Anuario del Centro de la Universidad Nacional de Educación a Distancia en Calatayud.
N.º 24, pp. 105-126, 2018

DOS EJEMPLOS DE APLICACIÓN DE SCILAB Y MAXIMA

Andrés MARTÍN SÁNCHEZ
UNED Calatayud
Profesor Tutor UNED Calatayud

Resumen: En este artículo se ilustra la aplicación de los programas de cálculo Scilab y Maxima en la resolución de dos ejemplos. Uno de ellos, consiste en el diseño de una estructura metálica de un invernadero. El otro, en la determinación del tiempo de vaciado de un depósito esférico.

Palabras clave: Maxima; Scilab; diseño estructura tubular; vaciado depósito.

Abstract In this article, we illustrate the application of the programs Scilab and Maxima in the resolution of two examples. One of them, consists in a methalic estructural design. The other, in the determination of how much time it takes for a spherical deposit to become empty.

Keywords: Maxima; Scilab; structural tubular design; emptiness of a deposit.

El presente trabajo recoge dos ejemplos de aplicación de Scilab y Máxima a la resolución de dos tipos de problemas.

El enunciado del primer problema corresponde a una de las Prueba de Evaluación a Distancia de la asignatura Herramientas Informáticas para las Matemáticas del Grado en Matemáticas, estando formado el Equipo Docente por D. Fernando Morilla García y D. Miguel Ángel Rubio González.

El enunciado del segundo corresponde a la Prueba de Evaluación a Distancia de la asignatura Resolución Numérica de Ecuaciones del Grado en Matemáticas, y el Equipo Docente está formado D. Carlos Moreno.

EJEMPLO 1. APARTADO 1

Un taller de cerrajería tiene el encargo de montar una estructura metálica para un invernadero. En el montaje está obligado a emplear tubos de cuatro longitudes (t1, t2, t3 y t4 en metros), no necesariamente diferentes, con tal de conseguir la forma que muestra la figura 1.

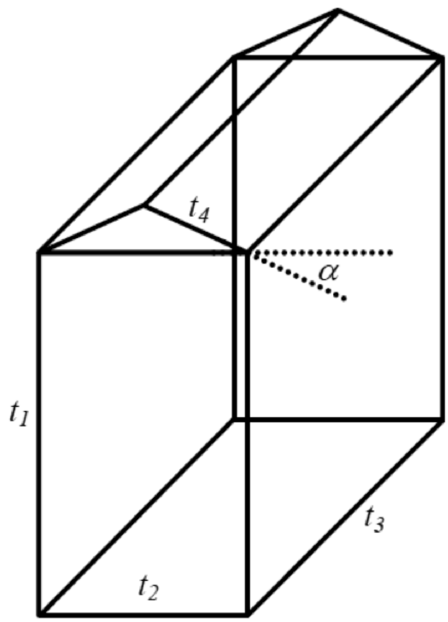

Figura 1 Estructura metálica para un invernadero.

Observe que el techo de la estructura debe formar un ángulo α en grados con la horizontal. Pero además, para ajustar bien su presupuesto, debería emplear una longitud total de tubos igual a L metros.

Documentar y programar una función tanto en Maxima como en Scilab que ayude al taller a decidir el tamaño de los cuatro tubos. En definitiva, se trata de determinar cuaternas (t_1, t_2, t_3, t_4) de longitudes de tubos compatibles con las especificaciones de diseño (estructural y longitud total de tubos). Para ello se propone que la función tenga por nombre "estructura tubular", que reciba como argumentos de entrada: t_1, t_2, L y α. Que genere a su salida las otras dos longitudes de tubo (t_3 y t_4) y el volumen (V) de aire que cabría en el invernadero. Y que, cuando la solución no sea posible devuelva un valor nulo para t_2 y para el volumen.

A continuación se muestra una tabla compuesta por seis escenarios recreados con esta función que les vendrán bien para depurarla. Los dos primeros escenarios muestran dos posibles soluciones para unas mismas especificaciones de diseño. El primer escenario sería el más favorable de los dos si se desea un invernadero con mayor volumen. Los tres siguientes escenarios son posibles soluciones con otras especificaciones de diseño. Y el sexto escenario recrea una situación imposible.

t_1	t_2	L	α	t_3	t_4	V
5	20	180	15º	7,72	10,35	978,58
5	5	180	15º	25,93	2,59	691,66
10	10	150	15º	9,86	5,18	1.051,93
10	10	150	20º	9,74	5,32	1.062,99
10	10	180	20º	15,74	5,32	1.717,58
10	25	180	20º	0,00	13,30	0,00

Tabla 1 Escenarios parámetros diseño estructura tubular

SOLUCIÓN

Cálculos a lápiz y papel

De la inspección de la Figura 1, es claro que las expresiones a calcular vienen dadas por las siguientes funciones:

$$L=4t_1+4t_2+5t_3+4t_4 \rightarrow 5t_3+4t_4 = L- 4(t_1+t_2)$$

$$t_4 *\cos\alpha =t_2/2 \rightarrow t_4 = t_2/(2 *\cos\alpha)$$

$$5t_3= L- 4(t_1+t_2 +t_4) =(1/5)*(L- 4*(t_1+t_2 + t_2/(2 *\cos\alpha)))$$

$$t_3=\frac{1}{5}.(L-4t_1-4t_2\left(1+\frac{1}{2.cos\propto}\right))$$

$$V=t_1.t_2.t_3+\frac{1}{2}t_2.t_3.t_4.sen\propto$$

$$V=t_3.t_2.\left[t_1+\frac{1}{2}.t_4.sen\propto\right]$$

$$V=\frac{1}{5}.\left(L-4t_1-4t_2\left(1+\frac{1}{2.cos\propto}\right)\right).t_2.\left[t_1+\frac{1}{4}\mathrm{t}_2\ .\mathrm{sen}\alpha.\cos\alpha\right]$$

Podemos comprobar lo anterior a través de una hoja de cálculo, en que coinciden los valores con los de la tabla del enunciado (salvo quizá algún valor negativo que se depurará correctamente en el código de Máxima)

t1	t2	L	α	cosα	senα	t3	t4	V
5	20	180	15	0,96592583	0,25881905	7,72	10,35	979
5	5	180	15	0,96592583	0,25881905	25,93	2,59	692
10	10	150	15	0,96592583	0,25881905	9,86	5,18	1052
10	10	150	20	0,93969262	0,34202014	9,74	5,32	1063
10	10	180	20	0,93969262	0,34202014	15,74	5,32	1718
10	25	180	20	0,93969262	0,34202014	-2,64	13,30	0

Tabla 2 Recreación en una hoja de cálculo escenarios estructura

Las fórmulas que introducimos para el cálculo de la salidas, corresponden con las calculadas arriba. Por ejemplo, abajo se detalla el cálculo de la columna del volumen

t1	t2	L	α	cosα	senα	t3	t4	V
5	20	180	15	0,96592583	0	=SI(H3*C3*(B3+0,5*I3*G3)>0;H3*C3*(B3+0,5*I3*G3);0)		
5	5	180	15	0,96592583	0,258	SI(**prueba_lógica**; [valor_si_verdadero]; [valor_si_falso])		

Figura 2 Comprobación en una hoja de cálculo de datos tabulares del ejemplo 1.

Cálculos con Máxima

El entorno de Máxima permite la edición de comentarios y organización en el propio código, de modo que presentamos directamente éste autocomentado.

```
(c)Andrés Martín

1 Revista Anales 2018- Ejemplo 1

1.1 Definición de los argumentos de salida

Definimos t4,t3 y V en función de los argumentos de entrada t1,t2,L y alfa

 --> t4(t2,alfa):=t2/(2*cos(alfa));
```

(%o9) $t4(t2,alfa):=\frac{t2}{2\cos(alfa)}$

```
 --> t3(t1,t2,L,alfa):=if 1/5*(L-4*t1-4*t2*(1+1/(2*cos(alfa))))>0
     then 1/5*(L-4*t1-4*t2*(1+1/(2*cos(alfa)))) else 0;
```

(%o10) $t3(t1,t2,L,alfa):=\text{if}\ \frac{1}{5}\left(L-4\,t1+(-4)\,t2\left(1+\frac{1}{2\cos(alfa)}\right)\right)>0\ \text{then}\ \frac{1}{5}\left(L-4\,t1+(-4)\,t2\left(1+\frac{1}{2\cos(alfa)}\right)\right)\ \text{else}\ 0$

```
 --> V(t1,t2,L,alfa):=if 1/5*(L-4*t1-4*t2*(1+1/(2*cos(alfa))))*t2*(t1+1/4*t2*tan(alfa))>0
     then 1/5*(L-4*t1-4*t2*(1+1/(2*cos(alfa))))*t2*(t1+1/4*t2*tan(alfa))else 0;
```

(%o12) $V(t1,t2,L,alfa):=\text{if}\ \frac{1}{5}\left(L-4\,t1+(-4)\,t2\left(1+\frac{1}{2\cos(alfa)}\right)\right)t2\left(t1+\frac{1}{4}\,t2\tan(alfa)\right)>0\ \text{then}\ \frac{1}{5}\left(L-4\,t1+(-4)\,t2\left(1+\frac{1}{2\cos(alfa)}\right)\right)t2\left(t1+\frac{1}{4}\,t2\tan(alfa)\right)\text{else}\ 0$

Figura 3 Argumentos de salida en código de Máxima (ejemplo 1)

```
1.2 Definición de la estructura tubular

En Máxima no es posible definir una función con varios argumentos de salida.
Por lo que para resolver lo que pide el apartado, definimos una estructura,
que llamamos Estructura tubular, cuyos campos sean precisamente los argumentos
de salida, t4, t3 y V de la función que nos piden.

 --> defstruct(Estructura_tubular(t3,t4,V));
(%o4) [Estructura_tubular(t3,t4,V)]

1.3 Entradas

Definimos una matriz con los datos de los argumentos de entrada para los seis
escenarios de depuración de la función...

 --> Entradas:matrix([Escenario, t1,t2,L,alfa],[Uno,5,20,180,15],[Dos,5,5,180,15],
     [Tres,10,10,150,15],[Cuatro,10,10,150,20],[Cinco,10,10,180,20],[Seis,10,25,180,20]);
```

(%o5) $\begin{bmatrix} Escenario & t1 & t2 & L & alfa \\ Uno & 5 & 20 & 180 & 15 \\ Dos & 5 & 5 & 180 & 15 \\ Tres & 10 & 10 & 150 & 15 \\ Cuatro & 10 & 10 & 150 & 20 \\ Cinco & 10 & 10 & 180 & 20 \\ Seis & 10 & 25 & 180 & 20 \end{bmatrix}$

Figura 4 Definición de la estructura tubular y entradas en Máxima (ejemplo 1)

1.4 Salidas a través de disp

Un modo de calcular y presentar los valores de t3, t4 y V, es agregar los registros correspondientes a la estructura definida y presentar los campos a través de la función disp.

```
(%i6) for i:1 thru 6 do
      (Salidas:new(Estructura_tubular(float(t3(Entradas[i+1,2],Entradas[i+1,3],
      Entradas[i+1,4],Entradas[i+1,5]*%pi/180)),
      float(t4(Entradas[i+1,3],Entradas[i+1,5]*%pi/180)),
      float(V(Entradas[i+1,2],Entradas[i+1,3],Entradas[i+1,4],
      Entradas[i+1,5]*%pi/180)))),
      disp("Los valores t3,t4 y V son para el escenario"),disp(i),
      disp(Salidas@t3),
      disp(Salidas@t4),
      disp(Salidas@V));
```

Los valores t3,t4 y V son para el escenario
1
7.717790556719336
10.35276180410083
978.5766303744828
Los valores t3,t4 y V son para el escenario
2
25.92944763917984
2.588190451025207
691.6597819488917
Los valores t3,t4 y V son para el escenario
3
9.858895278359668
5.176380902050415
Los valores t3,t4 y V son para el escenario
4
9.743288910096354
5.32088886237956
1062.985569687912
Los valores t3,t4 y V son para el escenario
5
15.74328891009635
5.32088886237956
1717.581104827842
Los valores t3,t4 y V son para el escenario
6
0.0
13.3022221559489
0.0
(%o6) *done*

... donde se comprueba que los valores coinciden con la matriz de depuración del enunciado.

Figura 5 Salidas para el diseño de la estructura tubular en Máxima (ejemplo 1)

1.5 Salidas a través de una matriz

Otra forma de presentar las salidas es a través de una matriz.
Para ell, primeramente damos forma a la matriz de salidas que iremos completando con los argumentos de salida.

```
(%i7) Matrizsalidas:zeromatrix(7,3);
```

$$(\%o7)\quad \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Figura 6 Preparación salida matricial para el diseño la estructura en Maxima (ejemplo 1)

Rellenamos los encabezados de la matriz con los nombres de los argumentos de salida...

```
(%i8) Matrizsalidas[1,1]:t3;
      Matrizsalidas[1,2]:t4;
      Matrizsalidas[1,3]:V;
(%o8) t3
(%o9) t4
(%o10) V
```

```
(%i11) Matrizsalidas;
```

(%o11)

t3	t4	V
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

Estos argumentos de salida los vamos calculando para cada uno de los escenarios y argumentos de entrada de cada escenario, a través de un bucle, y el acceso a los campos de la estructur a través de ...

```
(%i17) for i:1 thru 6 do
       (Salidas:new(Estructura_tubular(float(t3(Entradas[i+1,2],Entradas[i+1,3],Entradas[i+1,
       Entradas[i+1,5]*%pi/180)),
       float(t4(Entradas[i+1,3],Entradas[i+1,5]*%pi/180)),
       float(V(Entradas[i+1,2],Entradas[i+1,3],Entradas[i+1,4],Entradas[i+1,5]*%pi/180)))),
       Matrizsalidas[i+1,1]:Salidas@t3,
       Matrizsalidas[i+1,2]:Salidas@t4,
       Matrizsalidas[i+1,3]:Salidas@V);
(%o17) done
```

Vemos ahora los contenidos de la matriz de salidas...

```
(%i13) Matrizsalidas;
```

(%o13)

t3	t4	V
7.717790556719336	10.35276180410083	978.5766303744828
25.92944763917984	2.588190451025207	691.6597819488917
9.858895278359668	5.176380902050415	1051.931603538454
9.743288910096354	5.32088886237956	1062.985569687912
15.74328891009635	5.32088886237956	1717.581104827842
0.0	13.3022221559489	0.0

La matriz depuración la creamos uniendo las matrices de los argumentos de entrada y los escenarios y la matriz de los argumentos de salida.

```
(%i14) MatrizDepuracion:addcol(Entradas,Matrizsalidas);
```

(%o14)

Escenario	t1	t2	L	alfa	t3	t4	V
Uno	5	20	180	15	7.717790556719336	10.35276180410083	978.5766303744828
Dos	5	5	180	15	25.92944763917984	2.588190451025207	691.6597819488917
Tres	10	10	150	15	9.858895278359668	5.176380902050415	1051.931603538454
Cuatro	10	10	150	20	9.743288910096354	5.32088886237956	1062.985569687912
Cinco	10	10	180	20	15.74328891009635	5.32088886237956	1717.581104827842
Seis	10	25	180	20	0.0	13.3022221559489	0.0

Comprobamos que la matriz y su contenido coincide con los valores de la tabla del enunciado. ¡La depuración es correcta!

Figura 7 Salida matricial para el diseño de la estructura tubular en Máxima (ejemplo 1)

Cálculos con Scilab

Aquí presentamos el código comentado e insertamos las salidas de la consola Scilab por trozos relevantes del mismo.

```
// Revista Anales 2018

// Ejemplo 1

clc
funcprot(0)

// Definición de la función EstructuraTubular
// Tiene como entradas t1,t2, L y alfa y
// devuelve como salidas t3,t4 y V
function [t3,t4,V]=EstructuraTubular(t1,t2,L,alfa)
    t4=t2/(2*cos(alfa))
    t3=1/5*(L-4*t1+(-4)*t2*(1+1/(2*cos(alfa))))
    V=1/5*(L-4*t1-4*t2*(1+1/(2*cos(alfa))))*t2*(t1+1/4*t2*tan(alfa))
    // Aquí se da valor 0 a t3 y t4 si son negativos
    if t3>0 then t3
        else t3=0
    end
    if V>0 then V
        else V=0
    end
endfunction
```

Figura 8 Definición estructura tubular en Scilab (ejemplo 1)

```
// En esta parte cargamos y presentamos las entradas de la matriz de depuración
EncabezadoEntradas=["  t1 ",'  t2  ','   L   ','  alfa ']
Entradas=[5, 20, 180, 15;5 5 180 15;10 10 150 15;10 10 150 20;10 10 180 20;10 25 180 20]
// Alternativamente podemos alimentar las entradas de un fichero externo.
// Para ello, hay que guardar el fichero de texto t2.txt en el directorio de trabajo...
// y transformar en código ejecutable las tres siguientes líneas eliminando las barras de comentario
//a1=file('open','t2.txt','old')
//Entradas=read(a1,6,4)
//file('close',a1)
disp("ACTIVIDAD 2.1")
disp(" ")
disp('Las entradas que utilizaremos para la depuración son')
disp(EncabezadoEntradas)
disp(Entradas)

// En esta parte calculamos y presentamos las salidas..
EncabezadoSalidas=["  t3      ",'     t4   ','   V   ']
Salidas=zeros(6,3)
// A través de un bucle, vamos asignando a la matriz creada,
// los valores t3, t4 y V para las entradas de cada una de los escenarios
for i = 1:6
    [t3,t4,V]=EstructuraTubular(Entradas(i,1),Entradas(i,2),Entradas(i,3),Entradas(i,4)*%pi/180)
        Salidas(i,1)=t3
        Salidas(i,2)=t4
        Salidas(i,3)=V
end
// Mostramos la matriz depurada y comprobamos que ooinciden con la plantilla del enunciado
disp('...que corresponden con las salidas')
disp(EncabezadoSalidas)
disp(Salidas)
disp("")
```

Figura 9 Código salidas depuración estructura tubular en Scilab (ejemplo 1)

```
Las entradas que utilizaremos para la depuración son

  t1     t2      L       alfa   !

   5.     20.    180.    15.
   5.     5.     180.    15.
   10.    10.    150.    15.
   10.    10.    150.    20.
   10.    10.    180.    20.
   10.    25.    180.    20.

...que corresponden con las salidas

  t3             t4        V      !

   7.7177906    10.352762    978.57663
   25.929448    2.5881905    691.65978
   9.8588953    5.1763809    1051.9316
   9.7432889    5.3208889    1062.9856
   15.743289    5.3208889    1717.5811
   0.           13.302222    0.
```

Figura 10 Salidas depuración estructura tubular en Scilab (ejemplo 1)

Como comentario fuera de código en este punto, destacamos que no ha sido posible en Scilab, la concatenación de matrices que sin embargo si se ha conseguido en Maxima.

Ni siquiera se ha podido encontrar una manera de hacer coexistir en la misma matriz datos de distinto tipo, que sin embargo si hemos conseguido en Máxima (la matriz de entradas por ejemplo, contenía datos de tipo string-escenarios, t1,t2,..- y numérico-los valores de los parámetros para cada escenario).

```
//Aquí pedimos la interacción del usuario para que
// él mismo proporcione las entradas y compruebe
// las salidas
t1=input("Escribe el valor de t1  ")
t2=input("Escribe el valor de t2 ")
L=input("Escribe el valor L ")
alfa=input("Escribe el valor de alfa ")
[t4,t3,V]=EstructuraTubular(t1,t2,L,alfa*%pi/180)
disp("Los valores de t4 t3 y V para dichos valores son")
disp([t4 t3 V])
```

Figura 11 Código salidas estructura tubular en Scilab (ejemplo 1)

```
Escribe el valor de t1  10
Escribe el valor de t2 25
Escribe el valor L 180
Escribe el valor de alfa 20

 Los valores de t4 t3 y V para dichos valores son

    0.    13.302222    0.
```

Figura 12 Salidas estructura tubular en Scilab (ejemplo 1)

La interacción con el usuario tal como se ha presentado en Scilab, se podía haber hecho igualmente en Maxima, a través de argumentos similares a los presentados arriba.

EJEMPLO 1. APARTADO 2

Definitivamente se ha decidido emplear 200 metros de tubo y que el techo de invernadero forme un ángulo de 15ª con la horizontal, ¿qué cuaterna (t_1,t_2,t_3,t_4) de longitudes de tubos debería utilizar el taller para que el volumen del invernadero sea máximo? Se pide resolver este problema mediante optimización en Scilab. En su planteamiento se le sugiere reutilizar parte de la función programada en el apartado anterior y hacerlo lo suficientemente general como para que el código presentado se pueda probar en otras condiciones (L y α) de diseño. Por ejemplo, se sabe que la cuaterna óptima con 150 m de tubo y techo a 20º es (11,71, 8.6, 10.00, 4.62).

Cálculos a lápiz y papel (Maxima)

El problema consiste en optimizar el volumen de la estructura, dadas unas restricciones no lineales, por lo que es posible que no podamos dar con una solución exacta del problema, y por ello se pide una resolución mediante cálculo numérico.

Aunque en el apartado no piden cálculos con Máxima (por la razón apuntada en el anterior párrafo), nos ayudamos de ella, sin embargo, para calcular las derivadas que utilizaremos posteriormente en los cálculos numéricos de Scilab….

```
Nos sirve para calcular los gradientes de la función que utilizamos en Scilab

 --> diff(1/5*(L-4*t1-4*t2*(1+1/(2*cos(alfa))))*t2*(t1+1/4*t2*tan(alfa)),t1,1);
```

$$(\%o24)\quad \frac{t2\left(L-4\left(\frac{1}{2\cos(alfa)}+1\right)t2-4\,t1\right)}{5}-\frac{4\,t2\left(\frac{\tan(alfa)\,t2}{4}+t1\right)}{5}$$

```
 --> diff(1/5*(L-4*t1-4*t2*(1+1/(2*cos(alfa))))*t2*(t1+1/4*t2*tan(alfa)),t2,1);
```

$$(\%o25)\quad \frac{\left(\frac{\tan(alfa)\,t2}{4}+t1\right)\left(L-4\left(\frac{1}{2\cos(alfa)}+1\right)t2-4\,t1\right)}{5}+\frac{\tan(alfa)\,t2\left(L-4\left(\frac{1}{2\cos(alfa)}+1\right)t2-4\,t1\right)}{20}-\frac{4\left(\frac{1}{2\cos(alfa)}+1\right)t2\left(\frac{\tan(alfa)\,t2}{4}+t1\right)}{5}$$

Figura 13 Cálculos Maxima optimización volumen (ejemplo1)

Cálculos con Scilab

Resolvemos este apartado en Scilab a través de dos vías alternativas. Una es utilizando la función *optim* y otra la función *fminsearch*, con similares resultados numéricos, que coinciden con los valores de comprobación que ofrece el enunciado.

```
disp(" ")
funcprot(0)
// Primer método. Función optim
disp("Función optim")
// El primer modo de resolver el apartado 2 es a través de la función optim
// Para ello definimos la función Volumen en que reutilizamos como piden parte de
// la función del apartado anterior (en concreto t4, t3 y V)
function [V,dV,ind]=Volumen(x,ind,L,alfa)
    // El hacer depender la función de los parámetros genéricos L y alfa, y no
    // de valores concretos, pretende cumplir con la especificación del enunciado
    // de definir lo suficientemente general para probar otras condiciones (L y alfa)
    // de diseño
    t1=x(1);t2=x(2);
    t4=t2/(2*cos(alfa));
    t3=1/5*(L-4*t1-4*t2*(1+1/(2*cos(alfa))));
    //V=1/5*(L-4*t1-4*t2*(1+1/(2*cos(alfa))))*t2*(t1+1/4*t2*tan(alfa))
    V=-t3*t2*(t1+1/2*t4*sin(alfa))
    dV(1)=-t2/5*((L-4*t1-4*t2*(1+1/(2*cos(alfa))))-4*(t1+1/4*t2*tan(alfa)))
    dV(2)=-1/5*((-4)*(1+1/(2*cos(alfa)))*(t2*t1+1/4*(t2)^2*tan(alfa))+(t1+t2/2*tan(alfa))
endfunction
```

+(t1+t2/2*tan(alfa))*(L-4*t1-4*t2*(1+1/(2*cos(alfa)))))

```

// Condiciones L=150 y alfa=20
L=150;
alfa=%pi/9;
[V,x,dV]=optim(Volumen,[25;25],'gc');
t1=x(1)
t2=x(2)
t4=t2/(2*cos(alfa))
t3=1/5*(L-4*t1+(-4)*t2*(1+1/(2*cos(alfa))))
disp('La cuaterna óptima para L=150 m y alfa=20ª corresponde a los valores t1, t2, t3 y t4 de')
disp([t1 t2 t3 t4])

// Condiciones L=200 y alfa=15
L=200;
alfa=%pi/6;
[V,x,dV]=optim(Volumen,[25;25],'gc');
t1=x(1)
t2=x(2)
t4=t2/(2*cos(alfa))
t3=1/5*(L-4*t1+(-4)*t2*(1+1/(2*cos(alfa))))
disp('La cuaterna óptima para L=200 m y alfa=15ª corresponde a los valores t1, t2, t3 y t4 de')
disp([t1 t2 t3 t4])

```

Figura 14 Cálculos Scilab optimización volumen con función *optim* (ejemplo1)

… lo cual produce la siguiente salida…

```
Función optim

La cuaterna óptima para L=150 m y alfa=20ª corresponde a los valores t1, t2, t3 y t4 de

   11.710735    8.6739517    10.    4.6153133
```

```
La cuaterna óptima para L=200 m y alfa=15ª corresponde a los valores t1, t2, t3 y t4 de

   14.987947    11.630509    13.333333    6.7148773
```

Figura 15 Resultados Scilab optimización volumen con función *optim* (ejemplo1)

El primero de los escenarios que se ofrece (L=150 m y alfa=20º) coincide con los resultados numéricos del enunciado, confirmando que nuestros cálculos son correctos.

Como se verá en el siguiente apartado, el escenario segundo (L=200 y alfa=15º) también coincide con los resultados gráficos de Máxima, reafirmando la bondad del código diseñado.

Hay que observar, que tanto la función *optim*, definida arriba como en la función *fminsearch* que se ofrece a continuación, son programas de minimización, por lo que hemos dado un signo negativo a la función volumen, para convertir este algoritmo de minimización en la maximización que nos pide el enunciado.

La función *fminsearch* (como también la función *optim)* la hemos definido adaptando el ejemplo de los apuntes de la asignatura al problema del enunciado.

```
// Segundo método. Función fminsearch
disp("Función fminsearch")
funcprot(0)
function V=Volumen(x,ind,L,alfa)
    t1=x(1);t2=x(2);
    t4=t2/(2*cos(alfa));
    t3=1/5*(L-4*t1-4*t2*(1+1/(2*cos(alfa))));
    //V=1/5*(L-4*t1-4*t2*(1+1/(2*cos(alfa))))*t2*(t1+1/4*t2*tan(alfa))
    V=-t3*t2*(t1+1/2*t4*sin(alfa))
endfunction

// Condiciones L=150 y alfa=15
opt=optimset("TolX",0.001);
L=150;
alfa=%pi/9;
xo=[10;10]
[x,fval,exitflag,output]=fminsearch(Volumen,xo,opt)
t1=x(1)
t2=x(2)
t4=t2/(2*cos(alfa))
t3=1/5*(L-4*t1+(-4)*t2*(1+1/(2*cos(alfa))))
disp('La cuaterna óptima para L=150 m y alfa=20ª corresponde a los valores t1, t2, t3 y t4 de')
disp([t1 t2 t3 t4])

```

Figura 16 Definición Scilab función *fminsearch* y primeras condiciones (ejemplo1)

```
// Condiciones L=200 y alfa=20
L=200;
alfa=%pi/9;
xo=[10;10]
[x,fval,exitflag,output]=fminsearch(Volumen,xo,opt)
t1=x(1)
t2=x(2)
t4=t2/(2*cos(alfa))
t3=1/5*(L-4*t1+(-4)*t2*(1+1/(2*cos(alfa))))
disp('La cuaterna óptima para L=200 m y alfa=15ª corresponde a los valores t1, t2, t3 y t4 de')
disp([t1 t2 t3 t4])
```

Figura 17 Segundas condiciones Scilab con función fminsearch (ejemplo1) con las salidas…

```
Función fminsearch

La cuaterna óptima para L=150 m y alfa=20ª corresponde a los valores t1, t2, t3 y t4 de

   11.710964    8.6739211    9.9998545    4.615297

La cuaterna óptima para L=200 m y alfa=15ª corresponde a los valores t1, t2, t3 y t4 de

   15.614159    11.565168    13.333581    6.1536975
```

Figura 18 Salidas Scilab *fminsearch* (ejemplo1)

Los resultados son muy similares a los obtenidos con la otra función con una diferencia que no excede de milésimas en la mayor parte de los datos, y a lo sumo de algunas décimas en algún dato (t1=14.98 con *optim* vs. t1=15.61 con *fminsearch*)

EJEMPLO 1. APARTADO 3

Comprobar de forma gráfica en Máxima que la solución encontrada en el apartado 2 corresponde a un máximo absoluto y no a un máximo local. Se le sugiere utilizar una representación tridimensional del volumen (V) frente a las longitudes (t1,t2) en el rango de 1 a 30 m.

El enunciado pide la corroboración gráfica con Maxima de los resultados obtenidos en el punto 2 de Scilab. Para ello nos servimos de dos representaciones tridimensionales y una curva de nivel.

En cada una de ellas, se confirma que los valores máximos corresponden a un t115 y un t211, lo cual confirma los resultados del apartado 2, obtenidos con una mayor precisión.

Por otra parte, aunque ya se confirmó comparando la matriz de depuración, el gráfico muestra como la función volumen está bien definida, pues se omiten los valores negativos.

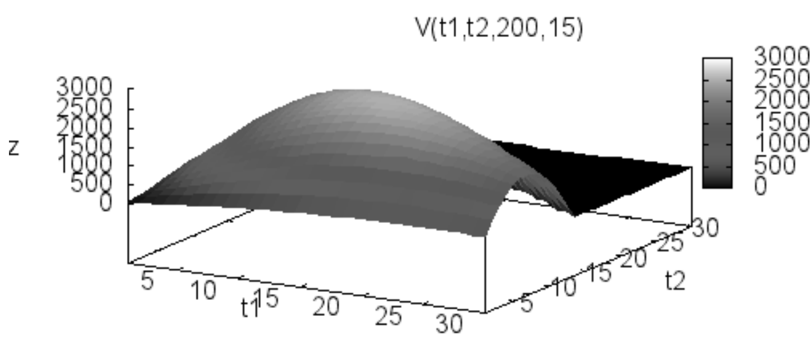

Gráfico 1 Volumen estructura tubular con gráficos Maxima

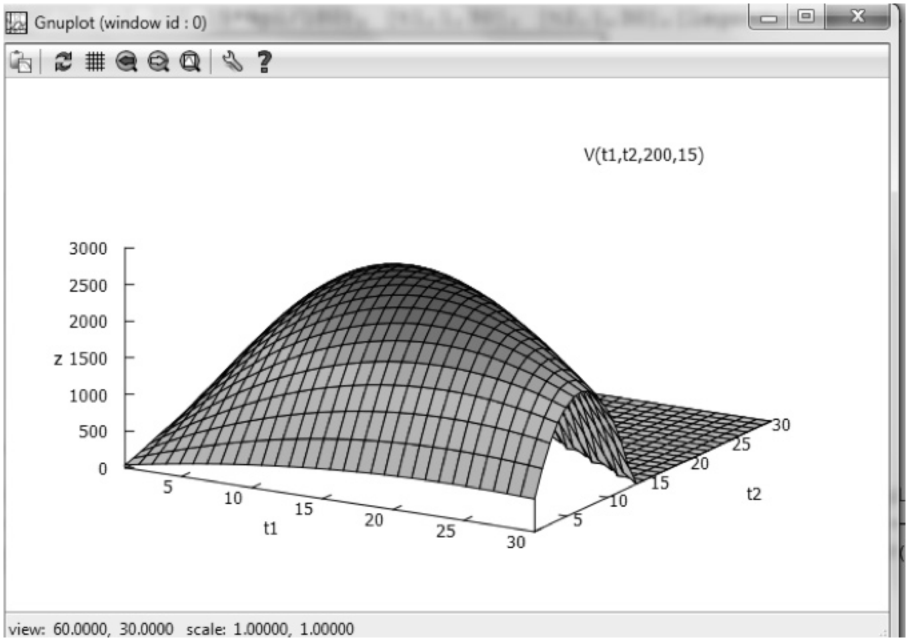

Gráfico 2 Volumen estructura tubular con Maxima (ejemplo 1)

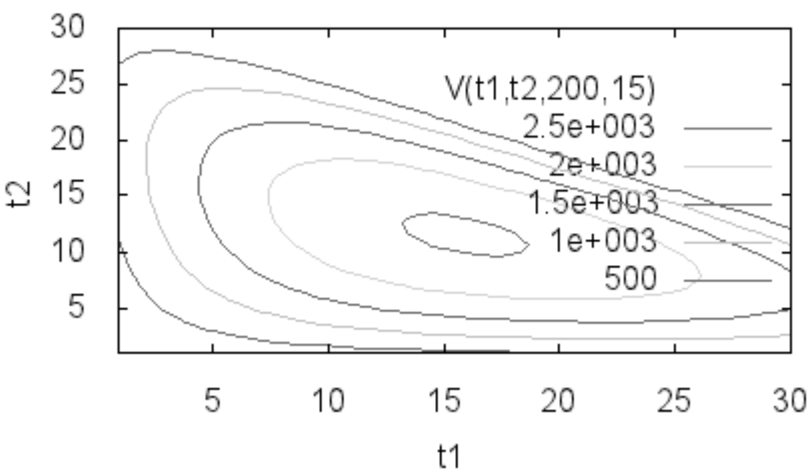

Gráfico 3 Curva nivel del volumen de estructura tubular con Maxima (ejemplo 1)

En el apartado 1, se pide definir la función de la geometría de la estructura y la comprobación de que esta definición es correcta, por comparación con una plantilla de depuración. Tanto en Maxima como en Scilab, se ha definido satisfactoriamente este función que tiene la forma…

$$(t4,t3,V)=\text{EstructuraTubular}(t1,t2,L,\alpha)$$

Mientras que en Máxima, hemos recurrido al auxilio de una estructura, para almacenar las variables de salida; en Scilab, las funciones soportan varias variables de salida, por lo que esto no ha sido necesario.

La presentación de resultados se ha hecho en ambos entornos a través de una matriz, y en Maxima adicionalmente presentando los resultados línea a línea. Aquí, Maxima ha ofrecido la ventaja frente a Scilab de una mayor flexibilidad en la concatenación de matrices, por lo que el resultado ofrecido es más compacto.

En el apartado 2, se utilizan las funciones *fminsearch* y *optim* de cálculo numérico de Scilab para obtener los valores t1 y t2 que maximizan el volumen para unos L y alfa genéricos. Para ello, reaprovechamos parte de las funciones (V,t3 y t4) definidas en el apartado 1, tal como se pide. En concreto, para L=200 y alfa=15 estos valores son aproximadamente t1=15 m y t2=11m. El hecho de que para L=180 y alfa=20º, los valores de t1 y t2 coincidan con la solución del enunciado, confirma que el apartado está correctamente resuelto.

En el apartado 3, se muestra el potencial gráfico de Máximo, mostrando la función Volumen calculada en el apartado 1 en gráfico tridimensional, con dos representaciones alternativas y una curva de nivel. Al realizar la representación gráfica para L=200 y alfa=20º, se observa que el máximo, se alcanza en torno a (t1,-t2)=(15m,11m), confirmando los resultados obtenidos en el apartado 2 con Scilab.

EJEMPLO 2: TIEMPO DE VACIADO DE UN DEPÓSITO

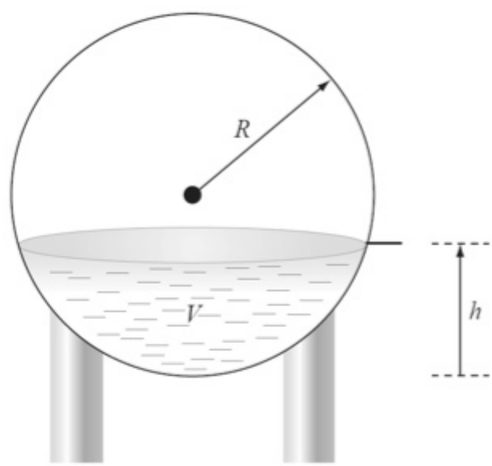

Figura 19 Depósito cilíndrico

Un tanque esférico de 3 m de radio está destinado a almacenar un determinado líquido. El tanque contiene 40 metros cúbicos de líquido. Se desea conocer la altura h que alcanza el líquido, medida respecto al plano tangente al tanque en su punto más bajo (polo sur).

El tanque tiene un orificio circular en su punto más bajo de 3 cm de diámetro por donde comienza a fluir el líquido con la siguiente ley

$$\frac{dV}{dt} = -CA\sqrt{2gh}$$

donde

V(t) representa el volumen de líquido en m^3 que contiene el tanque en el instante t

h(t) representa la altura que alcanza el líquido en el instante t

C es una constante empírica adimensional asociada al líquido. En este caso se supone que C=0.55

g es la constante gravitatoria cuyo valor es g=9,8 m/s^2

A es el área de la sección del orificio.

El volumen del casquete suprimido al colocar el orificio se considera poco significativo. Determinar la evolución de la altura del líquido al lo largo del tiempo.

SOLUCION

Empezamos con una interpretación del enunciado, en que destacamos la ley de variación temporal del volumen del líquido como

$$\frac{dV}{dt} = -CA\sqrt{2gh}$$

Parece lógico que dicha variación dependa linealmente de la sección del orificio de escape y de las características del líquido, así como de la raíz de la gravitación y la altura (cuanto mayor sea la sección del orificio o la altura, mayor será la variación del volumen).

El que la ley de variación sea negativa tiene sentido, pues a medida que transcurre el tiempo, el depósito se va vaciando (el volumen decrece).

Comenzamos pues la resolución. Por geometría elemental,

$$V = \pi\left(h^2.R - \frac{h^3}{3}\right)$$

Aplicando la regla de la cadena

$$\frac{dV}{dt} = \frac{d}{dt}\left[\pi\left(h^2.R - \frac{h^3}{3}\right)\right] = \pi\left(2Rh.\frac{dh}{dt} - \frac{1}{3}.3h^2.\frac{dh}{dt}\right)$$

Igualando con el segundo miembro de la ley de variación del volumen y despejando

$$\frac{dh}{dt} = -\frac{CA}{\pi}.\frac{\sqrt{2gh}}{2hR - h^2}$$

Según se indica en el enunciado V(0)=40, es decir

$$40 = \pi\left(h(0)^2.R - \frac{h(0)^3}{3}\right)$$

$$\frac{\pi}{3}.h(0)^3 - 3\pi.h(0)^2 + 40 = 0$$

en donde hemos sustituido R por 3 m (condiciones del enunciado).

Se trata de una ecuación cúbica, que tiene una solución exacta pero nada manejable por la teoría de radicales, y que en nuestro caso aproximamos por el método de la bisección, cuyo algoritmo se ha implementado en Scilab (ver código correspondiente) y cuyo valor se ha comparado con el valor dado por la orden roots de Scilab y Geogebra.

A continuación, se muestran las salidas de aplicación del algoritmo de bisección en Scilab y la orden root

```
// Revista Anales 2018
// Ejemplo 2

funcprot(0)
clc

// 1. DETERMINACIÓN h(0)

// Se determina h(0), es decir las condiciones iniciales del problema de valor
//inicial que queremos resolver por dos vías:

// Método 1-Bisección (capítulo 7)

// Método 1- Definición del algoritmo
// El algoritmo de la bisección es una transcripción del código
// para el método de bisección de la asignatura HIM
// con una mejora para presentación de resultados en forma
// matricial, según código de Guillermo García

```

Figura 20 Presentación función *biseccion* (ejemplo 2)

```

function x2 = biseccion(f,a,b,niteraciones,precision)
    n=0,x0=a,x1=b
    while n<niteraciones & abs(x1-x0)>precision
        n=n+1
        x2=(x0+x1)/2
        tabla(n,1)=n
        tabla(n,2)=x0
        tabla(n,3)=x1
        tabla(n,4)=(b-a)/(2^n)
              x=x0
        f0=evstr(f)
        x=x2
        f2=evstr(f)
        if f2==0 then
            disp("Tras"), disp(n),disp("iteraciones"),disp("la solución exacta es")
            disp(x2)
            return
        elseif f2*f0<0 then
```

```
            x1=x2
        else
            x0=x2
        end

    end
```

Figura 21 Definición función *biseccion* (ejemplo 2)

```
        disp("ALGORITMO BISECCIÓN")
        if n==niteraciones then
            disp("No alcanzamos la solución con la precisión deseada, la aproximación es")
        else
            disp("Tras"), disp(n),disp("iteraciones"),
            disp("la solución encontrad  con precision"),
            disp(precision),disp(" es"),
        end
        disp(x2);
        disp('Iteración     ak      bk        Error')
        disp(tabla);
endfunction
// Método 1- Se utiliza el algoritmo de la bisección para calcular las raíces
//de la ecuación cúbica que resulta de igualar el volumen inicial del tanque
//(40) con el volumen del tanque en función de la altura.
biseccion('3.14*3*x^2-3.14*x^3/3-40',0,6,50,0.01);

// Método 2- Se determinan las raíces a través de la orden roots de Scilab
p=poly([-40,0,%pi*3,-%pi/3],"x","coeff");
roots(p)
disp("ROOTS")
disp("Las raices de la ecuacion -40+%pi*3*x^2-%pi/3*x^3 con la orden roots son")
disp(roots(p))
```

Figura 22 Aplicación función *bisección* y la orden *roots* (ejemplo 2)

En las Figuras 23 y 24 se ilustran las salidas de las funciones *biseccion* y de la orden *roots* definidas anteriormente.

```
ALGORITMO BISECCIÓN
Tras
   10.
iteraciones
la solución encontrad  con precision
   0.01
 es
   2.4082031

Iteración     ak          bk          Error
   1.     0.           6.           3.
   2.     0.           3.           1.5
   3.     1.5          3.           0.75
   4.     2.25         3.           0.375
   5.     2.25         2.625        0.1875
   6.     2.25         2.4375       0.09375
   7.     2.34375      2.4375       0.046875
   8.     2.390625     2.4375       0.0234375
   9.     2.390625     2.4140625    0.0117188
   10.    2.4023438    2.4140625    0.0058594
```

Figura 23 Salidas función *biseccion* (ejemplo 2)

```
ROOTS

Las raices de la ecuacion -40+%pi*3*x^2-%pi/3*x^3 con la orden roots son

   8.4672175
   2.4069868
 - 1.8742044
```

Figura 24 Salidas de la orden roots (ejemplo 2)

En el gráfico 4, se detalla la obtención de estas raíces utilizando Geogebra.

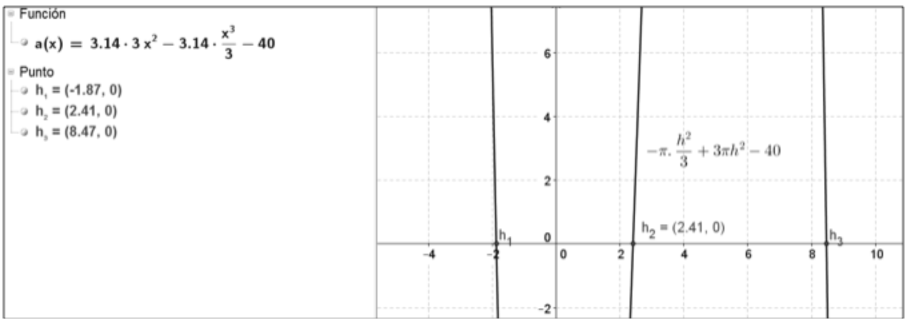

Gráfico 4 *Determinación con Geogebra de h(0)*

En la siguiente tabla se comparan las tres aproximaciones

Método	Bisección(Scilab)	Roots(Scilab)	Geogebra
h (0)	2.4082031	2.4069868	2.41

Tabla 2*- Comparativa resultados obtención h(0)*

Tomamos por simplificar h(0)=2,41, con lo que nuestro problema se puede plantear como

$$\frac{dh}{dt} = -\frac{CA}{\pi} . \frac{\sqrt{2gh}}{2hR - h^2}$$

con f(t,h(t)):D=[0,t(h=0)]x[2,41,2R)→R

Aquí vamos a utilizar el método de Euler con algoritmo implantado en Scilab (ver código), que muestra la siguiente salida en su ejecución, es decir, la evolución temporal de la altura del líquido en el interior del tanque, que es lo que pide el enunciado de la PEC.

```

// 2.RESOLUCION DEL PROBLEMA DE VALOR INICIAL

// Se resuelve el problema de valor inicial dh/dt=f(t,h(t)) con h(0) el valor
// obtenido anteriormente por dos métodos:
// Método 1- Se utiliza el método de Euler
// Para ello, definimos previamente el algoritmo tal como se muestra a continuación.
// Este algoritmo se ha adaptado de un video de Internet modificando ligeramente
//la notación (x por t), y favoreciendo la presentación de resultados a través de una tabla

```

```
function [t,y] = euler(f,t0,y0,h,n)
    t(1)=t0
    y(1)=y0
        for i=1:n
        t(i+1)=t(i)+h;
        y(i+1)=y(i)+h*f(t(i),y(i))
        tabla(i,1)=t(i+1)
        tabla(i,3)=y(i+1)
        tabla(i,2)=t(i+1)/60
    end
disp("EVOLUCIÓN TEMPORAL ALTURA DEL TANQUE")
disp("  t(seg)    t(min)      h(t)")
disp(tabla)
endfunction

// Aqui definimos f(t,y), tal como ha resultado de la manipulación de la ley de
// variación del volumen del enunciada
function h=f(t,y)
    C=0.55;A=%pi*0.03^2;g=9.8;R=3
    h=-C*A/%pi*sqrt(2*g*y)/(2*y*R-y^2)
endfunction

// Finalmente aplicamos el algoritmo de Euler a la h(t)=f(t,h(t)) anteriormente definida.
// Se han escogido pasos de 60 en 60 segundos, es decir cada minuto, hasta los 100 minutos
// Esta parametrización es el resultado de varias depuraciones del código que han
// dado finalmente con el dominio de definición de h(t) (domh={0<t<88})
// La condicion inicial h(0)=2,41 resulta de los resultados del apartado anterior
[t,y] = euler(f,0,2.41,60,87);

// Graficamos los resultados, transformando las unidades temporales a minutos,
//para una lectura más cómoda de la gráfica
plot(t/60,y,'red');

xgrid();
```

Figura 25 Definición función de euler en Scilab —continuación— (ejemplo 2)

El resultado se ha comparado con el algoritmo numérico (ode) que incorpora Scilab para resolver este tipo de ecuaciones diferenciales de valor inicial.

```

//Método 2- Utilizando la función ode de Scilab
// Los saltos se hacen de 60 en 60 segundos. Tras varias depuraciones con este
//código se ha obtenido que para valores de t=5160 seg,
// es decir a los 86 minutos del comienzo del vaciado, el tanque esta casi vacío.
// Si extendieramos el t más allá de estos 86 minutos, la función devuelve error
// ya que en ese caso h(t)=0 y f(t,h(t)) no estaría definida
// pues el denominador se anula.
t0=0;y0=2.41;t=0:60:5160;
yt = ode(y0,t0,t,f)
plot2d(t/60,yt)

//Maquillamos la gráfica
legend("Método Euler","Algoritmo Scilab");
xtitle("h(t) vs  t", "t(minutos)","h(t) en m");

//THE END
```

Figura 26 Definición función ode de Scilab (ejemplo 2)

El resultado de la comparativa se muestra en el gráfico 5.

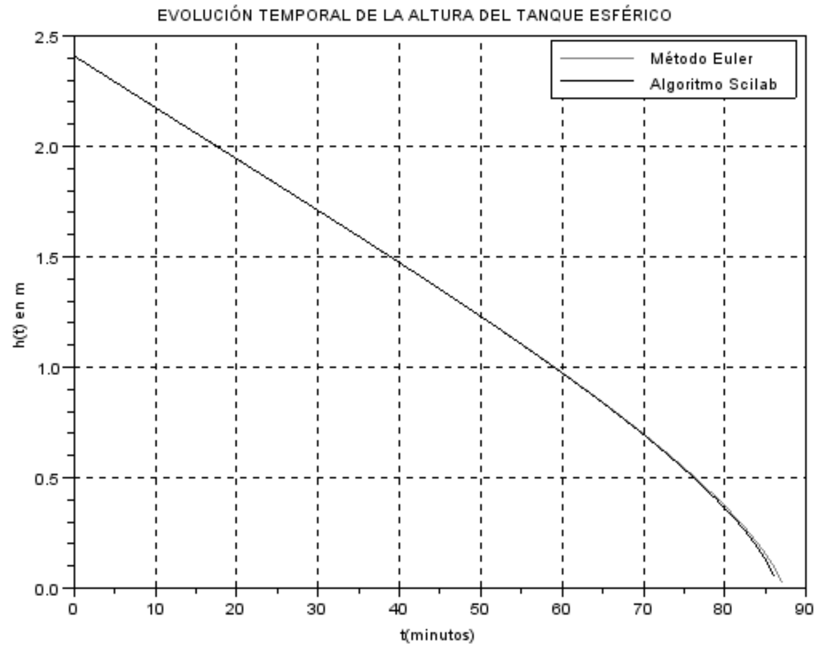

Gráfico 5 Comparativa método Euler y función ode

Al contrario, que en otros ejemplos similares de vaciado de depósitos (en la guía de la asignatura de Herramientas Informáticas para las Matemáticas se propone un ejercicio con depósitos cilíndricos y con depósitos cónicos) aquí, la resolución exacta del problema no es posible.

En efecto, si tratamos de resolver la ecuación diferencial mediante integración separando variables…

$$\frac{dh}{dt} = -\frac{CA}{\pi}.\frac{\sqrt{2gh}}{2hR - h^2} \overset{R=3}{\Longrightarrow} \frac{6h - h^2}{\sqrt{h}}.dh = -\frac{CA}{\pi}.\sqrt{2g}dt$$

... y poniendo límites obtenemos la integral

$$\int_{h(0)}^{h(t)} \frac{6h - h^2}{\sqrt{h}}.dh = \int_{o}^{t} -\frac{CA}{\pi}.\sqrt{2g}dt$$

..cuya resolución es inmediata…

$$4h^{3/2} - \frac{12}{5}h^{5/2} - \left(4h(0)^{3/2} - \frac{12}{5}h(0)^{5/2}\right) = -\frac{CA}{\pi}.\sqrt{2g}t$$

Es decir, sustituyendo h(0), y las constantes C,A, pi y g

$$4h^{3/2} - \frac{12}{5}h^{5/2} = 0{,}0022t - 6{,}67$$

… en que no podemos obtener explícitamente h(t).

En la depuración del código, se han probado varias posibilidades para el paso y los límites de la variable temporal, hasta dar con la solución que se propone.

Dada la situación del enunciado (tamaño del depósito y del agujero de escape), era lógico pensar que el vaciado del depósito tardaría varios minutos en producirse y al menos una hora (como así muestra el resultado).

Por ello, se ha considerado un paso de 60 en 60 segundos, que en la tabulación y graficación de resultados se ha transformado en minutos.

El vaciado del depósito se produce aproximadamente a los 87 minutos aproximadamente de la situación inicial, lo cual parece razonable.

La evolución temporal del vaciado es casi lineal hasta minutos antes del vaciado total, en que se produce una variación de la altura del líquido más brusca (parece razonable que esto sea asi, si pensamos en lo que esto significa, es decir, que cuando el tanque casi está vacío, el vaciado se produce más rápidamente, pues apenas hay volumen).

En definitiva, el tanque esférico de 3 metros de diámetro tarda casi una hora y media en vaciarse, partiendo de una situación inicial de semillenado (h(0)=2,41 m) siendo la variación de altura casi lineal con un aumento ligero en la variación al final del vaciado.

Como variantes casi inmediatas del problema, se pueden recrear situaciones geométricas diferentes (variación de R e incluso de A- tamaño del orificio de salida), de características del líquido (variaciones de C) o de situación inicial (variacio de h(0) entre 2R y 0).

Otras posibilidades de resolución, incluyendo comparativas entre métodos diferentes de resolución de ecuaciones para h (0) o de problemas de valor inicial para dh/dt requieren algo más de tiempo, pero el resultado no ha de diferir sustancialmente a los resultados ofrecidos en la presente solución. El comando *ode* de Scilab ofrece parámetros que permiten variaciones en el método de resolución de la ecuación diferencial del comando.

CONCLUSIÓN

Los dos ejemplos mostrados en este artículo de resolución de sendos problemas de diseño de una estructura para un invernadero y determinación del tiempo de vaciado de un depósito utilizando Scilab y Maxima, muestran las bondades de dichos programas de software libre, tanto en el cálculo numérico como en las posibilidades gráficas.

BIBLIOGRAFÍA

[1] R . L . Burden, J . D . Faires. *Análisis Numérico*. Grupo Editorial Norteamericana (1985)

[2] A. Cordero, J.L. Hueso, E .Martínez, J.R. Torregrosa. *Problemas Resueltos de Métodos Numéricos*. Thompson (2006)

[3] C. Moreno. *Introducción al cálculo numérico*. UNED (2011)

[4] F. Morilla, M.A. Rubio. *Herramientas Informáticas para Matemáticas*. Apuntes elaboradas por el equipo docente (2011)