PROBLEMAS CON VALORES INICIALES *STIFF* Y NO *STIFF* EN EDO USANDO MATLAB

Iojany Abigail Valle Queb*
Justino Alavez Ramírez
Universidad Juárez Autónoma de Tabasco
`*202a31002@alumno.ujat.mx`

1. Introducción

En este trabajo discutimos resultados numéricos de dos problemas con valores iniciales en ecuaciones diferenciales ordinarias (EDO) que reportan Omale *et al.* (2014) [2]. Ellos resuelven un problema *stiff* y un problema no *stiff*, utilizando los solucionadores de MATLAB. La importancia de estos problemas radica en que se utilizan para modelar sistemas complejos en diversas áreas científicas. Aquí resolvemos nuevamente los dos problemas mencionados anteriormente usando los solucionadores ode45, ode23, ode113, ode15s, ode23s y ode23t de MATLAB y OCTAVE, y se comparan los resultados con los resultados que se reportan en [2].

2. Ecuaciones diferenciales ordinarias *stiff*

Las EDO *stiff* son comunes en áreas como la cinética química, los circuitos eléctricos, las vibraciones y los sistemas de control. La propiedad *stiff* es un concepto importante pero difícil de definir en el estudio de estas ecuaciones. Aunque no existe una definición ampliamente aceptada, se han propuesto varios enfoques para describir la propiedad *stiff.* Algunas de las definiciones que menciona Omale *et al.* (2014) [2] son las siguientes:

1. Un problema se considera *stiff* cuando no hay componentes inestables en su solución (es decir, ningún valor propio de la matriz Jacobiana tiene una parte real significativamente grande y positiva) y al menos una componente es altamente estable (al menos un valor propio tiene una parte real grande y negativa).
2. Un problema se considera *stiff* si la solución que se busca varía lentamente pero existen soluciones cercanas que varían rápidamente, lo cual implica que el método numérico debe tomar pasos muy pequeños para obtener resultados satisfactorios.
3. La propiedad *stiff* ocurre cuando algunos componentes de la solución decaen más rápidamente que otros.
4. Un sistema de ecuaciones diferenciales lineales

   $$\dot{u} = Au(t), \quad t \in [0,T], \quad u(0) = u_0,$$

   se considera *stiff* cuando los valores propios $\lambda_1, \lambda_2, \ldots, \lambda_m$ de la matriz A de coeficientes constantes del sistema tienen las siguientes propiedades:

   a) Re $(\lambda_k) < 0$ para cada $k = 1, 2, \ldots, m$ (todos los valores propios tienen parte real negativa).

   b) El número S definido como

   $$S = \frac{\max_k |\mathrm{Re}\,(\lambda_k)|}{\min_k |\mathrm{Re}\,(\lambda_k)|}$$

   es grande, es decir, $S \gg 1$.
5. Un problema es *stiff* si los métodos explícitos fallan para obtener soluciones del problema o trabajan demasiado lento.

La mejor manera de detectar la propiedad *stiff* es intentar resolver el problema con un solucionador diseñado para sistemas no *stiff.* Si esto falla, entonces es probable que el problema sea *stiff* y se deben utilizar solucionadores especializados para este tipo de problemas.

3. Ejemplo de un problema no *stiff*

Las ecuaciones de Lorentz son un ejemplo de un problema no *stiff*, se utilizan para simular la convección de una capa de fluido de extensión horizontal infinita calentada desde abajo. El modelo es una versión simplificada del calentamiento de la atmósfera. El sistema es:

$$\begin{aligned} \dot{x} &= \sigma(-x+y) \\ \dot{y} &= rx - y - xz \\ \dot{z} &= xy - bz, \end{aligned} \tag{3.1}$$

donde σ, r y b son constantes que resultan de combinar parámetros físicos del problema. Resuelva las ecuaciones de Lorentz sobre el intervalo de tiempo $[0, 2]$ para la siguiente combinación de parámetros:

$$\sigma = 10,\ r = 75,\ b = 2.666,\ x(0) = 1,\ y(0) = 1,\ z(0) = 1.$$

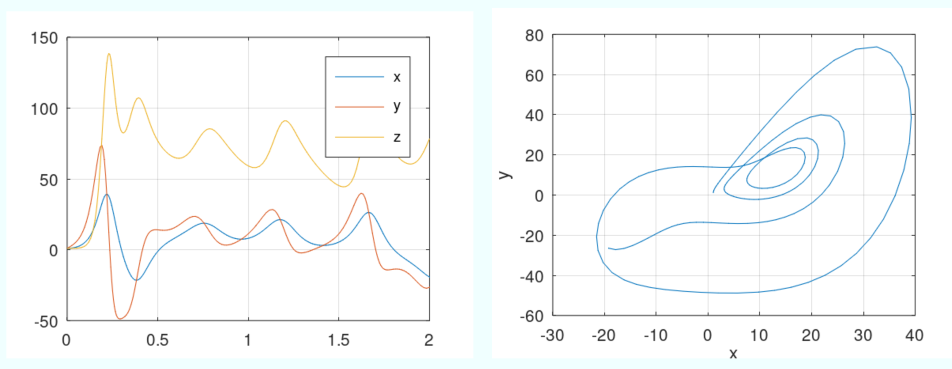
Grafica las señales x vs t, y vs t, z vs t, así como los retratos fase y vs x, z vs x y z vs y.

Solución. La codificación del sistema es:

```
function dydt = p4 (t,y)
dydt=[10*(-y(1)+y(2));
75*y(1)-y(2)-y(1)*y(3);
y(1)*y(2)-2.666*y(3)];
end
```

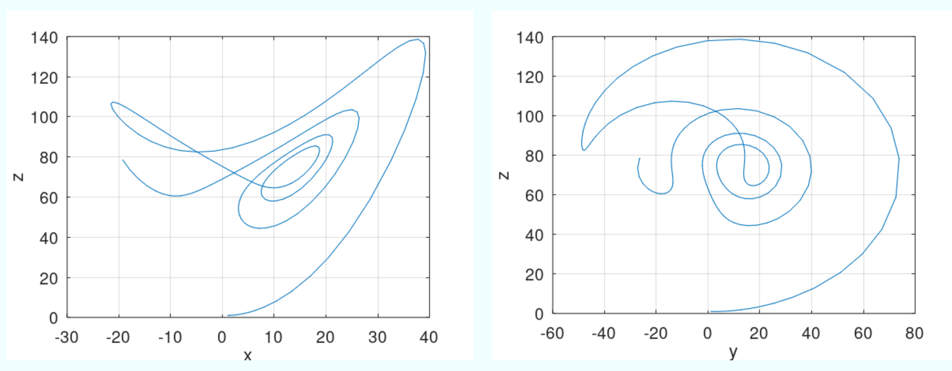
Luego se aplica el solucionador Ode45 (similarmente para Ode23 y Ode113) en MATLAB Online (básico) 2024 y en OCTAVE versión 9.2.0. La salida gráfica del solucionador se muestra en la figura 1(a), y la salida gráfica de los retratos fase en el plano se muestran en las figuras 1(b) y 2; todo con el siguiente código:

```
tspan=[0,2];
y0=[1;1;1];
%
opts = odeset('Stats','on')
t1=cputime;
[t,y]=ode45(@p4,tspan,y0,opts);
t2=cputime-t1
%
% Gr\'afico de soluciones
figure(1)
plot(t,y); grid on
legend('x','y','z')
% Retrato de fase
figure(2)
plot(y(:,1),y(:,2)); grid on
xlabel('x')
ylabel('y')
figure(3)
plot(y(:,1),y(:,3)); grid on
xlabel('x')
ylabel('z')
figure(4)
plot(y(:,2),y(:,3)); grid on
xlabel('y')
ylabel('z')
```



a) Gráficos de x, y, z vs t. b) Gráfico de y vs x.

Figura 1: *Solución numérica del sistema* (3.1).



a) Gráfico de z vs x. b) Gráfico de z vs y.

Figura 2: *Retratos de fase en el plano del sistema* (3.1).

La comparación de los resultados del sistema (3.1) obtenidos con los solucionadores Ode45, Ode23 y Ode113 con MATLAB y OCTAVE y con los resultados que reportan Omale *et al.* (2014) [2], se muestran en la tabla 1.

Sol.	Software	Tiempo (seg)	PE	IF	EF
Ode45	MATLAB (art)	0.025690	49	8	343
	MATLAB	0.052000	49	8	343
	OCTAVE	0.056250	46	12	349
Ode23	MATLAB (art)	0.035048	145	14	478
	MATLAB	0.016998	145	14	478
	OCTAVE	0.0796875	155	7	487
Ode113	MATLAB (art)	0.054591	140	10	291
	MATLAB	0.074997	140	10	291
	OCTAVE	No existe código			

Tabla 1: *Comparación de ode45, ode23 y ode113 en la solución del sistema* (3.1). *Los resultados que reportan Omale et al. (2014) [2] corresponden a MATLAB (art). Pasos exitosos (PE), intentos fallidos (IF) y evaluación de funciones (EF).*

4. Ejemplo de un problema *stiff*

Se aplican ahora los solucionadores Ode15s, Ode 23s y Ode23t de MATLAB para resolver un típico problema *stiff*, el llamado problema de Robertson que se cita a continuación. Considere el siguiente sistema en el intervalo $0 \leq t \leq 5000$.

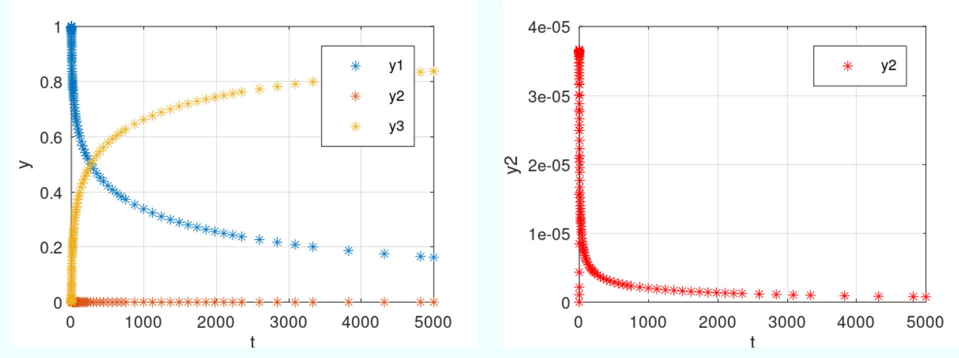
$$\begin{aligned} \dot{y}_1 &= -0.04y_1 + 10^4 y_2 y_3 \\ \dot{y}_2 &= 0.04y_1 - 10^4 y_2 y_3 - 3 \times 10^7 y_2^2 \\ \dot{y}_3 &= 3 \times 10^7 y_2^2 \end{aligned} \tag{4.1}$$

con las condiciones iniciales $y_1(0) = 1$, $y_2(0) = 0$, $y_3(0) = 0$.

Solución. La codificación del sistema queda de la siguiente manera:

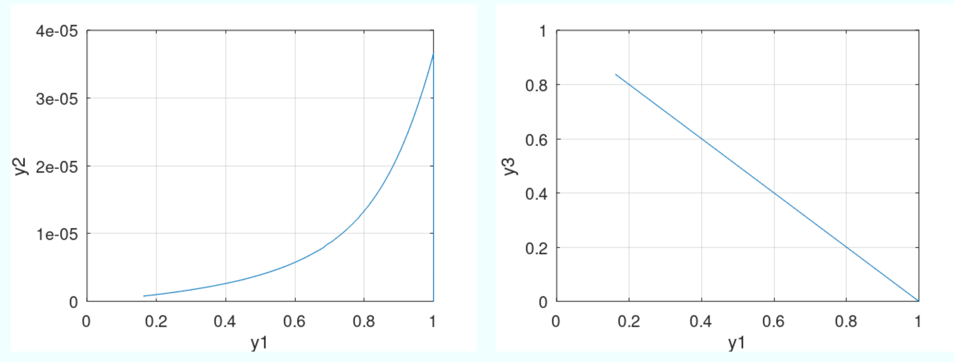
```
function dydt = p6(t,y)
dydt=[-.04*y(1)+10^4*y(2)*y(3);
0.04*y(1)-10^4*y(2)*y(3)-3*10^7*y(2).^2;
3*10^7*y(2).^2];
end
```

Con un código similar al usado en la sección 3, pero con los solucionadores Ode15s, Ode23s y Ode23t, se resuelve el sistema (4.1). Las salidas gráficas del solucionador se muestran en la figura 3, y las salidas gráficas de los retratos de fase en el plano se muestran en las figuras 4 y 5.



a) Gráficos de y_1, y_2, y_3 vs t. b) Gráfico de y_2 vs t.

Figura 3: *Solución numérica del sistema* (4.1).



a) Gráfico de y_2 vs y_1. b) Gráfico de y_3 vs y_1.

Figura 4: *Retratos de fase en el plano del sistema* (4.1).

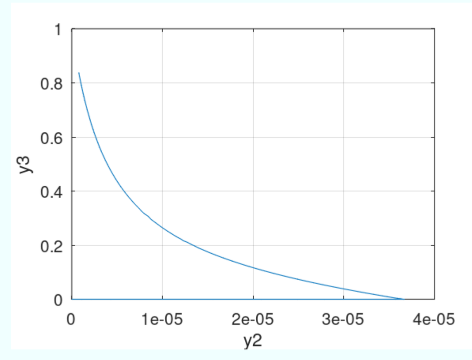


Figura 5: *Retrato de fase en el plano:* y_3 *vs* y_2 *del sistema* (4.1).

La comparación de los resultados del sistema (4.1) obtenidos con Ode15s, Ode23s y Ode23t en MATLAB y OCTAVE y con los resultados que reportan Omale *et al.* (2014) [2], se muestran en la tabla 2.

Sol.	Software	Tiempo (seg)	PE	IF	EF	DP	DLU	SSL
Ode15s	MATLAB (art)	0.067788	107	11	261			227
	MATLAB	0.077993	87	11	199	7	31	169
	OCTAVE	0.042188	110	4	188			
Ode23s	MATLAB (art)	0.705399	93	2	565			285
	MATLAB	0.058999	34	2	211	34	36	108
	OCTAVE	0.076563	45	8	265			
Ode23t	MATLAB (art)	0.520620	99	8	284			250
	MATLAB	0.078992	79	11	223	8	55	189
	OCTAVE	No existe código						

Tabla 2: *Comparación del ode15s, ode23s y ode23t para el sistema* (4.1). *Los resultados que reportan Omale et al. (2014) [2] corresponden a MATLAB (art). Pasos exitosos (PE), intentos fallidos (IF), evaluación de funciones (EF), derivadas parciales (DP), descomposición LU (DLU) y solución de sistemas lineales (SSL).*

5. Conclusiones

Lo primero que notamos es que OCTAVE no dispone del solucionador Ode113 y Ode23t de MATLAB (ver tablas 1 y 2). También hacemos notar que todos los solucionadores se aplicaron con tolerancia predeterminada, es decir, con tolerancia absoluta de 10^{-6} y tolerancia relativa de 10^{-3}. Solo se incluyó en cada caso el parámetro opcional *Stats* para reportar las estadísticas del solucionador.

Para el caso del problema no *stiff*, se observa en la tabla 1 que MATLAB es más eficiente para resolverlo que OCTAVE.

En el caso del problema *stiff*, en la resolución del problema de Robertson, se observa en la tabla 2 que los resultados obtenidos tanto en MATLAB como en OCTAVE con los solucionadores Ode15s y Ode23s, son diferentes entre sí y diferentes con los resultados reportados por Omale *et al.* (2014) [2]. También se observa que los gráficos de las figuras 3 al 5 difieren de las gráficas reportadas por Omale *et al.* (2014); consideramos que la razón es porque ellos interpretan equivocadamente los coeficientes del sistema (4.1).

Finalmente, se corrobora lo que afirma Omale *et al.* (2014) [2], que MATLAB es muy efectivo y preciso para obtener soluciones numéricas de problemas con valor inicial de EDO, además de que esta solución se nos presenta en cuestión de segundos, lo cual si lo hacemos de manera manual nos llevaría hasta varios días.

Referencias

[1] Hairer, E. and Wanner, G. (1996). *Solving Ordinary Differential Equations. II. Stiff and Differential-Algebraic Problems*, 2^{nd} ed. Berlin: Springer-Verlag.

[2] Omale, D., Ojih, P.B., Ogwo, M.O. (2014). Mathematical Analysis of Stiff and Non-Stiff Initial Value Problems of Ordinary Differential Equation Using Matlab. *International Journal of Scientific & Engineering Research*, **5**(9): 49–59.