E.T.S. de Ingenieros Industriales **2011-12**

MÉTODOS MATEMÁTICOS I **(todos los bloques)**

TRABAJO PRÁCTICO núm 1 **hasta 18 NOV 11**

almacenar en f???rkor.c

```
/* indicar si se debe corregir con  DJGPP o con ANSI_C_Carpanta */

void f???rkor(int opcion,int n,double (*der)(),double (*ver)(),double y[],
int npas,double trab1[],double trab2[],double trab3[],double trab4[],double *orden)

/*
DESCRIPCION:

integra en  [0,1]  un sistema autonomo de  n  ecuaciones de primer orden
con  n  funciones incognita,
          y1' = f1 ( y1, ..., yn)
          ......................
          yn' = fn ( y1, ..., yn)
con una condicion inicial en  0
          y1(0) = z1, ..., yn(0) = zn
Para ello emplea el metodo 'Runge-Kutta clasico'.
La función argumento  *ver proporciona la verdarera solucion para el sistema
dado por  *der : la funcion  *ver  proporciona el valor en cualquier abscisa
t  de la verdadera solucion que en   0   vale  (z1,...,zn)

Cuando opcion vale 1, la función realiza una integracion en  [0,1]
(en cualquier intervalo de longitud 1 ), dando  npas  pasos.
A la entrada, el argumento  y[]  contiene los valores de  z1,...,zn ;
a la salida, contiene el valor de la solucion en  1 .

Cuando opcion vale 2, la función se emplea para averiguar el orden
efectivo que el método alcanza  para el sistema *der . Para ello integra el
sistema  *der  con pasos 1/100 y 1/1000, evalua los errores (se usará la
norma euclídea para la evaluación de los errores) y calcula el orden
efectivo que se alcanza, valor que incluye en  *orden. No se examinan en
este caso los valores de los otros argumentos

PRECISION: doble

DESCRIPCION DE LOS ARGUMENTOS:

La funcion posee vectores de tamano arbitrario (pasado por el usuario)
y[]  y  trab1[],trab2[],trab3[],trab4[] . Cualquier otro vector o matriz con
que se desee trabajar en la funcion debera ser de tamano constante
Como hay que utilizar la raiz cuadrada y log10 sera necesario  #include <math.h>

int opcion      contiene  1 o 2  para indicar el modo de empleo de la
                funcion

int n           contiene el numero de ecuaciones y de funciones incognita

double (*der)() puntero de una funcion que se encarga de evaluar las
                derivadas  f1, ...,fn .  La funcion  der()  debe ser
                creada por el usuario en la siguiente forma
                double der(int n,int k,double y[])
                ...........
                A la salida,  der()  debe devolver el valor de la derivada
                fk  en  y[1],...,y[n]  sin modificar los argumentos
```

```
double (*ver)() puntero de una funcion que se encarga de evaluar la
                verdadera solucion. La funcion  ver()  debe ser creada
                por el usuario en la siguiente forma
                double ver(int n,int k,double t)
                ...........
                A la salida,  ver()  debe devolver el valor en  t  de la
                componente  k  de la verdadera solucion que en  0  vale
                (z1,...,zn)

double y[]      vector de  n+1  componentes que pasa el usuario.
                A la entrada,  y[k]  contiene el valor  zk  de la solucion
                yk  en 0 . A la salida,  y[k]  contiene el valor de la
                solucion  yk  en la abscisa final  1

int npas        contiene el numero de pasos que se van a dar en  [0,1]

double trab1[],trab2[],trab3[],trab4[] son vectores de  n+1  componentes que
                pasa el usuario. Son vectores de trabajo de la rutina.
                Sus valores a la entrada y a la salida son irrelevantes.

double *orden   puntero que apunta a la variable double orden. En la opcion 1
                y a la entrada en cualquier caso, el valor de  *orden
                es irrelevante. Para la opcion 2 y a la salida,  *orden
                contiene el orden efectivo del metodo para el sistema *der
                calculado con ayuda de los pasos 1/100 y 1/1000
*/
```