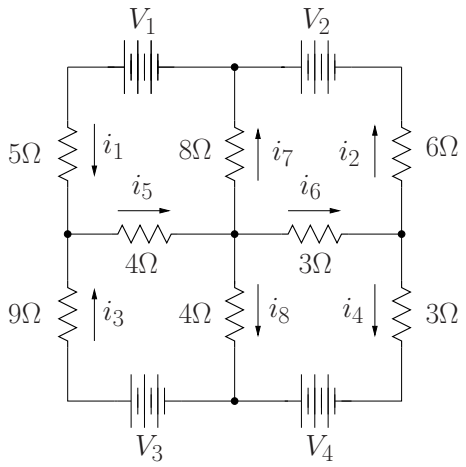
$$\begin{aligned} 4x_1 + 0.5x_2 - 0.3x_3 + x_4 - 0.2x_5 &= b_1 \\ 4x_2 - x_3 + x_5 &= b_2 \\ -x_1 + 3x_3 - x_4 + &= b_3 \\ 0.7x_1 + 0.2x_2 - 3x_4 + x_5 &= b_4 \\ 0.9x_2 - 0.2x_3 + 2x_5 &= b_5 \end{aligned}$$

siendo $\boldsymbol{b}_1 = (-0.4, 0.1, 0.2, -1.3, 0)^\top$, $\boldsymbol{b}_2 = (1.4, 0.7, -0.8, 1.1, 0.3)^\top$, $\boldsymbol{b}_3 = (0.4, -0.8, -0.2, 0, 1.5)^\top$, $\boldsymbol{b}_4 = (3, 2.5, -2.5, 0, 1.8)^\top$, $\boldsymbol{b}_5 = (-1.2, 0, -0.8, 1.2, 0.9)^\top$. Ejecutarlo y comprobar el resultado.

Resultados: $\det(A) = -265.25$, $A^{-1}(3,3) = 0.34$, $\boldsymbol{x}_2(5) = 0.08$

114. Las leyes de Kirchhoff para analizar un circuito eléctrico establecen que: i) a lo largo de cualquier recorrido cerrado la suma algebraica de las fuerzas electromotrices es igual a la caída neta de potencial, y ii) la suma algebraica de intensidades en cualquier nudo es nula. La aplicación de ambas leyes en el circuito de la figura conduce a las siguientes ecuaciones para determinar las intensidades $i_1, \ldots, i_8$ que se obtienen con los potenciales $V_1, \ldots, V_4$.

$$\begin{aligned} 5\,i_1 + 4\,i_5 + 8\,i_7 &= V_1 \\ 6\,i_2 + 3\,i_6 - 8\,i_7 &= V_2 \\ 9\,i_3 + 4\,i_5 + 4\,i_8 &= V_3 \\ 3\,i_4 + 3\,i_6 - 4\,i_8 &= V_4 \\ i_1 + i_3 - i_5 &= 0 \\ -i_2 - i_4 + i_6 &= 0 \\ -i_1 + i_2 + i_7 &= 0 \\ -i_3 + i_4 + i_8 &= 0 \end{aligned}$$



a) Escribir el sistema anterior en forma matricial $A\boldsymbol{x} = \boldsymbol{b}$.

b) Escribir un programa para resolver el sistema en los casos:

1.– $V_1 = 1,\ V_2 = V_3 = V_4 = 0$

2.– $V_2 = 1,\ V_1 = V_3 = V_4 = 0$

3.– $V_3 = 1,\ V_1 = V_2 = V_4 = 0$

4.– $V_4 = 1,\ V_1 = V_2 = V_3 = 0$

5.– $V_1 = V_2 = V_3 = V_4 = 1$

Para cada caso utilizar un vector `x` diferente, por ejemplo `x_1`, `x_2`, `x_3`, `x_4` y `x_5`

c) Comprobar que la solución del caso 5 puede obtenerse como suma de las soluciones de los casos 1, 2, 3 y 4, es decir `x_5=x_1+x_2+x_3+x_4`. ¿Qué propiedad matemática justifica este resultado?

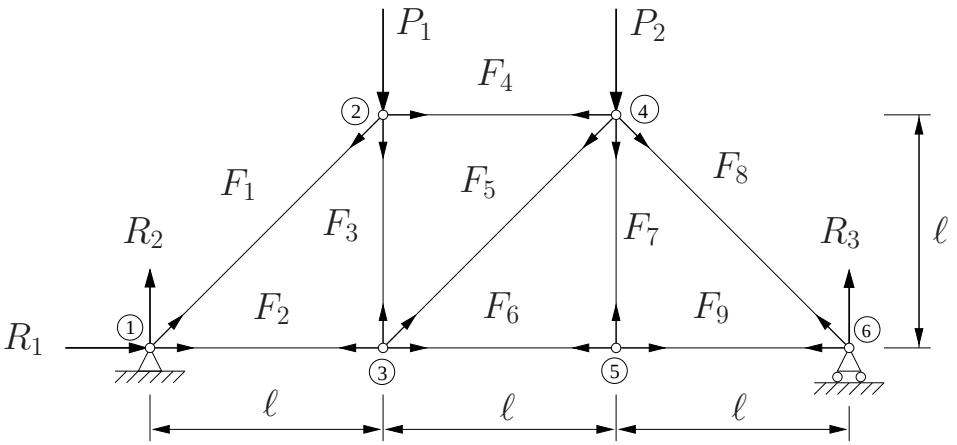
Resultados: 1) $i_1 = 0.08587$, 2) $i_2 = 0.08563$, 3) $i_3 = 0.07305$, 4) $i_4 = 0.12378$, 5) $i_5 = 0.14471$

115. Una cercha es un elemento estructural que se utiliza para soportar cargas en puentes y edificios con grandes distancias entre los apoyos (almacenes, naves industriales, polideportivos, etc.). Está formada por barras rectas unidas en los nudos formando triángulos. Se supone que los nudos

se comportan como articulaciones, de manera que las barras sólo admiten cargas axiales (tracción o compresión) y que las cargas externas se aplican sobre uno o varios nudos. La figura muestra una cercha típica.

Las ecuaciones que determinan los valores de las cargas en las barras $(F_1, \ldots, F_8)$ y las reacciones en los apoyos (R_1, R_2, R_3) bajo la acción de las cargas exteriores (P_1, P_2) se obtienen mediante las condiciones de equilibrio de cada nudo: $\Sigma F_x = 0$ y $\Sigma F_y = 0$, dando lugar a las siguientes ecuaciones ($F_i > 0$ corresponde a una carga de tracción):

Nudo	Componente	Ecuación
1	y	$R_1 + (1/\sqrt{2})F_1 = 0$
1	x	$R_2 + (1/\sqrt{2})F_1 + F_2 = 0$
2	x	$-(1/\sqrt{2})F_1 + F_4 = 0$
3	x	$-F_2 + (1/\sqrt{2})F_5 + F_6 = 0$
2	y	$-(1/\sqrt{2})F_1 - F_3 = P_1$
4	x	$-F_4 - (1/\sqrt{2})F_5 + (1/\sqrt{2})F_8 = 0$
3	y	$F_3 + (1/\sqrt{2})F_5 = 0$
5	x	$F_6 - F_9 = 0$
5	y	$F_7 = 0$
4	y	$-(1/\sqrt{2})F_5 - F_7 - (1/\sqrt{2})F_8 = P_2$
6	x	$(1/\sqrt{2})F_8 + F_9 = 0$
6	y	$(1/\sqrt{2})F_8 + R_3 = 0$



a) Escribir el sistema anterior en forma matricial $A\boldsymbol{x} = \boldsymbol{b}$, siendo $\boldsymbol{x} = (R_1, R_2, F_1, \ldots, F_9, R_3)$ (se recomienda no alterar el orden de las incógnitas ni de las ecuaciones).

b) Escribir un programa para resolver el sistema en los casos:

1.– $P_1 = 1\times10^4,\ P_2 = 0$

2.– $P_1 = 0,\ P_2 = 1\times10^4$

3.– $P_1 = P_2 = 1\times10^4$

Para cada caso utilizar un vector `x` diferente, por ejemplo `x_1`, `x_2` y `x_3`. La matriz A en este problema está "casi" vacía. Una buena estrategia para asignar los valores es hacerlos todos iguales a 0 y después asignar, de forma individual, los valores a los elementos no nulos.

c) Comprobar que, de nuevo, la solución del caso 3 puede obtenerse como suma de las soluciones

de los casos 1 y 2 , es decir `x_3=x_1+x_2`. ¿Cuál es, sin resolver el correspondiente sistema, la solución del caso $P_1 = 1.5\times10^4,\ P_2 = 3.5\times10^4$?

Resultados: b.1) $F_1 = -0.943 \times 10^4$, b.2) $F_2 = 0.333 \times 10^4$, b.3) $F_3 = 0$, c) $F_4 = -0.217 \times 10^5$

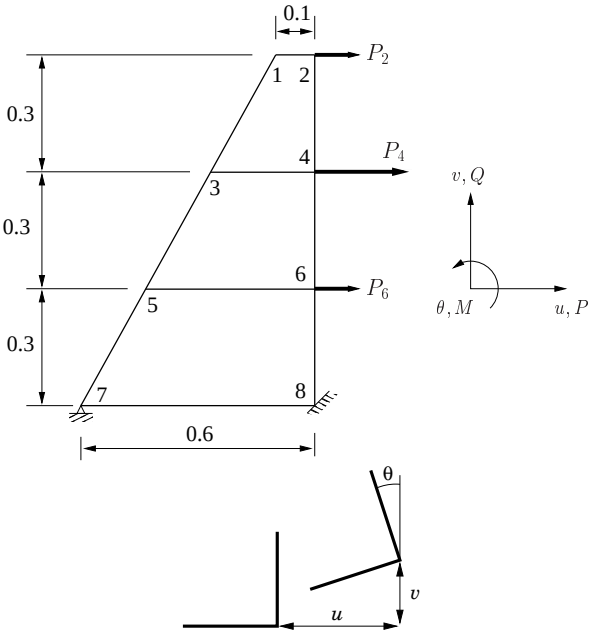
116. La figura representa la estructura del estabilizador vertical de la avioneta RANS S-7 Courier y las cargas generadas en el plano del estabilizador por una deflexión del timón de dirección. Se desea determinar la deformación de la estructura, caracterizada por los desplazamientos horizontal y vertical y el giro de cada nudo, u, v, θ, bajo la acción de las cargas P_2, P_4 y P_6.
La relación entre cargas y desplazamientos está dada por

$$K\delta = c$$

siendo $\delta = (u_1, v_1, \theta_1, \ldots, u_6, v_6, \theta_6, \theta_7)^\top$ el vector de desplazamientos ($u_7 = v_7 = u_8 = v_8 = \theta_8 = 0$), $c = (P_1, Q_1, M_1, \ldots, P_6, Q_6, M_6, M_7)^\top$ el vector de cargas y K la matriz de rigidez que puede escribirse como

$$K = EA \cdot K_{EA} + EI \cdot K_{EI}$$

siendo E el módulo de elasticidad del material, A e I el área y el momento de inercia de la sección, y K_{EA} y K_{EI} las matrices adjuntas, definidas en las subrutinas `matriz_kei.f90` y `matriz_kea.f90`, cuyos elementos dependen exclusivamente de las dimensiones de las barras.
Determinar δ resolviendo el sistema $K\delta = c$ mediante el método LU en los siguientes supuestos:



Desplazamiento de un nudo

Material: alumnio, $E = 7 \cdot 10^{10}\,\mathrm{N/m^2}$.
Sección de cada barra:

a) Circular, radio $0.0075\,\mathrm{m}$($I = \pi R^4/4$).
b) Anular, radio exterior $0.0075\,\mathrm{m}$, radio interior $0.006\,\mathrm{m}$ ($I = \pi(R^4 - r^4)/4$).

Cargas:

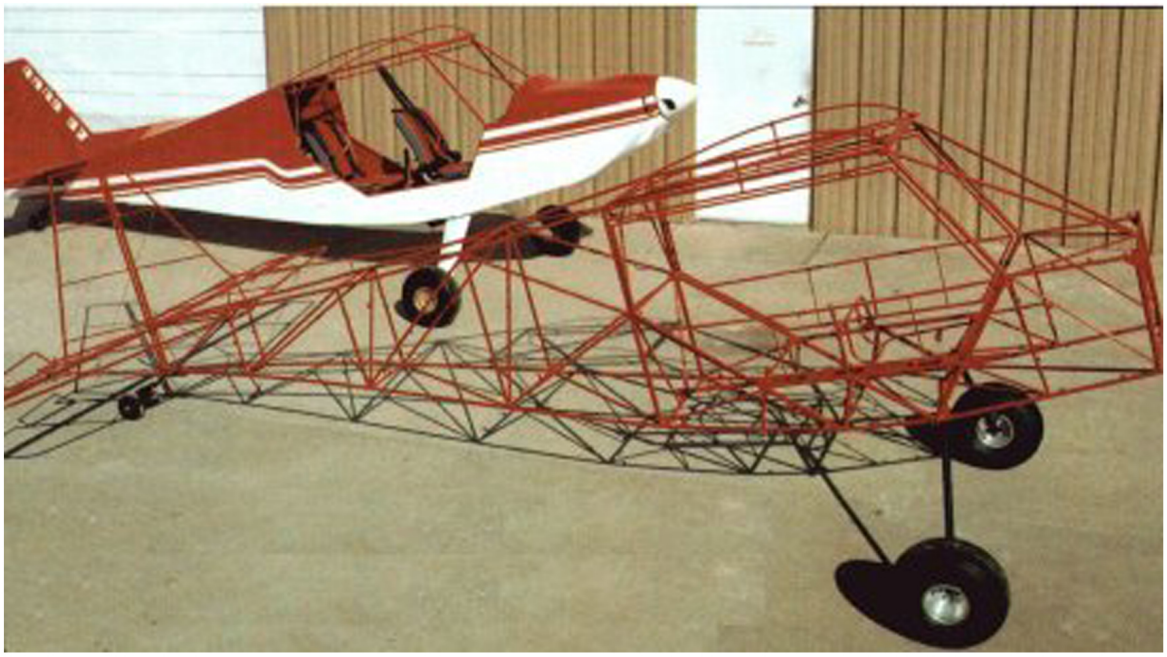
1) $P_2 = 0.3 \times 10^3\,\mathrm{N}$, $P_4 = 0.7 \times 10^3\,\mathrm{N}$,$P_6 = 0.5 \times 10^3\,\mathrm{N}$ y las restantes componentes de c nulas.
2) $P_4 = 1.5 \times 10^3\,\mathrm{N}$ y las restantes componentes de c nulas.

Presentar los siguientes resultados para cada sección:
- valores de la diagonal de la matriz $L : l_{ii}$, $i = 1, \ldots, 19$,
- tabla con las deformaciones de los nudos (una para cada caso de carga),

Caso de carga:

Nudo	u	v	θ
⋮	⋮	⋮	⋮

Resultados: a) $\ell_{19,19} = 0.14167 \times 10^4$, a.1) $u_6 = 0.18501 \times 10^{-1}$, a.2) $v_6 = -0.83197 \times 10^{-4}$,
b) $\ell_{77} = 0.54905 \times 10^8$, b.1) $\theta_1 = 0.38183 \times 10^{-1}$, b.2) $\theta_2 = 0.48055 \times 10^{-1}$



$$K_{EA} = \begin{bmatrix}
10.7 & 1.2 & 0.0 & -10.0 & 0.0 & 0.0 & -0.7 & -1.2 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
1.2 & 2.2 & 0.0 & 0.0 & 0.0 & 0.0 & -1.2 & -2.2 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-10.0 & 0.0 & 0.0 & 10.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 3.3 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -3.3 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.7 & -1.2 & 0.0 & 0.0 & 0.0 & 0.0 & 5.1 & 2.5 & 0.0 & -3.7 & 0.0 & 0.0 & -0.7 & -1.2 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-1.2 & -2.2 & 0.0 & 0.0 & 0.0 & 0.0 & 2.5 & 4.5 & 0.0 & 0.0 & 0.0 & 0.0 & -1.2 & -2.2 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -3.7 & 0.0 & 0.0 & 3.7 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & -3.3 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 6.7 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -3.3 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.7 & -1.2 & 0.0 & 0.0 & 0.0 & 0.0 & 3.7 & 2.5 & 0.0 & -2.3 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -1.2 & -2.2 & 0.0 & 0.0 & 0.0 & 0.0 & 2.5 & 4.5 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -2.3 & 0.0 & 0.0 & 2.3 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -3.3 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 6.7 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0
\end{bmatrix}$$

$$K_{EI} = \begin{bmatrix}
227.2 & -126.2 & 44.6 & 0.0 & 0.0 & 0.0 & -227.2 & 126.2 & 44.6 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-126.2 & 12070.0 & 575.2 & 0.0 & -12000.0 & 600.0 & 126.2 & -70.1 & -24.8 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
44.6 & 575.2 & 51.7 & 0.0 & -600.0 & 20.0 & -44.6 & 24.8 & 5.8 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 444.4 & 0.0 & 66.7 & 0.0 & 0.0 & 0.0 & -444.4 & 0.0 & 66.7 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & -12000.0 & -600.0 & 0.0 & 12000.0 & -600.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 600.0 & 20.0 & 66.7 & -600.0 & 53.3 & 0.0 & 0.0 & 0.0 & -66.7 & 0.0 & 6.7 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-227.2 & 126.2 & -44.6 & 0.0 & 0.0 & 0.0 & 454.5 & -252.5 & 0.0 & 0.0 & 0.0 & 0.0 & -227.2 & 126.2 & 44.6 & 0.0 & 0.0 & 0.0 & 0.0 \\
126.2 & -70.1 & 24.8 & 0.0 & 0.0 & 0.0 & -252.5 & 770.7 & 84.2 & 0.0 & -630.5 & 84.2 & 126.2 & -70.1 & -24.8 & 0.0 & 0.0 & 0.0 & 0.0 \\
44.6 & -24.8 & 5.8 & 0.0 & 0.0 & 0.0 & 0.0 & 84.2 & 38.3 & 0.0 & -84.2 & 7.5 & -44.6 & 24.8 & 5.8 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & -444.4 & 0.0 & -66.7 & 0.0 & 0.0 & 0.0 & 888.9 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -444.4 & 0.0 & 66.7 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -630.5 & -84.2 & 0.0 & 630.5 & -84.2 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 66.7 & 0.0 & 6.7 & 0.0 & 84.2 & 7.5 & 0.0 & -84.2 & 41.6 & 0.0 & 0.0 & 0.0 & -66.7 & 0.0 & 6.7 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -227.2 & 126.2 & -44.6 & 0.0 & 0.0 & 0.0 & 454.5 & -252.5 & 0.0 & 0.0 & 0.0 & 0.0 & 44.6 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 126.2 & -70.1 & 24.8 & 0.0 & 0.0 & 0.0 & -252.5 & 288.1 & 32.0 & 0.0 & -147.8 & 32.0 & -24.8 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 44.6 & -24.8 & 5.8 & 0.0 & 0.0 & 0.0 & 0.0 & 32.0 & 32.6 & 0.0 & -32.0 & 4.6 & 5.8 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -444.4 & 0.0 & -66.7 & 0.0 & 0.0 & 0.0 & 888.9 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -147.8 & -32.0 & 0.0 & 147.8 & -32.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 66.7 & 0.0 & 6.7 & 0.0 & 32.0 & 4.6 & 0.0 & -32.0 & 35.9 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 44.6 & -24.8 & 5.8 & 0.0 & 0.0 & 0.0 & 18.3
\end{bmatrix}$$

117. La resolución de grandes sistemas de ecuaciones requiere la realización de gran número de operaciones (del orden de $n^3/3$), con sus correspondientes errores de redondeo. Éstos pueden acumularse y afectar a la exactitud de la solución.

Aunque en la mayoria de los casos prácticos el algoritmo *LU* proporciona soluciones cuya exactitud es acorde con la precisión utilizada (`REAL*4` ó `REAL*8`), existen algunos tipos de matrices (denominadas genéricamente "mal condicionadas") donde la acumulación de los errores produce resultados erróneos.

Considérese el sistema $A\boldsymbol{x} = \boldsymbol{b}$, con A y $\boldsymbol{b}$ definidos por

$$a_{ij} = \begin{cases} \dfrac{1}{4^{j-i}} & j > i \\ \dfrac{1}{i} & j = i \\ \dfrac{1}{5^{i-j}} & j < i \end{cases} \qquad b_i = \begin{cases} \dfrac{4}{3}\left(1 - \dfrac{1}{4^n}\right) & i = 1 \\ \dfrac{1}{i} + \dfrac{1}{3}\left(1 - \dfrac{1}{4^{n-i}}\right) + \dfrac{1}{4}\left(1 - \dfrac{1}{5^{i-1}}\right) & 1 < i < n \\ \dfrac{1}{n} + \dfrac{1}{4}\left(1 - \dfrac{1}{5^{n-1}}\right) & i = n \end{cases}$$

a) Mediante las subrutinas `factor` y `m_inversa` y la función `det`, calcular para $n = 6$ la matriz inversa A^{-1} y el determinante $\det(A)$.

b) Mediante las subrutinas `factor`, `solve` y la función `norma_euclidea` (ver ejercicio 910, primera parte de la asignatura) para $n = 10, 20, 30, \ldots, 250$ resolver mediante el método *LU* el sistema $A\boldsymbol{x} = \boldsymbol{b}$ cuya solución exacta es $\boldsymbol{\alpha} = (1, 1, \ldots, 1)^\top$ y calcular $\|\boldsymbol{x} - \boldsymbol{\alpha}\|_2$. La salida del programa debe ser un fichero con los valores de n y $\|\boldsymbol{x} - \boldsymbol{\alpha}\|_2$.

Sugerencias:

1.– Escribir un programa para cada apartado.

2.– Construir 2 subrutinas `matriz_a(a,n)` y `vector_b(b,n)` para calcular A y $\boldsymbol{b}$.

3.– En el programa del apartado b) dimensionar A y $\boldsymbol{b}$ con $nmax = 250$. Utilizar un bucle `do n=10,250,100` y para pasar la submatriz de tamaño n a cada subrutina o función utilizar, por ejemplo, `factor(a(1:n,1:n),n)`.

Los resultados deberán mostrar que para $n = 250$ la diferencia entre la solución numérica y la exacta es grande. Téngase en cuenta que este caso es un poco "patológico"y que en la mayoría de las situaciones el método *LU* puede emplearse con seguridad

Resultados: a) $\det(A) = 0.373 \times 10^{-3}$, b) $n = 200$, $\|\boldsymbol{x} - \boldsymbol{\alpha}\|_2 = 10^{-3}$; $n = 250$, $\|\boldsymbol{x} - \boldsymbol{\alpha}\|_2 = 10$

2.4 Ejercicios

201. Utilizar el método de Newton (con lápiz y papel) para determinar aproximaciones a las raíces de las siguientes ecuaciones tales que $|f(x_n)| < 10^{-4}$.

a) $x^3 - 2x - 5 = 0$ en $[1, 4]$ (esta ecuación fue utilizada por Newton en 1669 para ilustrar el método que lleva su nombre). Tomar $x_0 = 2$.

b) $x - \cos x = 0$ en $[0, \pi/2]$ con $x_0 = 1$.

c) $x^2 - 0.8 - 0.2 \operatorname{sen} x = 0$ en $[0, \pi/2]$ con $x_0 = 1$.

Resultados: a) $n = 3$, $x = 2.09455$, b) $n = 2$, $x = 0.73911$, c) $n = 2$, $x = 0.98308$

202. Repetir el ejercicio anterior usando el método de la Secante.

Resultados: a) $n = 5$, b) $n = 5$, c) $n = 7$

203. Encontrar y corregir los errores del siguiente código para calcular mediante el método de Newton la raíz de $x \operatorname{sen} x + \cos x = 0$ con el criterio de parada $|x_n - x_{n-1}|/|x_n| < \varepsilon$ y $|f(x_n)| < \varepsilon$. Compilarlo, ejecutarlo y comprobar la solución.

```
1  program newton
2  ! datos: x(aproximacion inicial), eps,nmax
3  ! resultado: x (raiz)
4  implicit none
5  integer :: nmax=10,i
6  real*8 :: x=1,eps=1d-6,errorx,errory,dx
7  do i=1,nmax
8      dx=tan(x)-1d0/x
9      errorx=dx/x
10     if (errorx<eps) then
11         print*, 'la raiz es:', x
12         exit
13     end if
14     x=x-dx
15 end do
16 if (n.gt.nmax) print*, 'número máximo de iteraciones'
17 end program newton
```

204. Una vez resuelto el ejercicio anterior, escribir los códigos de dos funciones `f(x)` y `df(x)` para calcular, dado x, el valor de $f(x) = x \operatorname{sen} x + \cos x$ y de su derivada. Utilizar `real*8`.

205. Completar y corregir el siguiente código correspondiente a la resolución del ejercicio 203 pero utilizando las funciones `f(x)` y `df(x)` del ejercicio 204. Escribir los tres códigos en un mismo fichero, compilarlo y ejecutarlo.

```
1  program newton
```

```
! datos: x(aproximacion inicial), eps, nmax
! resultado: x (raiz)
implicit none
integer :: nmax=10,i
real*8 :: x=1,eps=1d-6, errorx, errory,dx,f,df
do i=1,nmax
    dx=f(x)/
    x=x+dx
    errorx=abs(dx/x)
    errory=
    ! escribe resultados intermedios
    print '(i2,3(1x,d12.5))',i,x, errorx, errory
    if ((errorx<eps).and.                ) then
        print*, 'a raiz es':, x
        exit
    end if
end do
if (n.gt.nmax) print*,'numero máximo de iteraciones'
end program newton
```

206. Tanto en el método de Newton como en el de la Secante el error relativo $|\Delta x/x_n|$ se calcula mediante `errorx=abs(dx/x)`. ¿Qué ocurriría si la solución del problema fuese $x = 0$, o próximo a 0?
¿Cómo debe modificarse la expresión $|\Delta x/x_n|$ para evitar dividir por 0, o por un número muy pequeño?
Aunque esta modificación no aparecerá de forma explícita en el resto de estos apuntes, debe tenerse en cuenta en la programación de los algoritmos.

Resultados: $x = 56.53098$

207. En el código del ejercicio 205 el valor de la función `f(x)` se calcula 2 veces para un mismo valor de `x`. Esto puede evitarse definiendo una variable `y` que guarde el valor de `f(x)`. Hacer esta modificación en el código del ejercicio 205. Compilarlo y ejecutarlo.

208. Completar y corregir el siguiente código correspondiente al método de la Secante para resolver la ecuación $x \operatorname{sen} x + \cos x = 0$ con el criterio de parada $|x_n - x_{n-1}|/|x_n| < \varepsilon$ y $|f(x_n)| < \varepsilon$. Escribir en un mismo fichero el código y el de la función `f(x)`. Compilarlo, ejecutarlo y comprobar la solución.

```
program secante_0
! datos: x0, x1(aproximaciones iniciales), f(x), eps, nmax
! resultado: x2 (raiz)
```

```
implicit none
integer :: nmax=10,i
real*8 :: x0=0d0,x1=1d0,eps=1d-6
real*8 :: y0,y1,y2,errorx,errory,p,dx,f
! calcula los valores de f(x) en las aproximaciones iniciales
y0=f(x0)
y1=
! bucle de iteraciones
do i=1,nmax
    ! calcula la pendiente y la correccion de x
    p=
    dx=y1/p
    ! calcula la nueva x y la nueva funcion
    x2=x1-
    y2=f(x2)
    ! calcula los errores y efectua el test de parada
    errorx=
    =abs(y2)
     ! escribe resultados intermedios
     print (i2,3(1x,d12.5)), i, x2, errorx, errory
    if ((errorx<eps) (errory<eps)) then
        print*, 'la raiz es:', x2
        exit
    endif
    !actualiza los valores de x e y
    x1=x2
    y0=
    =y2
enddo
if (n.gt.nmax) print*,'numero máximo de iteraciones'
end program secante_0
```

Resultados: $x = -2.798386$, ¿por qué no coincide con el resultado de los ejercicios 205 y 207.

209. Modificar el programa del ejercicio 205, ó 207, para resolver, mediante el método de Newton, las ecuaciones del ejercicio 201. Elegir razonadamente la aproximación inicial.

210. Modificar el programa del ejercicio 208 para resolver las ecuaciones del ejercicio 201 mediante el

método de la Secante. Elegir razonadamente las aproximaciones iniciales.

211. Calcular una aproximación de $4^{3/4}$ mediante el método de la Secante.

Resultados: 2.828427

212. Determinar aproximadamente los puntos de intersección de las gráficas de $y = e^x$ e $y = 3x$.

Resultados: $(0.61906, 1.85718)$, $(1.51213, 4.53640)$

213. Tomando como punto de partida el programa del ejercicio 205, ó 207, escribir el código de una subrutina `newton(x,eps,nmax)` para resolver la ecuación $f(x) = 0$ mediante el método de Newton. En la llamada a la subrutina `x` contiene la aproximación inicial, `eps` la tolerancia de error y `nmax` el número máximo de iteraciones. En la salida `x` debe contener la solución. La subrutina debe "linkarse" con un programa principal y los códigos de las funciones `f(x)` y `df(x)`.

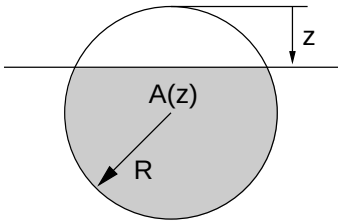
214. A partir del programa del ejercicio 208 escribir una subrutina `secante_0(x0,x1,x2,eps,nmax)` para resolver la ecuación $f(x) = 0$ mediante el método de la Secante. En la llamada a la subrutina `x0` y `x1` contienen la aproximaciones iniciales, `eps` la tolerancia de error y `nmax` el número máximo de iteraciones. En la salida `x2` debe contener la solución. La subrutina debe "linkarse" con un programa principal y el código de la función `f(x)`.

215. Una esfera de plástico de densidad $\rho_p = 700\,\mathrm{Kg/m^3}$ y radio $R = 0.08\,\mathrm{m}$ se sumerge en agua ($\rho_a = 1000\,\mathrm{Kg/m^3}$). Sea h la altura de de la parte sumergida. El volumen sumergido es $V(h) = \frac{1}{3}\pi(3Rh^2 - h^3)$. Del principio de Arquímedes se deduce la siguiente ecuación para determinar h: $\rho_a V(h) = \frac{4}{3}\rho_p \pi R^3$. Introduciendo $x = h/R$ la ecuación anterior puede escribirse como $f(x) \equiv 3x^2 - x^3 - 2.8 = 0$. Resolverla mediante el método de Newton. Tomar como cota de error $\varepsilon = 10^{-6}$. Por consideraciones geométricas determinar en qué intervalo de valores tiene que estar la solución y tomar el punto medio como aproximación inicial.

Resultados: $h = 1.273485\,R$

216. Un cilindro de plástico ($\rho_p = 700\,\mathrm{Kg/m^3}$) de longitud $L = 0.1\,\mathrm{m}$ y radio $R = 0.03\,\mathrm{m}$ se sumerge en agua ($\rho_a = 1000\,\mathrm{Kg/m^3}$). Sea z la altura de la parte no sumergida medida desde el borde superior del cilindro. El principio de Arquímedes proporciona la siguiente ecuación para determinar z:

$$\rho_a L A(z) = \rho_p \pi R^2 L$$



donde

$$A(z) = R^2\left[\frac{\pi}{2} + \operatorname{arc\,sen}\frac{R-z}{R} + \frac{R-z}{R}\sqrt{\frac{z}{R}\left(2 - \frac{z}{R}\right)}\right].$$

Introduciendo $x = z/R$ la ecuación anterior puede escribirse como

$$f(x) \equiv \operatorname{arc\,sen}(1-x) + (1-x)\sqrt{x(2-x)} - 0.2\pi = 0.$$

Calcular su solución mediante el método de Newton. Tomar como cota de error $\varepsilon = 10^{-6}$. Por consideraciones geométricas determinar en qué intervalo de valores tiene que estar la solución. Usar el punto medio como aproximación inicial.

Resultados: $z = 0.680308\,R$

217. Resolver las ecuaciones de los ejercicios 215 y 216 mediante el método de la Secante.

218. El techo en vuelo a punto fijo de un helicóptero puede estimarse conociendo la densidad del aire para la cual la potencia suministrada al rotor (P_M) es igual a la potencia de vuelo, necesaria para vencer la resistencia inducida (P_i) y la parásita (P_1, P_2, P_3). La dependencia de estas potencias con la densidad conduce a la siguiente expresión para determinar el valor de la densidad, ρ, a la que tiene lugar el equilibrio anterior:

$$P_M \left(\frac{\rho}{\rho_0}\right)^{0.8} = P_i\sqrt{\frac{\rho_0}{\rho}} + P_1\left(\frac{\rho}{\rho_0}\right) + P_2\sqrt{\frac{\rho}{\rho_0}} + P_3\left(\frac{\rho_0}{\rho}\right),$$

donde ρ_0 es la densidad al nivel del mar, $P_M = 1.5\,\text{MW}$, $P_i = 0.9\,\text{MW}$, $P_1 = 0.26\,\text{MW}$, $P_2 = 0.04\,\text{MW}$ y $P_3 = 0.035\,\text{MW}$.

Definiendo $x = \rho/\rho_0$ esta expresión puede escribirse como

$$f(x) \equiv 1.5x^{0.8} - 0.9\frac{1}{\sqrt{x}} - 0.26x - 0.04\sqrt{x} - 0.035\frac{1}{x} = 0.$$

Calcular la solución de la ecuación anterior mediante el método de Newton y el método de la Secante. Utilizar como cota de error en cada método $\varepsilon = 10^{-8}$. Teniendo en cuenta que $x \in]0, 1]$, tomar $x_0 = 0.5$ (Newton) y $x_0 = 0.1$ y $x_1 = 0.9$ (Secante).

Nota: La densidad del aire en función de la altura (en metros) viene dada por la siguiente expresión $H = 4.433\times10^4\left(1 - (\rho/\rho_0)^{0.23496}\right)$.

Resultados: $H = 1973.4\,\text{m}$

219. En el código del método de la Secante del ejercicio 208 se calcula la pendiente de la recta secante mediante $p = (y_1 - y_0)/(x_1 - x_0)$. Cuando se hayan realizado algunas iteraciones x_1 y x_0 serán valores próximos a la raíz y al restarlos es posible que se pierdan cifras significativas . Para evitar esta situación conviene guardar la diferencia entre dos aproximaciones sucesivas x_0 y x_1 en una variable dx de forma que no sea necesario calcular explícitamente su diferencia. A continuación se muestra un posible código para resolver la ecuación $f(x) = 0$ que incluye esta modificación. Completarlo y corregirlo. Escribir el código de la función `f(x)`, siendo $f(x) = 2xe^x - 1$, adjuntarlo al código anterior, comprobar si los valores de `x0` y `x1` son adecuados, compilarlo y ejecutarlo.

```
1  program secante_1
2  ! datos: x0, x1(aproximaciones iniciales),f(x),eps,nmax
3  ! resultado: x2 (raiz)
4  implicit none
5  integer :: nmax=10,i
6  real*8 :: x0=0d0,x1=1d0,eps=1d-6
7  real*8 ::
8  ! calcula los valores de f(x) en las aproximaciones iniciales
9  y0=
10 y1=
11 ! calcula la diferencia de las aproximaciones iniciales
12 dx=x1+x0
```

```
13 ! bucle de iteraciones
14 do i=1,nmax
15      ! calcula la correccion de x
16      dx=          /(y1-y0)
17      ! calcula la nueva x y la nueva funcion
18      x2=    -dx
19          =f(x2)
20      ! calcula los errores y efectua el test de parada
21          =abs(dx/x2)
22          
23      if ((errorx<eps).or.(errory<eps)) then
24           print*, 'la raiz es:', x2
25           exit
26      endif
27      !actualiza los valores de x e y
28      
29      
30      
31      
32 enddo
33 if (n.gt.nmax) print*,'numero máximo de iteraciones'
34 end program secante_1
```

Resultados: $x = 0.35173$

220. A partir de los ejercicios 214 y 219 escribir una subrutina `secante_1(x0,x1,x2,eps,nmax)` para resolver la ecuación $f(x) = 0$ mediante el método de la Secante. En la llamada a la subrutina `x0` y `x1` contienen la aproximaciones iniciales, `eps` la tolerancia de error y `nmax` el número máximo de iteraciones. En la salida `x2` debe contener la solución. La subrutina debe "linkarse" con un programa principal y el código de la función `f(x)`.

220.bis Modificando las subrutinas de los ejercicios 213 y 214 escribir dos subrutinas `newton_f(x,eps,nmax,f,df)` y `secante_f(x0,x1,x2,eps,nmax,f)`, para resolver la ecuación $f(x) = 0$ mediante los métodos de Newton y de la Secante, en las que se incluya el nombre de los subprogramas función para calcular $f(x)$ y $f'(x)$ en la lista de argumentos formales (esto requiere obligatoriamente utilizar el atributo `external`, (ver sección §9.3 de los apuntes de la primera parte de la asignatura).

221. Sea la función $G(x, y) = y + x^3 - x^2 e^{-y}$. La ecuación $G(x, y) = 0$ define implícitamente a $y = f(x)$ para $x \in [0, 2]$ (comprobarlo mediante el Teorema de la Función Implícita).

a) Determinar $y_0 = f(0)$.

b) La ecuación que determina $v \equiv y_1 = f(0.1)$ es $G(0.1, v) \equiv v + 0.001 - 0.01e^{-v} = 0$. Escribir un programa para resolverla mediante el método de Newton con $\varepsilon = 10^{-6}$ tomando como aproximación inicial $v = y_0 = f(0)$. Compilarlo y ejecutarlo.

c) Construir un algoritmo para calcular los valores $y_i = f(x_i)$ para $x_i = i/10$ $(i = 1, 2, \ldots, 20)$ utilizando el método de Newton.

d) El siguiente código corresponde al algoritmo del apartado anterior. En cada punto x_i se utilizan las variables auxiliares $t \equiv x_i$ y $v \equiv y_i$, de manera que el valor y_i se obtiene resolviendo la ecuación $G(t, v) = 0$ mediante el método de Newton tomando como aproximación inicial de la incógnita $v = y_{i-1}$. Una vez resuelta se toma $y_i = v$. Completar y corregir el código. Las funciones `g(x,y)`y `gy(x,y)` calculan los valores de $G(x, y)$ y $\partial G(x, y)/\partial y$.

```
1  program funcion_implicita
2  ! datos: g(x,y), gy(x,y), eps, nmax
3  ! resultado: x(0:n),y(0:n)
4  implicit none
5  integer, parameter :: n=20
6  integer :: nmax=10, i, j
7  real*8 :: h=0.1d0, y0=0d0, eps=1d-6
8  real*8 :: x(0:n), y(0:n), h, t, v, dv, [          ]
9  ! calcula el vector x
10 x(0)=0d0
11 do i=0,n
12     x(i)=x(i-1)+[    ]
13 enddo
14 ! asigna valor a y(0)
15 y(0)=y0
16 ! calcula y(1) a y(n) resolviendo G(x(i),y(i))=0
17 do i=0,n
18    ! asigna aproximacion inicial
19    t=x(i)
20    v=[          ]
21    ! resuelve la ecuacion
22    fg=g(t,v)
23    do j=1,nmax
24       !calcula dv, v y fg
25       dv=fg/[          ]
26       v=v+dv
27       [    ]=g(t,v)
28       ! calcula los errores y efectua el test de parada
```

```
29        errorv=dv/v
30        errory=
31        if (errorv<eps.and.              ) exit
32     enddo
33     if (              ) then
34        print*, 'numero maximo de iteraciones'
35        stop
36     endif
37     ! guarda la solucion en y(i)
38     y(i)=v
39  enddo
40  end program funcion_implicita
```

222. Escribir el código de `g(x,y)`y `gy(x,y)` en el mismo fichero del ejercicio 221, compilarlo y ejecutarlo.

Resultados: $y(0) = 0$, $y(0.1) = 8.911283 \times 10^{-3}$

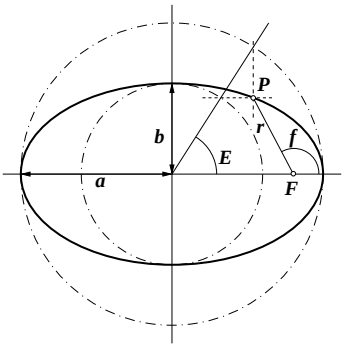
223. Modificar el programa del ejercicio 221 para que $n = 200$ (x debe seguir recorriendo el intervalo $[0, 2]$) y escriba los vectores `x(0:n)`, `y(0:n)` en un fichero. Compilarlo y ejecutarlo. Dibujar la gráfica de la función $y = f(x)$ en $[0, 2]$.

224. La ecuación $G(x, y) \equiv y^3 + x^2y + xe^{-x} - 1 = 0$ define implícitamente a $y = f(x)$ para $x \in \mathbb{R}$. Se pretende determinar una tabla de valores (x, y) en el intervalo $[0, 5]$. Para ello se divide $[0, 5]$ en n subintervalos iguales de amplitud $h = 5/n$ y se definen los puntos $x_i = ih$, $(i = 0, 1, 2, \ldots, n)$.
   a) Determinar $y_0 = f(0)$.
   b) Modificar el programa del ejercicio 222 para calcular $y_i = f(x_i)$.
   c) Con el segmento del programa que resuelve la ecuación $G(t, v) = 0$ construir una subrutina `ec_g(t,v,eps,nmax)`. En la llamada a la subrutina `t` contiene el valor fijo x_i que corresponda, `v` contiene la aproximación inicial, `eps` es la tolerancia de error y `nmax` es el número máximo de iteraciones. En la salida `v` contiene la solución. Compilarlo y ejecutarlo.
   d) Dibujar la gráfica de la función $y = f(x)$ en $[0, 5]$.
   e) Una vez calculado el vector $y(0\!:\!n)$ obtener las expresiones que detrminan las derivadas y' e y'' y calcular los vectores $\mathrm{d}y(0\!:\!n)$ y $\mathrm{d}2y(0\!:\!n)$. Dibujar las gráficas de $y'(x)$ e $y''(x)$.

Resultados: $y(0) = 1$, $y(1) = 0.50405$, $y'(1) = -0.57207$, $y''(1) = 0.37356$

225. El movimiento de los planetas y cometas alrededor del Sol o de un satélite alrededor de la Tierra puede describirse, en primera aproximación, mediante las leyes del movimiento kepleriano (o problema de Kepler).

Bajo ciertas condiciones la órbita es una elipse, con el centro del cuerpo atractor (el Sol o la Tierra) situado en uno de sus focos. La ley horaria del movimiento, que determina la posición a lo largo de la órbita, está dada por la ecuación de Kepler



$$F(t,E) \equiv E - e\,\mathrm{sen}\,E - 2\pi\frac{t}{T} = 0,$$

donde E: anomalía excéntrica, e: excentricidad de la órbita, T: período de la órbita y t: tiempo transcurrido desde el paso por el pericentro (punto más próximo al foco).

Dado t, la solución de la ecuación $F(t,E)=0$ determina la anomalía excéntrica, $E(t)$, a partir de la cual se obtienen los valores de la anomalía verdadera, $f(t)$, y de la distancia al centro atractor, $r(t)$, mediante

$$\mathrm{tg}\,\frac{f}{2} = \sqrt{\frac{1+e}{1-e}}\;\mathrm{tg}\,\frac{E}{2}, \qquad r = a(1-e\cos E),$$

siendo a el semieje mayor de la elipse (el menor es $b = a\sqrt{1-e^2}$). Respecto de unos ejes situados en el foco, las coordenadas cartesianas de la posición están dadas por $x(t) = r(t)\cos\big(f(t)\big)$, $y(t) = r(t)\,\mathrm{sen}\,\big(f(t)\big)$.

Se pretende calcular la posición a lo largo de la órbita en el caso $a = 1, b = 0.6, T = 1$.

a) Modificar el programa del ejercicio 224 para calcular los valores $E_i = E(t_i)$ en los puntos $t_i = ih$ $(i = 0,\ldots,n)$ resolviendo la ecuación $F(t,E)=0$ siendo $n = 100$ y $h = 1/n$.

b) Una vez calculado $E(t_i)$ calcular $f(t_i)$, $r(t_i)$, $x(t_i)$ e $y(t_i)$. El programa debe proporcionar cuatro vectores: $t(0:n)$, $E(0:n)$, $x(0:n)$, $y(0:n)$, con los valores t_i, $E(t_i)$, $x(t_i)$ e $y(t_i)$ $(i = 0,\ldots,n)$.

c) Dibujar, en un mismo gráfico, las curvas $E(t)$, $x(t)$ e $y(t)$.

d) Dibujar los puntos $(x(t_i), y(t_i))$, $(i = 0,\ldots,n)$ que representan la posición del cuerpo en la órbita a intervalos $\Delta t = T/n$.

3.3 Ejercicios

301. Escribir una función real `norma_2(v,n)`, siendo `v` un vector real de tamaño `n`, tal que dado `v(n)` calcule su norma euclídea.

302. El punto de intersección de la circunferencia x^2+y^2-1 y la cúbica $y=x^3$ se obtiene resolviendo el sistema:

$$\begin{aligned} f_1(x_1,x_2) &\equiv x_1^2+x_2^2-1=0 \\ f_2(x_1,x_2) &\equiv x_1^3-x_2 \quad\;\; =0 \end{aligned}$$

a) Escribir una subrutina `f(x,y)`, siendo `x` e `y` vectores de dimensión 2, tal que dado el vector `x` calcule el vector de términos independientes del sistema lineal que se obtiene al aplicar el método de Newton y lo guarde en `y`.

b) Escribir una subrutina `jacob_f(x,a)`, siendo `a` una matriz cuadrada de dimensión 2, tal que dado `x` calcule la matriz jacobiana del sistema y la guarde en `a`.

c) El siguiente programa corresponde al algoritmo 3.2 para resolver el sistema no lineal mediante el método de Newton. El programa utiliza las subrutinas `f`, `jacob_f`, `factor` y `solve` y la función `norma_2`. Obsérvese que dado que en la subrutina `solve` se usa el mismo vector para el término independiente y para la solución del sistema, el mismo vector y representará a los vectores $\Delta\boldsymbol{x}$ y $-\boldsymbol{f}(\boldsymbol{x})$. Completar y corregir el programa.

```
1  program newton_sistemas
2  implicit none
3  integer, parameter :: m=2
4  integer :: i,n,nmax=10
5  real*8 :: x(m),a(m,m),y(m),errf,errx,norma_2,eps=1d-6
6  ! asigna valores iniciales
7  x(1)=sqrt(1d0/2d0)
8  x(2)=
9  n=0
10 ! calcula la función para la aproximacion inicial
11 call f(x,y)
12 ! comienza el ciclo de iteraciones
13 do n=1,nmax
14     ! calcula el jacobiano
15     call
16     ! calcula la descomposicion lu del jacobiano
17     call factor(a,n)
18     ! resuelve el sistema df*dx=-f
19
```

```
20      ! calcula la nueva aproximacion
21      do i=0,m
22          x(i)=
23      end do
24      ! calcula el error relativo de la correccion
25      errx=norma_2(x,m)/norma_2(y,m)
26      ! calcula la funcion para la nueva aproximacion
27      call f(x,y)
28      ! calcula el error de f
29      =norma_2(y,m)
30      ! escribe resultados
31      write(10,50) n,(x(i),i=1,m),errx,errf
32      if (min(errx,errf).lt.eps) exit
33  enddo
34  if (n.gt.nmax) write(6,'(''numero maximo de iteraciones'')')
35  50    format (i2,4(1x,d13.7))
36  end program newton_sistemas
```

303. Escribir el programa anterior junto con los códigos de las subrutinas `f`, `jacob_f`, `factor` y `solve` y la función `norma_2` en un mismo fichero. Compilarlo y ejecutarlo. Comprobar los resultados.

Resultados: $x = 0.82603$, $y = 0.56362$

304. Cambiar la aproximación inicial del problema 303 por $(-1/\sqrt{2}, -1/\sqrt{2})$ y comprobar que el método converge a la solución situada en el tercer cuadrante.

305. Mediante el método de Newton con $\boldsymbol{x}^{(0)} = \mathbf{0}$ resolver el siguiente sistema con $\varepsilon = 10^{-8}$:

$$4x_1^2 - 20x_1 + \frac{1}{4}x_2^2 + 8 = 0$$
$$\frac{1}{2}x_1x_2^2 + 2x_1 - 5x_2 + 8 = 0$$

Resultados: $x_1 = 0.5$, $x_2 = 0.2$

306. Mediante el método de Newton con $\boldsymbol{x}^{(0)} = (1, 1, 1)$ resolver el siguiente sistema con $\varepsilon = 10^{-6}$:

$$x_1^2 + x_2 - 37 = 0$$
$$x_1 - x_2^2 - 5 = 0$$
$$x_1 + x_2 + x_3 - 3 = 0$$

Resultados: $x_1 = 6$, $x_2 = 1$, $x_3 = -4$

307. Calcular mediante el método de Newton los puntos de intersección de las elipses:

$$3x^2 + y^2 + 9x - y - 12 = 0$$
$$x^2 + 36y^2 - 36 = 0$$

Representar de forma aproximada las elipses para saber cuántos puntos de intersección hay y elegir las aproximaciones iniciales.

Resultados: $(1.000921, 0.985987)$, $(-4,012683, 0.743460)$, $(-3.909423, -0.758588)$, $(0.865110, -0.989551)$

308. Considérense las parábolas $y = x^2$ e $y = 1 - (x-3)^2$. Ambas curvas pueden representarse en forma paramétrica como $\boldsymbol{x}_1(u) = (u, u^2)$ y $\boldsymbol{x}_2(v) = (v, 1-(v-3)^2)$. La distancia entre 2 puntos cualesquiera de las mismas está dada por

$$\delta(u,v) = \|\boldsymbol{x}_1(u) - \boldsymbol{x}_2(v)\|_2 = \sqrt{(u-v)^2 + (u^2 - 1 + (v-3)^2)^2}.$$

Un dibujo esquemático de ambas curvas muestra que no se cortan y que $\delta(u,v)$ alcanza un mínimo. El punto de cada curva en que se alcanza el mismo está determinado por la solución del sistema $\partial\delta(u,v)/\partial u = \partial\delta(u,v)/\partial v = 0$. Plantear el sistema anterior (sugerencia: tener en cuenta que cada una de las derivadas parciales es una fracción en la que el denominador no se anula). Resolverlo mediante el método de Newton con $\varepsilon = 10^{-6}$. Elegir la aproximación inicial utilizando el dibujo de las curvas. Obtener el valor de la distancia mínima.

Resultados: $u = 0.908560$, $v = 2.091440$, $\delta = 1.350169$

309. Determinar el valor mínimo de la distancia entre la cúbica $y = x^3$ y la circunferencia $(x-3)^2 + y^2 = 1$ y el punto de cada curva en que se alcanza. Utilizar las representaciones paramétricas $\boldsymbol{x}_1(u) = (u, u^3)$ y $\boldsymbol{x}_2(v) = (3 + \cos v, \operatorname{sen} v)$.

Resultados: $u = 0.928600$, $v = 2.772722$, $\delta = 1.220781$

310. La ecuación $F(x,y) \equiv y - x^2 + x - e^{-yx^2} = 0$ define implícitamente a una función $y(x)$ tal que $F(x, y(x)) = 0$.
   a) Calcular $y(0)$.
   b) Mediante el método de Newton calcular una tabla de valores de $y(x)$ en $[0,2]$ para $x_i = 0.02i$, $(i = 0, \ldots, 100)$.
   c) Dibujar la gráfica de $y(x)$ en $[0,2]$.
   d) La gráfica anterior muestra que $y(x)$ presenta un mínimo en $]0,2[$. Obtener el sistema de ecuaciones que hay que resolver para calcularlo.
   e) Resolver el sistema anterior mediante el método de Newton. Elegir la aproximación inicial a la vista de la gráfica.

Resultados: $(x,y)_{\text{mín}} = (0.808714, 0.545322)$

311. Considérese el sistema de ecuaciones

$$F_1 \equiv x + 2y - \mathrm{e}^z + 1 = 0$$
$$F_2 \equiv x - \mathrm{e}^y - 2z + 1 = 0$$

Sea C el conjunto de puntos de $\mathbb{R}^3$ que verifican el sistema anterior, es decir $C = \{(x, y, z) \in \mathbb{R}^3 / F_1(x, y, z) = F_2(x, y, z) = 0\}$.

a) Comprobar que el sistema anterior define a 2 funciones $y = y(x)$ y $z = z(x)$ en todos los puntos de C.

b) Calcular $y(0)$ y $z(0)$.

c) Sean $x_i, i = 0, 1, 2, \ldots, n$, los puntos del intervalo $[0, 5]$ que se obtienen al dividirlo en $n = 150$ subintervalos de igual amplitud. Comenzando con los valores calculados en el apartado anterior, construir una tabla con los valores $x_i, y(x_i), z(x_i)$ para $i = 0, 1, \ldots, n$.

d) Dibujar las gráficas de $y = y(x)$ y $z = z(x)$ en $[0, 5]$.

Resultados: $y(0) = z(0) = 0$, $y(5) = 0.693441$, $z(5) = 1.999706$

312. Considérese el sistema de ecuaciones

$$z^2 + xy = 2$$
$$z^2 + x^2 - y^2 = 1$$

a) Comprobar que el sistema anterior define a 2 funciones $x = x(z)$ e $y = y(z)$ en todos los puntos tales que $(x, y) \neq (0, 0)$.

b) Calcular $x(1)$ e $y(1)$ tales que $x(1) > 0$.

c) Construir una tabla de valores de las funciones $x = x(z)$ e $y = y(z)$ para $z = \{1, 1.05, 1.10, 1.15, \ldots, 3\}$.

d) Dibujar las gráficas de $x = x(z)$ e $y = y(z)$ en $[1, 3]$.

e) Calcular los valores de las funciones derivadas $x'(z)$ y $y'(z)$ en los mismos puntos z_i y dibujar sus gráficas.

Resultados: $x(1) = y(1) = 1$, $x(3) = -2.015504$, $y(3) = 3.473076$

4.5 Ejercicios

401. Aproximar (con lápiz y papel) las siguientes integrales usando la fórmula simple del Trapecio

$$\text{a) } \int_{0.5}^{1} x^4\,\mathrm{d}x, \qquad \text{b) } \int_{0}^{0.35} x\cos x\,\mathrm{d}x, \qquad \text{c) } \int_{0}^{1} x^2 e^{-x}\,\mathrm{d}x.$$

Resultados: a) 0.2656, b) 0.5754×10^{-1}, c) 0.1839

402. Aproximar las integrales del ejercicio 401 mediante la fórmula simple de Simpson.

Resultados: a) 0.1940, b) 0.5939×10^{-1}, c) 0.1624

403. Completar el siguiente código para calcular $\int_0^\pi e^x \cos x\,\mathrm{d}\,x$ mediante la fórmula del Trapecio con $n = 128$ subintervalos. Escribirlo en un fichero, compilarlo y ejecutarlo. Comprobar el resultado.

```
1  program trapecio
2  ! calcula la integral entre 0 y pi de exp(x)*cos(x)
3          implicit none
4          integer :: n,i
5          real*8 :: a, b, x, h, pi, t, s
6  ! asigna datos
7          pi=
8          n=128
9          a=0d0
10         b=pi
11         h=(b-a)/
12 ! calcula suma de las ordenadas 1 a n-1
13         =0d0
14         do i=1,n-1
15             x=a+dble(i)*
16             s=  +exp(x)*cos(x)
17         end do
18 !completa la formula del trapecio
19         t=h*(0.5d0*(exp(a)*cos(a)+              )+s)
20         write(*,‘(‘‘trapecio=’’,d12.5)’) t
21 end program trapecio
```

Resultados: -0.12072×10^2

404. Escribir una función `f(x)` para calcular $e^x \cos x$. Modificar el programa del ejercicio 403 para calcular las ordenadas mediante llamadas a la función `f(x)`. Escribirlo en un fichero, compilarlo y ejecutarlo.

405. Completar y corregir el siguiente código para calcular $\int_0^\pi e^x \cos x \, \mathrm{d}\, x$ mediante la fórmula de Simpson con $n = 128$ subintervalos. Las ordenadas se calculan mediante el subprograma función `f(x)` escrito en el ejercicio 404. Escribirlo en un fichero junto con el código de `f(x)`, compilarlo y ejecutarlo. Comprobar el resultado.

```
1  program simpson
2          implicit none
3          integer :: n,i
4          real*8 :: a, b, pi, s1, s2, x, h, s
5          real*8, external :: f
6          pi=acos(-1)
7          n=128
8          a=0d0
9          b=pi
10         h=[                ]
11 ! calcula la suma de las ordenadas impares
12         s1=0
13         do i=2,n-1,2
14              x=a+[                ]
15              [    ]=s1+f(x)
16         end do
17 ! calcula la suma de las ordenadas pares
18         s2=0
19         do i=1,[          ],2
20              x=a+dble(i)*h
21              [                ]
22         end do
23 !completa la formula de simpson
24         s=(h/3d0)*(f(a)+[                  ]+f(b))
25         write(*,‘(‘‘simpson=’’,d12.5)’) s
26 end program simpson
```

Resultados: -0.12070×10^2

406. A partir del programa del ejercicio 404 escribir una subrutina `trapecio(a,b,n,t)` para aproximar $\int_a^b f(x)\,\mathrm{d}\,x$ mediante la fórmula del trapecio con n subintervalos. Los datos que hay que proporcionar a la subrutina son `a`, `b` y `n`. La subrutina devolverá el resultado en la variable `t`. Para calcular las ordenadas se considera disponible un subprograma función `f(x)`. Escribir la subrutina en un fichero llamado `trapecio.f90`, compilarlo y corregir los errores.

407. A partir del programa del ejercicio 405 escribir una subrutina `simpson(a,b,n,s)` para aproximar $\int_a^b f(x)\,\mathrm{d}\,x$ mediante la fórmula de Simpson con n subintervalos. Los datos que hay que proporcionar a la subrutina son `a`, `b` y `n`. La subrutina comprobará que `n` es par y devolverá el resultado en la variable `s`. Para calcular las ordenadas se considera disponible un subprograma función `f(x)`. Escribir la subrutina en un fichero llamado `simpson.f90`, compilarlo y corregir los errores.

408. Escribir un programa para calcular integrales utilizando las subrutina `trapecio(a,b,n,t)`. Para aplicarlo a un caso concreto será necesario especificar `a`, `b` y `n` así como la función `f(x)`. Aplicarlo para calcular con `n=50` las integrales del ejercicio 401.

Resultados: a) 0.193779, b) 0.593862×10^{-1}, c) 0.160615

409. Calcular las siguientes integrales mediante el método del Trapecio. El valor de `n` debe ser tal que en todos los casos `h` ≈ 0.01.

a) $\displaystyle\int_0^4 \frac{1}{1+x^2}\,\mathrm{d}\,x$, b) $\displaystyle\int_{-2}^2 \frac{2+x^2}{(1+x^2)(4+x^2)}\,\mathrm{d}\,x$, c) $\displaystyle\int_\pi^{2\pi} e^{-2x}\,\mathrm{sen}\,\frac{x}{2}\,\mathrm{d}\,x$,

d) $\displaystyle\int_0^{\pi/2} \frac{1}{4\,\mathrm{sen}\,x + 3\cos x}\,\mathrm{d}\,x$, e) $\displaystyle\int_0^1 \frac{x+2}{\sqrt{x^2+x+3}}\,\mathrm{d}\,x$.

Resultados: a) 1.325817, b) 1.261696, c) 0.879238×10^{-3}, d) 0.358357, e) 1.276684

410. Realizar el ejercicio análogo al 408 pero para el método de Simpson.

Resultados: a) 0.193750, b) 0.593869×10^{-1}, c) 0.160603

411. En la lista de argumentos de la subrutina `trapecio(a,b,n,t)` figuran todos los datos necesarios para calcular la integral, excepto la función integrando que debe llamarse siempre `f(x)`. Puede escribirse la subrutina de forma que el nombre de la función integrando sea también un dato. Para ello en la lista de argumentos formales debe aparecer el nombre de la función tal y como se emplea en la subrutina. (Ver sección §9.3 de la primera parte de la asignatura).

El siguiente programa calcula las integrales de los apartados a) y c) del ejercicio 409. La subrutina `trapecio_f(a,b,n,f,t)` es la misma del ejercicio 406 sin más que añadir `f` en la lista de argumentos. Las funciones correspondientes se han incorporado como `function fa(x)` y `function fc(x)`. Completarlo y corregir los errores. Escribirlo en un fichero, compilarlo y ejecutarlo.

```
1 program ejercicio_411
2 implicit none
3 integer :: n
4 real*8 :: a,b,t,pi,h
5 real*8,external ::fa,fc
6 pi=acos(-1d0)
```

```
! calcula la integral de fa
h=1d-2
a=0d0
b=4d0
n=nint((b-a)/h)
call trapecio_f(a,b,n,fa,t)
write(*,``integral de fa'', d15.8)') t
!calcula la integral de fc
h=1d-2
a=pi
b=2d0*pi
n=nint((b-a)/h)
call trapecio_f(                    )
write(*,`(``integral de fc='', d15.8)')
end program ejercicio411
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
subroutine trapecio_f(a,b,n,f,t)
! calcula la integral entre a y b de f(x) con n subintervalos
implicit none
integer :: n,i
real*8 :: a, b, x, h, t, s
real*8, external :: f
      =(b-a)/dble(n)
! calcula suma de las ordenadas 1 a n-1
s=0d0
do i=1,n
    x=a+dble(i)*h
    s=f(x)
end do
!completa la formula del trapecio
t=h*(                +s)
return
end subroutine trapecio_f
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
function fa(x)
implicit none
real*8 :: x,f
```

```
44 f=1d0/(1d0+x*x)
45 return
46 end function fa
47 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
48 function fc(x)
49 implicit none
50 real*8 :: x,f
51 f=exp(-2d0*x)*sin(x/2d0)
52 return
53 end function fc
```

412. Ampliar el programa del ejercicio 411 para calcular todas las integrales del ejercicio 409. Compilarlo, ejecutarlo y comprobar los resultados.

413. a) Escribir una subrutina `simpson_f(a,b,n,f,s)` análoga a la del ejercicio 407 pero incluyendo el nombre de la función en la lista de argumentos formales. Compilarla y corregir los errores.
b) Escribir un programa que utilice la subrutina anterior para calcular las integrales del ejercicio 409. Compilarlo, ejecutarlo y comprobar los resultados.

414. Calcular las integrales del ejercicio 409 con el método de Simpson y `h` ≈ 0.02.

Resultados: a) 0.193750, b) 0.593869×10^{-1}, c) 0.160603

415. La ecuación $F(x, y) \equiv y + x^3 - x^2\,\mathrm{e}^{-y} = 0$ define implícitamente a una función $y(x)$ tal que $F\big(x, y(x)\big) = 0$.
a) Calcular $y(0)$. Calcular una tabla de valores de $y(x)$ en los puntos $x_i = ih$ $(i = 0, 1, \ldots, n)$ siendo $n = 100$ y $h = 1/n$ (ver ejercicios 221-224).
b) Obtener la expresión analítica de $y'(x)$. La longitud de la curva $y = y(x)$ comprendida entre $x = 0$ y $x = 1$ está dada por

$$L = \int_0^1 \sqrt{1 + \big(y'(x)\big)^2}\,\mathrm{d}\,x.$$

Escribir una subrutina `LONGITUD(x,y,n,L)` para calcular L, donde x e y son los vectores obtenidos en el apartado anterior y $n = 100$. La integral se aproximará mediante el método de Simpson. Calcular L.

Resultados: $y(0) = 0$, $y(0.5) = 0.100986$, $y(1) = 0$, $L = 1.0299$

416. El siguiente programa se ha construido a partir del ejercicio 411 para calcular $\displaystyle\int_0^4 \frac{1}{1+x^2}\,\mathrm{d}\,x$. Se ha introduciendo un bucle `do` que en cada paso duplica `n` y calcula la fórmula del Trapecio correspondiente a ese valor de `n`. Se obtienen así sucesivas aproximaciones que permiten estimar la exactitud de cada una mediante $|(T_n - T_{n/2})/T_n|$. En lugar de fijar el número de subintervalos, se establece una tolerancia de error, ε, y n se va duplicando hasta que $|(T_n - T_{n/2})/T_n| < \varepsilon$ y entonces se acepta T_n como resultado. Además de ε hay que establecer un número máximo de duplicaciones, $k_{\text{máx}}$. Completar el programa y corregirlo. Escribirlo en un fichero junto con la

subrutina `trapecio_f` y la función `fa`, compilarlo y ejecutarlo.

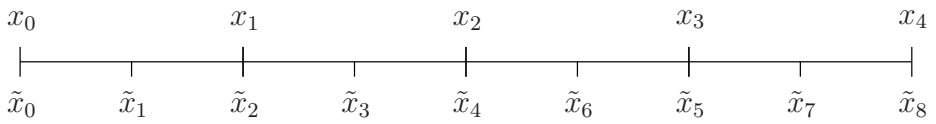
```
1  program ejercicio_416
2  implicit none
3  integer :: n,kmax=20,k
4  real*8 :: a,b,t0,t1,eps=1d-6,error
5  real*8, external :: fa
6  ! asigna valores a los datos
7  [    ]=0d0
8  b=[    ]
9  ! calcula el trapecio con 1 subintervalo
10 n=1
11 call trapecio_f(a,b,n,fa,t0)
12 write(*,‘(‘‘n=’’,i4,2x,‘‘t=’’, d15.8)’) n,t0
13 !comienzan las duplicaciones
14 do k=1,kmax
15   n=n+2
16   !calcula el trapecio con n subintervalos
17   call trapecio_f(a,b,n,fa,t1)
18   ! calcula la estimación del error
19   error=abs((t1-t0)/t0)
20   ! escribe resultados
21   write(*,‘(‘‘n=’’,i4,2x,‘‘t=’’, d15.8,2x, ‘‘error=’’, d10.3)’) n,t1,error
22   ! test de parada
23   if (error.lt.eps) exit
24   !guarda t1 en t0
25   [          ]
26 enddo
27 end program ejercicio416
```

Resultados: $n = 512$, $t = 0.13258175 \times 10^1$, error$= 0.319 \times 10^{-6}$

417. Escribir un programa análogo al del ejercicio 416 pero que calcule la integral mediante el método de Simpson duplicando sucesivamente el número de subintervalos hasta que $|(S_n - S_{n/2})/S_n| < \varepsilon$, siendo ε la tolerancia de error. Se utilizará la subrutina `simpson_f(a,b,n,f,s)`. Escribir el programa junto con la subrutina y la función `fa`, compilarlo y ejecutarlo. Comparar el número de subintervalos necesarios en los métodos del Trapecio y de Simpson para un mismo valor de la tolerancia de error.

Resultados: $n = 64$, $s = 0.13258177 \times 10^1$, error$= 0.165 \times 10^{-7}$

418. En el ejercicio 416 se calcularon los valores de T_1, T_2, T_4, T_8, etc. mediante sucesivas llamadas a la subrutina `trapecio_f(a,b,n,f,s)` con el valor apropiado de `n`. Este procedimiento, aunque sencillo, no es el más apropiado ya que al calcular, por ejemplo, T_8 se puede aprovechar el trabajo realizado para calcular T_4.



En efecto, para calcular T_4 necesitamos evaluar $f(x)$ en los puntos $x_i = a+ih_4$ para $i = 0, 1, 2, 3, 4$ siendo $h_4 = (b-a)/4$. Al duplicar el número de subintervalos para calcular T_8 necesitamos calcular $f(x)$ en los puntos $\tilde{x}_i = a + ih_8$ para $i = 0, 1, 2, \ldots, 8$, siendo $h_8 = (b-a)/8$, que son los mismos puntos que teníamos en T_4, ya que $x_0 = \tilde{x}_0,\ x_1 = \tilde{x}_2,\ x_2 = \tilde{x}_4,\ x_3 = \tilde{x}_6,\ x_4 = \tilde{x}_8$, más los puntos medios entre x_i y x_{i+1}.

Para ver cómo puede aprovecharse el cálculo de T_4 en el cálculo de T_8, escribimos la expresión de T_8 y "separamos" los valores de $f(x)$ calculados en T_4 de los "nuevos":

$$\begin{aligned}
T_8 &= h_8\Big(\frac{1}{2}f(\tilde{x}_0) + f(\tilde{x}_1) + f(\tilde{x}_2) + f(\tilde{x}_3) + f(\tilde{x}_4) + f(\tilde{x}_5) + f(\tilde{x}_6) + f(\tilde{x}_7) + \frac{1}{2}f(\tilde{x}_8)\Big) \\
&= h_8\Big(\frac{1}{2}f(\tilde{x}_0) + f(\tilde{x}_2) + f(\tilde{x}_4) + f(\tilde{x}_6) + \frac{1}{2}f(\tilde{x}_8)\Big) + h_8\Big(f(\tilde{x}_1) + f(\tilde{x}_3) + f(\tilde{x}_5) + f(\tilde{x}_7)\Big) \\
&= \frac{1}{2}h_4\Big(\frac{1}{2}f(x_0) + f(x_1) + f(x_2) + f(x_3) + \frac{1}{2}f(x_4)\Big) + h_8\Big(f(\tilde{x}_1) + f(\tilde{x}_3) + f(\tilde{x}_5) + f(\tilde{x}_7)\Big) \\
&= \frac{1}{2}T_4 + h_8\Big(f(\tilde{x}_1) + f(\tilde{x}_3) + f(\tilde{x}_5) + f(\tilde{x}_7)\Big)
\end{aligned}$$

Por tanto para calcular T_8 podemos aprovechar T_4 y sólo necesitamos evaluar $f(x)$ en los puntos "nuevos"que han aparecido al duplicar el número de subintervalos. El resultado puede generalizarse llegando a la siguiente relación entre $T_{n/2}$ y T_n:

$$T_n = \frac{1}{2}T_{n/2} + h\sum_{i=1}^{n/2} f(a + (2i-1)h),$$

siendo $h = (b-a)/n$.

La siguiente subrutina calcula la aproximación a la integral mediante la regla del Trapecio, duplicando sucesivamente el número de subintervalos, hasta que $|(T_n - T_{n/2})/T_n| < \varepsilon$. Completarla y corregirla. Escribir un programa para calcular todas las integrales del ejercicio 409 utilizando esta subrutina y las funciones necesarias. Compilarlo, corregir los errores y ejecutarlo. Comprobar los resultados.

```
1 subroutine trapecio_dup(a,b,f,eps,kmax,t)
2 implicit none
3 integer :: n,kmax,k,i
4 real*8 :: a,b,t0,t,eps,error,h,s,x
5 real*8, external :: f
6 ! calcula trapecio con 1 subintervalo
7 n=1
```

```
h=(b-a)/dble(n)
t0=          *(f(a)+f(b))
!comienzan las duplicaciones
do k=1,kmax
  n=n/2
  h=2d0*h
  !calcula la suma de las ordenadas impares
     =0
  do i=1,n/2
  x=     +dble(2*i-1)*h
  s=s+
  end do
  ! calcula trapecio con n subintervalos
  t=.5d0*t0+
  ! calcula la estimación del error
  error=(t-t0)/t
  ! test de parada
  if (error.lt.eps) return
  !guarda t en t0
  t=t0
enddo
write(*,*) 'maximo numero de duplicaciones'
return
end subroutine trapecio_dup
```

419. En el ejercicio 417 se calcula la aproximación al método de Simpson duplicando el número de subintervalos mediante sucesivas llamadas a la subrutina `simpson_f(a,b,n,f,s)`. Este procedimiento presenta inconvenientes análogos a los encontrados con el método del Trapecio. Sería conveniente disponer de una estrategia que, al igual que en el método del Trapecio, nos permitiese calcular S_n utilizando el valor de $S_{n/2}$ Sin embargo no existe una relación entre S_n y $S_{n/2}$ similar a la que existe entre T_n y $T_{n/2}$. Pero S_n puede calcularse a partir de T_n y $T_{n/2}$ mediante $S_n = (4T_n - T_{n/2})/3$, lo que en definitiva permite calcular S_n con el número mínimo posible de evaluaciones de la función integrando.

La siguiente subrutina calcula la aproximación a la integral mediante la regla de Simpson, duplicando sucesivamente el número de subintervalos hasta que $|(S_n - S_{n/2})/S_n| < \varepsilon$. Cada valor de S_n se calcula a partir de dos valores de la fórmula del Trapecio, T_n y $T_{n/2}$. Completarla y corregirla. Escribir un programa para calcular todas las integrales del ejercicio 409 utilizando esta subrutina y las funciones necesarias. Compilarlo, corregir los errores y ejecutarlo. Comprobar los resultados.

```
subroutine simpson_dup(a,b,f,eps,kmax,s)
```

```
2  implicit none
3  integer :: n,kmax,k,i
4  real*8 :: a,b,t0,t1,eps,error,h,s,x,s0
5  real*8, external :: f
6  ! calcula trapecio con 1 subintervalo
7  n=1
8  h=(b-a)/dble(n)
9  t0=.5d0*h*(f(a)+f(b))
10 ! calcula trapecio  y simpson con 2 subintervalos
11 n=2*n
12 h=0.5*h
13 t1=.5d0*t0+h*f(a+h)
14 s=(4d0*t1-    )/3d0
15 !guarda t1 y s en t0 y s0
16 t0=
17
18 !comienzan las duplicaciones
19 do k=1,kmax
20   n=
21   h=
22   !calcula trapecio con n subintervalos
23   !calcula la suma de las ordenadas impares
24   =0d0
25   do i=1,n/2
26    x=a+dble(2*i-1)*h
27
28   end do
29   ! calcula nuevo trapecio
30   t1=
31   !calcula simpson con n subintervalos
32   s=
33   ! calcula la estimación del error
34   error=abs((s-s0)/s0)
35   ! test de parada
36   if (error.lt.eps) return
37   !guarda t1 y s en t0 y s0
```

```
38      [          ]
39      [          ]
40  enddo
41  write(*,*) 'maximo numero de duplicaciones'
42  return
43  end subroutine simpson_dup
```

420. Para calcular $I = \iint f(x,y)\,\mathrm{d}\,x\,\mathrm{d}\,y$ en el intervalo $[a,b]\times[c,d]$ mediante la fórmula del Trapecio basta con disponer de un subprograma función para calcular $g(x) = \int_c^d f(x,y)\,\mathrm{d}\,y$ que nos permitirá calcular I mediante $\int_a^b g(x)\,\mathrm{d}\,x$.

Completar el siguiente código de una función `g_trap(x,c,d,h)` para calcular $g(x) = \int_c^d f(x,y)\,\mathrm{d}\,y$ mediante la fórmula del trapecio con m subintervalos de amplitud k tal que $k \approxeq h$.

```
1  function g_trap(x,c,d,h)
2  ! calcula para x dado la integral entre c y d de f(x,y) con m subintervalos
3    implicit none
4    integer :: m,j
5    real*8 :: [                    ]
6    real*8, external :: f
7  ! calcula m y k
8    m =[                    ]
9    k=(d-c)/dble(m)
10 ! calcula suma de las ordenadas 1 a m-1
11   s=0d0
12   do j=[                    ]
13      y=c+dble(j)*k
14      s=s+[                    ]
15   end do
16 ! completa la formula del trapecio
17   g_trap=k*(0.5d0*[                                ]+s)
18   return
19 end function g_trap
```

421. Utilizando la función `g_trap` del ejercicio 420 escribir un programa para calcular $I = \iint x\,\mathrm{e}^{-xy}\,\mathrm{d}\,x\,\mathrm{d}\,y$ en el intervalo $[0,2]\times[-1,1]$ mediante la fórmula del Trapecio. El programa principal será muy parecido al del ejercicio 404 salvo que habrá que reemplazar `f(x)` por `g_trap(x,c,d,h)`. Escribirlo en un fichero junto con el código de la función `g_trap` y el de una función para calcular $f(x,y) = x\,\mathrm{e}^{-xy}$. Compilarlo y ejecutarlo tomando $n = 100$. Comparar el resultado con el valor exacto $I = (\mathrm{e} - \mathrm{e}^{-1})^2$.

Resultados: 5.5250

422. Modificar el ejercicio 421 para calcular $I = \iint x\,\mathrm{e}^{-xy}\,\mathrm{d}\,x\,\mathrm{d}\,y$ en el conjunto $C = \{(x, y) \in \mathbb{R}^2, 0 \leq x \leq 2, u(x) \leq y \leq v(x)\}$ siendo $u(x) = x^2 - 2x + 2$ y $v(x) = 3\,\mathrm{e}^x$ (para cada valor de x, $c = u(x)$ y $d = v(x)$). Tomar $n = 200$ y utilizar dos subprogramas función para calcular $u(x)$ y $v(x)$. Una vez compilado y ejecutado añadir en la función `g_trap` una sentencia (provisional) para mostrar en cada llamada los valores de m y k y comprobar que en todos los casos $k \approxeq h$.

Resultados: 0.518074

423. Calcular mediante la fórmula del Trapecio $\displaystyle\iint_C \frac{(x-1)^2 + y^2}{x-1}\,\mathrm{d}\,x\,\mathrm{d}\,y$ siendo $C = \{(x, y) \in \mathbb{R}^2, x \geq 2, (x-2)^2 + y^2 \leq 1\}$ (solución exacta: $2\pi/3 + 4/9$).

424. Calcular mediante la fórmula del Trapecio $\iint_C x^2y^2\,\mathrm{d}\,x\,\mathrm{d}\,y$ siendo C la región comprendida entre $y = 4x$, $y = x/2$, $xy = 1$ y $xy = 2$ (solución exacta: $7\ln 2/2$).

425. El volumen de un sólido encerrado por 2 superficies que admiten representación explícita $z = z_s(x, y)$ y $z = z_i(x, y)$ y tales que $z_i(x, y) \leq z \leq z_s(x, y)$ puede reducirse a una integral doble sin más que plantear la integral triple e integrando sobre z reducirla a $\iint_C (z_s(x, y) - z_i(x, y))\,\mathrm{d}\,x\,\mathrm{d}\,y$. C es la región del plano en la que se verifica $z_i(x, y) \leq z_s(x, y)$ y su frontera es la proyección sobre $z = 0$ de la intersección de ambas superficies, es decir $z_i(x, y) = z_s(x, y)$.
Calcular mediante la fórmula del Trapecio el volumen del sólido acotado encerrado entre las superficies $x^2 + y^2 = 2z + 1$ y $x^2 + y^2 = 6 - 3z$ (solución exacta: $15\pi/4$)

426. Calcular mediante la fórmula del Trapecio el volumen del sólido acotado encerrado entre las superficies $z = 2x^2 + y^2$ y $z = 4 - y^2$ (solución exacta: 4π)

427. Escribir el código de una función `g_simp(x,c,d,h)` para calcular $g(x) = \int_c^d f(x, y)\,\mathrm{d}\,y$ mediante la fórmula de Simpson con m subintervalos de amplitud k tal que $k \approxeq h$ (téngase en cuenta que m debe ser par). Escribirlo en un fichero, compilarlo y corregir los errores.

428. Resolver el ejercicio 421 mediante la fórmula de Simpson.

429. Resolver el ejercicio 422 mediante la fórmula de Simpson.

430. Calcular mediante la fórmula de Simpson $\iint_C (y + 2x)\,\mathrm{d}\,x\,\mathrm{d}\,y$ siendo $C = \{(x, y) \in \mathbb{R}^2, 0 \leq y, x^2 + y^2 \leq 2x, y \leq x^2\}$ (solución exacta: $8/5 + \pi/2$).

431. Calcular mediante la fórmula de Simpson $\iint_C xe^{y-x}\,\mathrm{d}\,x\,\mathrm{d}\,y$ siendo C la región acotada comprendida entre la parábola $y = x - x^2$ y sus tangentes en los puntos (0,0) y (2,-2) (solución exacta: $(5 - e^{-4})/16$).

432. Comprobar los resultados del cálculo de $\iint \mathrm{e}^{y-x}\,\mathrm{d}x\,\mathrm{d}y$ en $[0, 1] \times [0, 1]$ que se muestran en la página 55 al aplicar las fórmulas del Trapecio y Simpson para $n = 2, 2^2, 2^3, \ldots, 2^8$.

433. Las ecuaciones

$$y + \mathrm{e}^{x^2 y} + \mathrm{e}^{-x^2 z} - 2 = 0$$
$$y + z + \mathrm{e}^{-x^2} - 1 = 0$$

determinan a 2 funciones $y(x)$ y $z(x)$, tales que $y(0) = z(0) = 0$.

a) Calcular una tabla de valores de las funciones $y(x)$ y $z(x)$ para $x = 2i/n$ con $n = 200$ e $i = 0, 1, \ldots, n$ (ver ejercicios 311 y 312).

b) Calcular $\displaystyle\iint_D \sqrt{x^2 + t}\, \mathrm{e}^{-xt}\, \mathrm{d}x\, \mathrm{d}t$ siendo $D = \{(x, y) \in \mathbb{R}^2,\ 0 \le x \le 2,\ y(x) \le t \le z(x)\}$

Resultados: $y(2) = 0.149046$, $z(2) = 0.832638$, $I = 0.50593$ (Simpson)

434. Considérese la integral triple $\iiint_C z\, \mathrm{d}x\, \mathrm{d}y\, \mathrm{d}z$ donde $C = \{(x, y, z) \in \mathbb{R}^3, z \ge \sqrt{4x^2 + y^2}, 2x^2 + 3y^2 + z^2 \le 2\}$. Integrando sobre z reducirla a una integral doble en una región D que hay que determinar. Calcular la integral doble mediante la fórmula de Simpson (solución exacta: $\pi/2\sqrt{6}$).

5.5 Ejercicios

501. El siguiente programa corresponde a la integración mediante el método del Trapecio del ejemplo considerado en §5.1 con $y(a) = \alpha$. Completarlo y escribirlo en un fichero. Compilarlo y corregir los errores. Ejecutarlo y comprobar los resultados. Dibujar la gráfica de la función $y(t)$.

```
1  program ejercicio_501
2  implicit none
3  integer,parameter :: n=100
4  integer :: i
5  real*8 :: a,b,t(0:n),y(0:n),h,h2,alfa
6
7  open(10,file='res_ejercicio_501.txt')
8  ! asigna valores a los datos
9  a=0d0
10 b=1d0
11 alfa=
12 h=
13 h2=h/2d0
14 ! asigna valores iniciales
15 t(0)=a
16 y(0)=
17 ! comienza el bucle para avanzar la solución
18 do i=1,
19     t(i)=t(i-1)+
20     =(y(i-1)*(1d0+h2*t(i-1))&
21         +h2*(exp(-sqrt(t(i-1)))+)/(1d0-h2*t(i))
22 end do
23 ! escribe resultados
24 do i=0,n
25     write(10,'(f6.3,1x,f10.7)') t(i),
26 end do
27 end program ejercicio_501
```

502. El siguiente programa se ha construido a partir del ejercicio 501 para resolver el mismo problema de condiciones iniciales de primer orden, pero escrito en la forma general: $y'(t) = f(t)y(t)+g(t), t \in [a,b], y(a) = \alpha$, con $f(t) = t$, $g(t) = e^{-\sqrt{t}}$, $a = 0$, $b = 1$ y $\alpha = 2$. Se ha añadido el código para el cálculo de las funciones $f(t)$ y $g(t)$. Obsérvese el uso de las variables auxiliares p_0, p_1, q_0 y q_1 para evitar recalcular en cada paso los valores $f(t_{i-1})$ y $g(t_{i-1})$. Completar y corregir el programa. Compilarlo y corregir los errores. Ejecutarlo y comprobar los resultados.

```
1  program ejercicio_502
2  implicit none
3  integer,parameter :: n=100
4  integer :: i
5  real*8 :: a,b,t(0:n),y(0:n),h,h2,alfa,p0,p1,q0,q1
6  real*8,external :: f,g
7  open(10,file='res_ejercicio_502.txt')
8  ! asigna valores a los datos
9  a=
10 b=
11 alfa=2d0
12 h=(b-a)/dble(n)
13 h2=
14 ! asigna valores iniciales
15 t(0)=alfa
16 y(0)=a
17 p0=h2*f(t(0))
18 q0=h2*g(t(0))
19 ! comienza el bucle para avanzar la solución
20 do i=0,n
21    =t(i-1)+h
22    p1=h2*f(t(i))
23    q1=
24    y(i)=(y(i-1)*(1d0+p0)+q0+q1)/
25    ! guarda p1 y q1 en p0 y q0
26    p0=
27    =q1
28 end do
29 do i=0,n
30    write(10,'(f6.3,1x,f10.7)')
31 end do
32 end program ejercicio_502
33 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
34 function f(x)
35 implicit none
36 real*8 :: x,f
```

```
f=
return
end function f
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
function g(x)
implicit none
real*8 :: x,g
g=
return
end function g
```

503. Escribir un programa para resolver el problema

$$\frac{\mathrm{d}\,y(t)}{\mathrm{d}\,t} = \frac{2}{t}y(t) - (1+\frac{2}{t})e^{1-t},\ t \in [1,3],\ y(1) = 2.$$

Tomar $h = 0.01$. Compilarlo y corregir los errores. Ejecutarlo y dibujar la gráfica de la función $y(t)$. La solución exacta es $y = e^{1-t} + t^2$, comprobar los resultados.

504. Escribir un programa para resolver el problema

$$\frac{\mathrm{d}\,y(t)}{\mathrm{d}\,t} = -2ty(t) + 5\sin(2t),\ t \in [0,\pi],\ y(0) = 2.$$

Tomar un número apropiado de puntos para que $h \approx 0.01$. Compilarlo y corregir los errores. Ejecutarlo y dibujar la gráfica de la función $y(t)$.

Resultados: $y(\pi) \approx -0.2633713$

505. Escribir un programa para resolver el problema

$$\frac{\mathrm{d}y(t)}{\mathrm{d}t} = -y(t)\cos t + 0.5\,\mathrm{sen}\,2t, \quad t \in [0,2],\ y(0) = 0.$$

Tomar $h = 0.01$. Compilarlo y corregir errores. Ejecutarlo y dibujar la gráfica de la función $y(t)$.

Resultados: $y(2) \approx 0.3120963$, solución exacta $y(t) = \mathrm{e}^{-\mathrm{sen}\,t} + \mathrm{sen}\,t - 1$

506. El siguiente programa corresponde a la integración mediante el método del Trapecio del ejemplo considerado en §5.2 (pág. 71). Completarlo, añadir los códigos de las subrutinas `solve` y `factor` y escribirlo en un fichero. Compilarlo y corregir los errores. Ejecutarlo y comprobar los resultados. Dibujar las gráficas de las funciones $u(t)$ y $v(t)$.

```
program ejercicio_506
implicit none
integer,parameter :: n=200
integer :: i
real*8 :: a,b,t(0:n),u(0:n),v(0:n),&
        h,h2,p0,p1,q0,q1,z(2,2),c(2),alfa(2)
```

```
external :: solve,factor
open(10,file='res_ejercicio_506.txt')
! asigna valores a los datos
a=1d0
b=3d0
alfa(1)=1d0
alfa(2)=2d0
h=(b-a)/dble(n)
h2=h/2d0
! asigna valores iniciales
t(0)=a
u(0)=alfa(1)
v(0)=alfa(2)
p0=h2*(-3d0*t(0))
q0=h2*t(0)*exp(-t(0))
! comienza el bucle para avanzar la solución
do i=1,n-1
   t(i-1)=t(i)+h
   p1=
   q1=
   ! calcula la matriz
   z(1,1)=1d0
   z(1,2)=h2
   z(2,1)=-h2
   z(2,2)=
   ! calcula el termino independiente
   c(1)=v(i-1)+h2*u(i-1)
   c(2)=-h2*v(i-1)+(1d0+p0)*u(i-1)+q1+q0
   !resuelve el sistema
   call factor(z,2)
   call
   !asigna valores
   =c(1)
   =c(2)
   !guarda p1 y q1 en p0 y q0
```

```
42      [          ]
43      [          ]
44  end do
45  do i=0,n
46     write(10,'(f6.3,1x,f10.7,1x,f10.7)') t(i),u(i),v(i)
47  end do
48  end program ejercicio_506
```

507. El ejemplo de §5.2 (pág. 71) puede escribirse en forma más general como:

$$y''(t) = f_2(t)y'(t) + f_1(t)y'(t) + g(t), t \in [a,b], y'(a) = \alpha_2, y(a) = \alpha_1.$$

Introducir las funciones auxiliares $u(t)$ y $v(t)$ y obtener, procediendo como en §5.2, el sistema de ecuaciones que determina u_i y v_i a partir de u_{i-1} y v_{i-1}. Con $f_2(t) = -3t$, $f_1(t) = -1$, $g(t) = te^{-t}$, $a = 1$, $b = 3$, $\alpha_2 = 1$ y $\alpha_1 = 2$ el problema general se reduce al ejemplo de §5.2. El siguiente programa es una modificación del programa del ejercicio 506 para resolver el problema de §5.2, pero escrito en la forma general. Completarlo y corregirlo. Añadir el código de las funciones `f2(t)`, `f1(t)` y `g(t)` y de la subrutinas `solve` y `factor` y escribirlo en un fichero. Compilarlo y corregir los errores. Ejecutarlo y comprobar los resultados.

```
1  program ejercicio_507
2  implicit none
3  integer,parameter :: n=200
4  integer :: i
5  real*8 :: a,b,t(0:n),u(0:n),v(0:n),&
6          h,h2,p0,p1,q0,q1,r0,r1,z(2,2),c(2),alfa(2)
7  external :: solve,factor
8  [                                    ]
9  open(10,file='res_ejercicio_507.txt')
10 ! asigna valores a los datos
11 a=1d0
12 b=3d0
13 alfa(1)=1d0
14 alfa(2)=2d0
15 [                  ]
16 h2=h/2d0
17 ! asigna valores iniciales
18 t(0)=a
19 [        ]=alfa(1)
20 [        ]=alfa(2)
21 p0=h2*f2(t(0))
```

```
22 q0=h2*f1(t(0))
23 r0=h2*g(t(0))
24 ! comienza el bucle para avanzar la solución
25 do i=1,n
26    t(i)=          +h
27    =h2*f2(t(i))
28    =h2*f1(t(i))
29    =h2*g(t(i))
30    ! calcula la matriz
31    z(1,1)=
32    =-h2
33    =-q1
34    z(2,2)=
35    ! calcula el termino independiente
36    =u(i-1)+h2*v(i-1)
37    =q0*u(i-1)+(1d0+p0)*v(i-1)+r1+r0
38    !resuelve el sistema
39
40
41    !asigna valores
42    u(i)=
43    v(i)=
44    !guarda p1, q1 y r1 en p0, q0 y r0
45
46
47
48 end do
49 do i=0,n
50    write(10,'(f6.3,1x,f10.7,1x,f10.7)') t(i),u(i),v(i)
51 end do
52 end program ejercicio_507
```

508. Modificar el programa del ejercicio 507 para resolver el problema

$$y''(t) = (1 - \tan t)y'(t) + \tan t\, y(t),\ t \in [0, 1],\ y'(0) = 0,\ y(0) = 1.$$

Tomar $h = 0.01$. Compilarlo y corregir los errores. Ejecutarlo y dibujar la gráfica de la función $y(t)$. La solución exacta es $y = \frac{1}{2}(e^t + \cos t - \operatorname{sen} t)$, comprobar los resultados.

Resultados: $y(1) \approx 1.208575$

509. Escribir un programa para resolver el problema

$$y''(t) = \sin(t/2)\, y'(t) - \cos(t/4)\, y(t) + \mathrm{e}^{-t^2/4},\ t \in [-\pi, \pi],\ y'(-\pi) = 1,\ y(-\pi) = 2.$$

Tomar $h \approx 0.01$. Compilarlo y ejecutarlo. Dibujar la gráfica de la función $y(t)$.

Resultados: $y(\pi) \approx 1.8699478$

510. Escribir un programa para resolver el problema

$$y'''(t) = 2y''(t) + 4y'(t) - 8y(t),\ t \in \left[0, \frac{\pi}{2}\right],\ y''(0) = 3.36,\ y'(0) = -0.16,\ y(0) = 1.$$

Tomar $h = 0.01$. Compilarlo y ejecutarlo. Calcular analíticamente la solución exacta y comprobar los resultados. Dibujar la gráfica de la función $y(t)$.

Resultados: $y(\pi/2) \approx 5.776074$

511. La estrategia empleada en §5.2 y §5.3 para resolver los sistemas de ecuaciones de primer orden que aparecen al transformar las ecuaciones de orden superior puede utilizarse para resolver sistemas de ecuaciones de primer orden lineales. Considérese el sistema

$$\begin{aligned} \frac{\mathrm{d}\,u}{\mathrm{d}\,t} &= -2u - 3v, & u(0) &= -1; \\ \frac{\mathrm{d}\,v}{\mathrm{d}\,t} &= -3u - 2v + 2\,\mathrm{e}^{2t}, & v(0) &= 1. \end{aligned}$$

para $t \in [0, 2]$. Obtener, mediante el método del Trapecio, el sistema de ecuaciones para calcular $u(t_i)$ y $v(t_i)$ a partir de $u(t_{i-1})$ y $v(t_{i-1})$ siendo $t_i = t_{i-1} + h$. Modificar el programa del ejercicio 506 ó 507 para calcular los valores $\{u_0, u_1, \dots\}$ y $\{v_0, v_1, \dots\}$ de las funciones $u(t)$ y $v(t)$. Tomar $h = 0.02$. La solución exacta es $u(t) = -\frac{1}{7}(\mathrm{e}^{-5t} + 6\,\mathrm{e}^{2t})$ y $v(t) = \frac{1}{7}(-\,\mathrm{e}^{-5t} + 8\,\mathrm{e}^{2t})$, comprobar los resultados. Dibujar las gráficas de $u(t)$ y $v(t)$.

Resultados: $u(2) \approx -46.810714$, $v(2) \approx 62.410767$

512. Resolver el sistema

$$\begin{aligned} \frac{\mathrm{d}\,u}{\mathrm{d}\,t} &= 3u - 2v - 2t\,\mathrm{e}^{-t}, & u(0) &= 1; \\ \frac{\mathrm{d}\,v}{\mathrm{d}\,t} &= 4u - v + t^3, & v(0) &= 0. \end{aligned}$$

para $t \in [0, 2]$.

Resultados: $u(2) \approx -14.223989$, $v(2) \approx -15.319331$

513. El siguiente programa corresponde a la integración mediante el método del Trapecio del ejemplo considerado en §5.4 (pág. 77). Completarlo, añadir los códigos de las funciones y subrutinas necesarias y escribirlo en un fichero. Compilarlo y corregir los errores. Ejecutarlo y comprobar los resultados. Dibujar las gráficas de las funciones $u(t)$, $v(t)$ e $y(t)$.

```
1 program ejercicio_513
2 implicit none
3 integer,parameter :: n=100
4 integer :: i
```

```
real*8 :: a,b,t(0:n),u(0:1,0:n),v(0:1,0:n),y(0:n),h,h2,
real*8 :: alfa,beta,gamma,f20,f21,f10,f11,g0,g1,z(2,2),c(2)
real*8,external :: f2,f1,g
external:: solve,factor
open(10,file='res_ejercicio_513.txt')
! asigna valores a los datos
a=0d0
b=1d0
alfa=2d0
beta=0d0
h=
h2=
! asigna valores iniciales
t(0)=a
u(:,0)=
v(:,0)=
f20=h2*f2(t(0))
f10=h2*f1(t(0))
g0=h2*g(t(0))
! comienza el bucle para avanzar la solución
do i=1,n
   t(i)=t(i-1)+h
   f21=
   f11=
   g1=
! calcula y factoriza la matriz
   z(1,1)=1d0
   
   
   =1d0+f21
   call factor(z,2)
! calcula el termino independiente del sistema para u
   c(1)=
   c(2)=-f10*u(0,i-1)+
!resuelve el sistema
   
!asigna valores a u
```

```
42      [          ]=c(1)
43      u(1,i)=[    ]
44  ! calcula el termino independiente del sistema para v
45      c(1)=v(0,i-1)+h2*v(1,i-1)
46      c(2)=-f10*v(0,i-1)+(1d0-f20)*v(1,i-1)
47  !resuelve el sistema
48      call solve(z,c,2)
49  !asigna valores a v
50      v(:,i)=(/c(1),c(2)/)
51  !guarda f21, f11 y g1 en f20, f10 y g0
52      [              ]
53      [              ]
54      [          ]
55  end do
56  ! calcula gamma y la solucion
57  gamma=(beta-u(0,n))/v(0,n)
58  do i=0,n
59      y(i)=u(0,i)+gamma*[          ]
60  end do
61  do i=0,n
62      write(10,'(f6.3,3(1x,f10.7))') t(i),u(0,i),v(0,i),y(i)
63  end do
64  end program ejercicio_513
```

514. Resolver el siguiente problema de contorno y dibujar la gráfica de la solución:

$$y'' + y = t \quad t \in [0, \pi/4]; \quad y(0) = 1, \quad y(\pi/4) = 1.$$

Calcular la solución exacta y comprobar los resultados.

Resultados: $y(\pi/8) \approx 1.050036$

515. El método del Disparo descrito en §5.4 puede modificarse para resolver otros tipos de condiciones de contorno. Desarrollar una versión del método que permita resolver el problema de contorno:

$$y'' + y = t \quad t \in [0, \pi/4]; \quad y'(0) = 1, \quad y(\pi/4) = 1.$$

Escribir un programa para obtener la solución y representarla gráficamente.

Resultados: $y(0) \approx 0.303492$, $y(\pi/8) \approx 0.673089$