Métodos Lineales Multipaso. Métodos de Adams y BDF

Guión de la práctica:

En esta práctica implementaremos en MATLAB los métodos lineales multipaso para resolver el problema de valor inicial

$$u' = f(t, u), \quad t \in [0, T], \quad u, f \in \mathbb{R}^m \tag{1}$$
$$u(t_0) = u_0 \tag{2}$$

Qué es un método multipaso

Consideramos un partición del intervalo de integración a paso fijo de longitud $h > 0$:

$$t_n = nh, \quad n = 0, 1, \ldots, N = T/h,$$

y u_n la aproximación de $u(t_n)$. Los métodos multipaso construyen u_{n+1} a partir de la información de varios pasos anteriores: u_n, u_{n-1}, u_{n-2},..., con el fin de reducir el número de veces que evaluamos f.

Un método lineal multipaso (MLM) de k pasos se define con una fórmula del tipo

$$\sum_{j=0}^{k} \alpha_j u_{n+j} = \Delta t \sum_{j=0}^{k} \beta_j f_{n+j}, \quad n = 0, 1, \ldots, N_h, \quad N_h = T/h - k, \tag{3}$$

donde $f_{n+j} = f(t_{n+j}, u_{n+j})$ e $\{u_{n+j}\}_{j=0}^{k-1}$ son k aproximaciones a la solución de (1) en los puntos anteriores de la red $\{t_n, t_{n+1}, \ldots t_{n+k-1}\}$, suponiendo que se parte de las k primeras aproximaciones a la solución en $t_0, t_1, \ldots, t_{k-1}$ conocidas.

Definición: Los valores α_j, β_j, $j = 0, \ldots k$ se llaman **coeficientes del MLM**.

Definición: Si $\beta_k = 0$ se dice que el MLM es **explícito**, en caso contrario se llama MLM **implícito** ya que en cada paso de integración hay que resolver un sistema implícito.
Observaciones:

- Si $\alpha_0 = \beta_0 = 0$ el método no usaría la información del punto (t_n, y_n), por lo que en realidad sería un método de $k - 1$ pasos. Por eso también se exigirá siempre que $|\alpha_0| + |\beta_0| > 0$.
- Si multiplicamos a ambos lados la ecuación del método por una constante $\lambda \neq 0$ obtenemos otros coeficientes pero el método no varía. Para fijar los coeficientes se normalizan considerando $\alpha_k = 1$.

Método explícito de Adams-Bashforth de k pasos

Los métodos de Adam-Bashforth (AB) se deducen a partir de la identidad

$$\int_{t_{n+k-1}}^{t_{n+k}} u'(t)dt = \int_{t_{n+k-1}}^{t_{n+k}} f(t, u(t))dt. \tag{4}$$

Se aproxima $f(t,u(t))$ mediante su polinomio de interpolación en los k puntos de la red anteriores

$$\{t_n, t_{n+1}, \ldots, t_{n+k-1}\},$$

donde los valores de la función son conocidos:

$$\{f(t_n,u_n), f(t_{n+1},u_{n+1}), \ldots, f(t_{n+k-1},u_{n+k-1})\}$$

Entonce, volviendo a (4), se evalúa la integral y resulta:

$$u_{n+k} = u_{n+k-1} + h\sum_{j=0}^{k-1} \beta_{k,j} f_{n+j}.$$

Estos coeficientes $\beta_{k,j}$ se calculan a partir de los coeficientes del polinomio de interpolación. Sus primeros valores son:

k	$\beta_{k,0}$	$\beta_{k,1}$	$\beta_{k,2}$	$\beta_{k,3}$
1	1			
2	$-\frac{1}{2}$	$\frac{3}{2}$		
3	$\frac{5}{12}$	$-\frac{16}{12}$	$\frac{23}{12}$	
4	$-\frac{9}{24}$	$\frac{37}{24}$	$-\frac{59}{24}$	$\frac{55}{24}$

Programemos en MATLAB el siguiente programa para este método:

```
function [u,t]=Adams_Bashforth(f,tf,t0,N,u0,beta)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  Esta función resuelve el problema de valor inicial
%       u'=f(t,u)
%       u(t0)=u0
% utilizando el método
%
%          [u,t]=Adams_Bashforth(f,tf,t0,h,u0,beta)
%
%  Variables de Entrada:
%
%       f: vector columna. función que rige el sistema de EDO,
%          tiene dos argumentos f(t,u) donde t es escalar
%          y u vector columna.
%       N: número de pasos
%       t0: tiempo inicial
%       tf: tiempo final
%       u0: vector columna. Dato inicial
%       beta: vector fila. Coeficientes de interpolación
%
%  Variables de Salida:
%
%       u: matriz de length(u0) x length(t) que contiene la solución
%       t: vector de tiempos
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Métodos basados en la diferenciación numérica o BDF de k pasos

Estos métodos parten de un planteamiento muy diferente al de los métodos de Adams. En estos se integra un polinomio de interpolación $u'(t)$, mientras que en los BDF (backward Differential Formulae) se deriva el polinomio interpolador de $u(t)$.

Entrando en detalle, se calcula el polinomio de interpolación $\pi(t)$ de la solución numérica en los k puntos anteriores y en el punto actual t_{n+k}, o sea, el polinomio que pasa por los $k+1$ puntos $\{(t_{n+j}, u_{n+j})\}_{j=0}^{k}$, y luego se exige a dicho polinomio que verifique la ecuación diferencial en el punto actual, es decir,

$$\pi'(t_{n+k}) = f(t_{n+k}, \pi(t_{n+k})).$$

A partir de esta ecuación, se obtiene el método BDF de k pasos implícito:

$$\sum_{j=0}^{k} \alpha_{k,j} u_{n+j} = h f_{n+k}.$$

Los valores de los coeficientes $\alpha_{k,j}$ para $k \leq 4$ son:

k	$\alpha_{k,0}$	$\alpha_{k,1}$	$\alpha_{k,2}$	$\alpha_{k,3}$	$\alpha_{k,4}$
1	-1	1			
2	$\frac{1}{2}$	-2	$\frac{3}{2}$		
3	$-\frac{1}{3}$	$\frac{3}{2}$	-3	$\frac{11}{6}$	
4	$\frac{1}{4}$	$-\frac{4}{3}$	3	-4	$\frac{25}{12}$

Programemos en MATLAB dos programas basados en BDF.

En primer lugar, transformaremos el BDF mediante el algoritmo de **predictor-corrector** en un método explícito usando como predictor el método explícito de Adams-Bashforth con mismo orden que BDF. Esto lo programaremos en Matlab con la siguiente función:

```
function [u,t]=BDF_pc(f,tf,t0,N,u0,alpha,beta)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  Esta función resuelve el problema de valor inicial
%       u'=f(t,u)
%       u(t0)=u0
% utilizando el método
%
%          [u,t]=BDF_pc(f,tf,t0,N,u0,alpha,beta)
%
%  Variables de Entrada:
%
%       f: vector columna. función que rige el sistema de EDO,
%          tiene dos argumentos f(t,u) donde t es escalar
%          y u vector columna.
%       N: número de pasos
%       t0: tiempo inicial
%       tf: tiempo final
%       u0: vector columna. Dato inicial
%       alpha: coeficientes  de interpolación del BDF de orden k
```

```
%       beta: coeficientes  de interpolación del AB de orden k
%
%  Variables de Salida:
%
%       u: matriz de length(u0) x length(t) que contiene la solución
%       t: vector de tiempos
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

En segundo lugar, aplicaremos el BDF resolviendo la ecuación no lineal asociada mediante el método de Newton. Esto lo programaremos en Matlab con la siguiente función:

```
function [u,t,it]=BDF_Newton(f,df,tf,t0,N,u0,alpha)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  Esta función resuelve el problema de valor inicial
%       u'=f(t,u)
%       u(t0)=u0
% utilizando el método
%
%          [u,t]=BDF_Newton(f,tf,t0,N,u0,alpha)
%
%  Variables de Entrada:
%
%       f: vector columna. función que rige el sistema de EDO,
%          tiene dos argumentos f(t,u) donde t es escalar
%          y u vector columna.
%       df: matriz cuadrada. jacobiano de la función que rige las EDOs,
%           tiene dos argumentos df(t,u) donde t es escalar
%           y u vector columna.
%       N: número de pasos
%       t0: tiempo inicial
%       tf: tiempo final
%       u0: vector columna. Dato inicial
%       alpha: coeficientes  de interpolación del BDF de orden k
%
%  Variables de Salida:
%
%       u: matriz de length(u0) x length(t) que contiene la solución
%       t: vector de tiempos
%       it: vector del número de iteraciones de Newton en cada paso
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Comentarios: Para realizar esta práctica se puede implementar todas las funciones o ficheros .m auxiliares que se necesiten aparte de los citados aquí.

Ejercicios:

1. Utilizar la función Adams-Bashforth y BDF para resolver los problemas de valor inicial:

a) En el intervalo $[0, 10]$,

$$\begin{cases} u' = \begin{bmatrix} -2 & 1 \\ 1 & -2 \end{bmatrix} u + \begin{bmatrix} 2\sin(t) \\ 2(\cos(t) - \sin(t)) \end{bmatrix} \\ u(0) = \begin{bmatrix} 2 \\ 3 \end{bmatrix} \end{cases}$$

b) Con $\lambda = 1$, en el intervalo $[0, 10]$.

$$\begin{aligned} u'(t) &= -\lambda(u(t) - \cos(t)) - \sin(t) \\ u(0) &= 1 \end{aligned}$$

2. Representar gráficamente la solución aproximada para cada uno de los problemas con $k = 1, 2, 3, 4$.

3. Para el problema *b*) :

   i) Hacer pruebas tomando diferentes datos iniciales.

   ii) Hacer pruebas tomando diferentes valores de $\lambda = 1, 2, 3, 4, 5, 10...$

4. Aportar conclusiones relacionando los resultados numéricos obtenidos con lo visto en teoría.