UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE CIENCIAS

"EL METODO ITERATIVO DE EULER CON PASO VARIABLE"

T E S I S

QUE PARA OBTENER EL TITULO DE

M A T E M A T I C O

P R E S E N T A

JOSE FEDERICO OLVERA RINCON

MEXICO, D. F. 1991

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

PROLOGO

La aplicación de las ecuaciones diferenciales en muchos fenómenos es amplia: al referirnos a razones de cambio, fenómenos de crecimiento económico, demográfico, epidemiológico, etc, estamos involucrando una ecuación diferencial la cual puede ser muy difícil o imposible de resolver analíticamente,es decir, no es posible representarla a través de una función elemental o puede ser muy difícil. Es por esto que desde hace muchos años se han desarrollado métodos que permiten "construir" al menos gráficamente la función solución, basándose en el comportamiento que esta presenta. Desafortunadamente durante muchos años no se dispuso de computadoras que permitieran aplicar desarrollos numéricos, ya que algunos requieren de una gran cantidad de iteraciones para llegar a la solución requerida. En la actualidad existe una gran variedad de métodos de distintas complejidades y características, los cuales resuelven numéricamente una ecuación diferencial mediante distintas metodologías.

Para cada función el comportamiento de la pendiente es distinto, y aun lo es dentro de distintos segmentos de ella misma. Es por esto que debemos tener especial cuidado al estimar valores para una ecuación diferencial en los intervalos que lo requieran. El método de paso variable se ocupa de esto.

El propósito de esta tesis es la de introducir el método del paso variable (el cual es muy utilizado en los cursos de análisis numérico a nivel licenciatura), y presentar el desarrollo de un programa que utilice este método tratando de aprovechar al máximo las propiedades del paso variable con un método numérico muy simple; todo esto sin pretender implantar un programa que resuelva con gran precisión todas las ecuaciones diferenciales, ya que es imposible, simplemente se busca aprovechar las propiedades del este método, entender cuando confiar en los resultados y ver el comportamiento de algunas variables que lo calculan.

El resolver una ecuación diferencial numéricamente no se limita a utilizar un programa que la construya en algún punto determinado, sino que existe un ambiente que primero debe ser entendido. Lo anterior será motivado, al igual del porqué resolver una ecuación diferencial numéricamente, si ni siquiera sabemos si existe la solución o si esta es única y en que regiones es más delicado calcular la solución.

"EL METODO ITERATIVO DE EULER CON PASO VARIABLE"

INTRODUCCION

1.1 PLANTEAMIENTO DEL PROBLEMA MATEMATICO A RESOLVER NUMERICAMENTE.

En la actualidad, el uso de las ecuaciones diferenciales ha abarcado muchos campos y ha tenido dentro de estos un importante desarrollo. Entre estos nos encontramos con la medicina y la ingenieria (principalmente la espacial y aérea).

Aunado al desarrollo de la computación y del análisis numérico, problemas que anteriormente no habian podido ser resueltos analiticamente han sido ya solventados en forma numèrica.

En el presente trabajo nos dedicaremos a resolver el problema matemático bien conocido, a saber: una ecuación diferencial ordinaria de orden 1 con condición inicial, la cual en notación matemática se expresa como:

$$y'(t) = f(t,y)$$
$$y(t_0) = y_0 \qquad \ldots\ldots\ldots\ldots (1)$$

1.2 TEOREMA DE EXISTENCIA Y UNICIDAD.

No tendría caso que resolviéramos una ecuación diferencial numéricamente para cualquier valor "μ" si no existe $y(\mu)$ o hay más de una solución; estaríamos calculando sin razón un valor que nada tiene que ver con nuestro problema. Es por esto que requerimos de una herramienta que nos permita saber si existe y es única $y(\mu)$ para una "μ" determinada.

Para evitar encontrarnos con problemas de existencia y unicidad, se citarán los 2 siguientes teoremas que garantizan que la solución existe y es única, el primero de estos esta basado en la condición de Lipschitz:

Teorema 1:

Dada la ecuación diferencial:

$y'(t) = f(t,y)$ con $t_0 \leq t \leq t_0 + a$, $(a > 0)$

si f cumple con:

$|f(t,y) - f(t,x)| \leq L|y-x|$ para algúna $L \in \mathbb{R}^+$, entonces $y(t)$ existe y es única para $t_0 \leq t \leq t_0 + a$. $(a > 0)$

Podemos ver más generalmente que existe al menos un punto "η" tal que:

$$|f(t,u)-f(t,v)| = \left|\frac{\partial f(t,\eta)}{\partial y}\right| |(u - v)|, \quad \text{donde} \quad \eta \in [u,v]$$

cuando $\partial f/\partial y$ existe.

Y con esto podemos definir:

$$\left|\frac{\partial f(t,\eta)}{\partial y}\right| \leq L.$$

Un teorema alternativo para demostrar existencia y unicidad, esta basado en que la parcial de "f" respecto a "y" sea acotada.

Teorema 2:

Sea la ecuación diferencial:

$y' = f(t,y)$ continua respecto a "t" y "y", con valor inicial $y(t_0) = y_0$ y cumple,

$|f_y(t,y)| \leq M$ para la región "D" donde esta es,

$t_0 \leq t \leq t_0 + a$ y $|y - y_0| \leq b$, $(a > 0)$

entonces $y(t)$ existe y es única para $t_0 \leq t \leq t_0 + a$, $(a > 0)$

La idea de este último teorema se encuentra ilustrada en la figura 1.1, la cual nos enseña como busca el teorema mostrar la existencia mediante cuadros cerrados y acotados, mismos que determina mediante la razón de cambio de la función $f(t,y)$ respecto de "y". La región "D" esta formada por los puntos $(t_0, y_0 - b)$, $(t_0, y_0 + b)$, $(t_0 + a, y_0 - b)$, $(t + a, y + b)$.

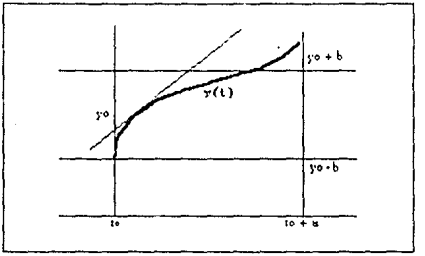

Figura 1.1.- Ilustración del teorema 2

La demostración a los 2 teoremas anteriores se puede encontrar en [9] o en algún otro libro de ecuaciones diferenciales.

Cabe señalar que el teorema 1 es suficiente pero no necesario. Con que se cumpla, sabremos que la solución existe y es única. Más adelante se mostrara esto mediante ejemplos. Por ahora, y antes de pasar a los ejemplos, es necesario señalar que para cada método numérico y a partir de un punto inicial y_0, obtendremos una solución única; es por esto que será necesario dar como punto inicial un punto lo más preciso posible (si es conocido mejor). Esto es esencial para

estar lo más cerca posible de la solución real; como lo muestra la figura 1.2 (curvas solución). A partir del valor inicial dado solo podremos construir una única curva solucion de las infinitas que existen.

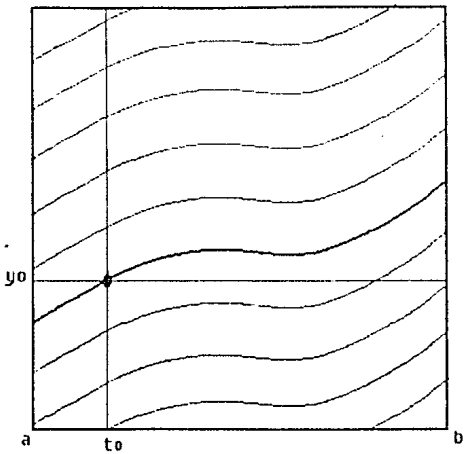

Figura 1.2.- Curvas solución

Para mostrar el uso de los teoremas anteriores se darán a continuación los siguientes ejemplos:

Ejemplo 1: Ecuación que cumple las hipótesis del teorema 2 vía:

$|f_y| \leq M$ y por tanto que

$|f(t,y)-f(t,x)| \leq L|y-x|$ y $M \leq L$

pero que no cumple $|f_y| \leq M$ (hipótesis del teorema 2)

La ecuación:

$$y' = |y|$$

Para Lipschitz en el rango $x, y \in [-p,p]$, $p \in \mathbb{R}^+$ necesitamos:

$$|f(t,x) - f(t,y)| = ||y| - |x|| \leq L|y - x|$$

pero como $||y| - |x|| = \sqrt{y^2 + x^2 - 2|y||x|}$

y además $|y - x| = \sqrt{y^2 + x^2 - 2yx}$

y obviamente $2yx \leq 2|y||x|$

$\Rightarrow ||y| - |x|| \leq |y - x|$ y $\therefore$ "L" es cualquier real tal que $L \geq 1$

Cumpliendose de esta manera Lipschitz; sin embargo el segundo teorema no corre con la misma suerte, ya que:

$$|f_y| = |y / |y|| \leq M \quad (y \neq 0)$$

Pero f_y no existe en cero.

Entonces, esta ecuación tiene solución por ser "f" Lipschitz, pero no por ser $|f_y| \leq M$

La gráfica de la función $f(t,y)$ (figura 1.3) nos muestra continuidad a lo largo de toda la recta "t", y una pendiente que permite el cumplimiento del teorema de Lipschitz; pero, esta ecuación no es derivable en cero, por lo que aquí falla nuestro segundo teorema.

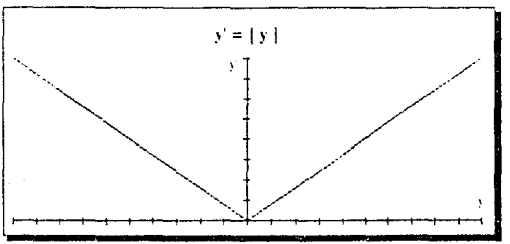

Figura 1.3

Ejemplo 3: Un ejemplo que no cumple las hipótesis y sin embargo tiene solución única.

$$y' = 1/y$$

Para esta ecuacion tenemos, usando Lipschitz:

$$|f(t,x) - f(t,y)| = |1/x - 1/y| \leq L|x-y|$$

Aquí, utilizando el caso particular $x,\ y \in (0,1]$ tenemos:

$$|1/x - 1/y| = |(y-x)/xy| = |x-y||1/xy| \leq L|x-y|$$

$$\Rightarrow |1/xy| \leq L$$

y $|1/xy|$ no se encuentra acotado, ya que cuando "x" ó "y" tiende a cero este tiende a infinito.

Veamos ahora utilizando el otro método:

$$|f_y| = |-1/y^2| = 1/y^2 \leq M$$

Pero aquí pasa lo mismo; utilizando $y \in (0,1]$

$$\lim_{y \to 0} 1/y^2 = \infty \quad \therefore \text{ No existe "M" tal que } |f_y| \leq M$$

De acuerdo a los anteriores resultados de nuestra ecuación diferencial $y' = 1/y$ no podemos saber si $y(t)$ existe en R. Sin embargo tenemos que la solución analítica es $y = \sqrt{2t}$ y por lo tanto existe $\forall\ t \in \mathbb{R}$ tal que $t \geq 0$.

Esto nos muestra que las 2 pruebas utilizadas en estos 3 ejemplos, aunque son suficientes, no son necesarias.

En la siguiente figura (1.4), podremos ver como se encuentran distribuidas las ecuaciones diferenciales dependiendo de el conjunto al que pertenecen.

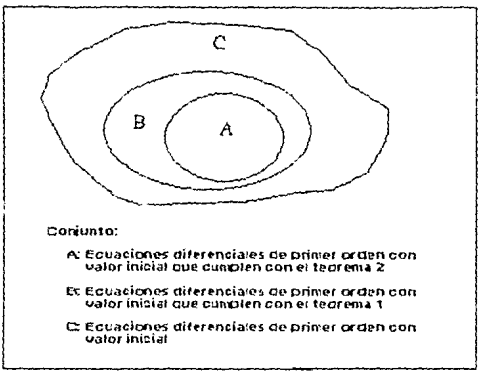

Figura 1.4

En base a lo anterior, una ecuación diferencial que cumple el teorema 2 ($|f_y| \leq M$) cumple también el teorema 1, ya que:

$$\lim_{x \to 0} |f(t,y) - f(t,x)|/|y - x| = \partial y'/\partial y = |f_y| \leq M$$

y como para el primer teorema se debe cumplir:

$$|f(t,y) - f(t,x)| \leq L|y - x| \;\rightarrow\; |f(t,y) - f(t,x)|/|y - x| \leq L$$

Lo cual no tiene solución analítica, es decir, expresada a través de funciones elementales, ya que la integral no puede ser resuelta; sin embargo, por medio de los 2 teoremas anteriores tenemos:

$$|f_y| = 0 \leq 0 = M$$

$$|f(t,x)| - f(t,y)| = 0 \leq 0|x-y| \ , \ L = 0, \ (L = M)$$

∴ La solución existe y es única.

Ejemplo 5:

$y' = dy/dt = t/e^{\text{sen } y}$, con valor inicial $y(1) = 0$

$$e^{\text{sen } y}\, dy = t\, dt \Rightarrow \int e^{\text{sen } y}\, dy = \int t\, dt + C$$

$$\int e^{\text{sen } y}\, dy = (1/2)t^2 + C$$

$$f(t,y) = t/e^{\text{sen } y}$$

$(a,b) \supset [t_0, t_f]$

$[1,10] \subset (0,11)$ $\quad a = 0, \ b = 11$

$$f_y = [te^{-\text{sen } y}]_y = -t\cos y e^{-\text{sen } y}$$

$$|f_y| = |t\cos y e^{-\text{sen } y}| = |t||\cos y||e^{-\text{sen } y}| \leq$$

$$11|e^{-\text{sen } y}| = 11|1/e^{\text{sen } y}| = 11/|e^{\text{sen } y}| \leq 11e$$

$\therefore$ la solución existe y es única.

Como se muestra en los 2 ultimos ejemplos no siempre se puede dar la solución analitica del problema (1), ya que parte de la solución involucra integrales indefinidas las cuales no se pueden escribir en términos de la funciones elementales, como son:

$\int e^{t^2} dt$ y $\int e^{\text{sen}y} dy$ en los ejemplos 4 y 5 respectivamente.

Los ejemplos anteriores podrian ser resueltos mediante algún método numerico de integración, pero otros ejemplos no se pueden representar como los anteriores.

Tomando por ejemplo las funciones:

$$y' = 2t^y$$

$$y' = -1 + 2t + y^2/(1 + t^2)^2$$

$$y' = e^{y^2} - 2t$$

que $y_i \cong y(t_i)$, para $i = 0, 1, \ldots, m$, donde $y(t_i)$ denota la solucion analitica valuada en t_i, y y_i es un valor numerico obtenido mediante algún método numerico, entonces se espera que geométricamente se cumpla $(t_i, y_i) \cong (t_i, y(t_i))$, figura (1.5).

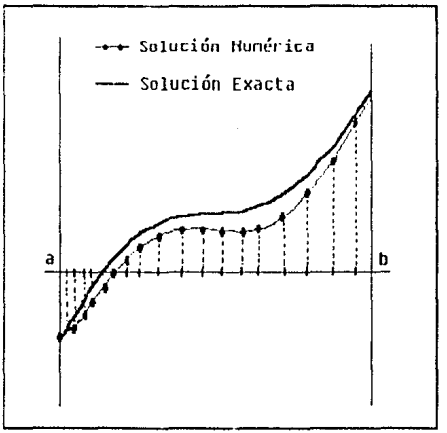

Figura 1.5.- Soluciónes numérica y exacta.

El objetivo principal del presente trabajo, es el de resolver, mediante un programa de cómputo, sistemas de ecuaciones diferenciales utilizando el metodo de Euler con paso variable. La razón por la cual se utiliza Euler es la de introducir mediante el método más sencillo, para no distraernos en otras metodologias, el paso variable, su comportamiento y otros factores que a su alrededor suceden (ambiente), como lo son la estabilidad e inestabilidad, existencia y unicidad,

METODOS RUNGE-KUTTA

Este trabajo ocupará exclusivamente los métodos de Euler (o Runge-Kutta explícito de una etapa) y el método Runge-Kutta explícito de dos etapas, los cuales por ser los más sencillos, simplificarán la metodología que se presentará respecto a estos. Esta metodología puede ser utilizada por otros métodos más complejos y más precisos, pero el objetivo aquí no es el de buscar métodos de alta precisión, sino mostrar el funcionamiento del "paso variable".

Los métodos considerados son llamados de un paso, ya que cada iteración requiere solamente del valor aproximado de la solución en el paso anterior a diferencia de otros métodos llamados multipaso que requieren de valores de dos o más pasos para la creación de una iteración. Esto se ilustra en las figuras 2.1 y 2.2. Para mayor detalle de los métodos multipaso consultar [8].

$y_{k+1} = y_k + hR(t_k, y_k)$, donde $h = t_{k+1} - t_k$ = cateto adyacente.

$y_{k+1} - y_k$ =cateto opuesto.$R(t_k, y_k) = \tan(\alpha)$, donde "$\alpha$" es la pendiente estimada de la función "y" en el punto (t_k, y_k).

La ecuación "R" depende además de la función "f" y de la magnitud de "h", lo que nos da que $y_p = y_{p-1} + hR(t_{p-1}, y_{p-1}, f, h)$, pero de aquí y en adelante se escribirá "R" solo como función de las 2 primeras sobreentendiendo que existe la dependencia con las otras.

Los métodos que veremos en este capítulo requieren de "h" constante, lo que implica que el tamaño de paso es independiente de la función "f". Esto no sucede para el paso variable como veremos más adelante.

Para obtener el valor $y(\ell)$, donde "ℓ" es un punto en el que la función "y" es continua, vamos a "acercarnos" mediante la mencionada sucesión $y_0, y_1, y_2, \ldots, y_n \approx y(\ell)$. Esto es, mediante el punto y_k obtenemos y_{k+1} de la forma siguiente:

$y_{k+1} = y_k + hR(t_k, y_k)$ donde $R(t_k, y_k)$ es una pendiente promedio, la cual puede ser obtenida mediante diversos métodos. La metodología varía en precisión y complejidad y será introducida más adelante.

Los métodos más conocidos son los de Euler, Taylor, Runge-Kutta, Monte Carlo, en diferencias, etc, de los cuales no utilizaremos en

Runge-Kutta de "s" etapas se define como:

$$y_{k+1} = y_k + hR(t_k, y_k) = y_k + h\sum_{n=1}^{s} b_1 f(Y_1)$$

Donde Y_i es el vector $\left(t_k + h\sum_{n=1}^{s} a_{in} \ , \ y_k + h\sum_{n=1}^{s} a_{in} Y_n \right)$

Entonces obtenemos:

$R(t_k, y_k) = b_1 f(Y_1) + b_2 f(Y_2) + \ \ldots \ + b_s f(Y_s)$ lo que nos da un sistema implicito de la forma:

$$f(Y_n) = f\left(t_k + h\sum_{r=1}^{s} a_{nr} \ , \ y_k + h\sum_{r=1}^{s} a_{nr} Y_r \right)$$

Definimos la matriz "A" con los parámetros de la ecuación anterior como:

$$A = \begin{bmatrix} a_{11} a_{12} \ldots\ldots a_{1s} \\ \vdots \qquad\qquad \vdots \\ a_{s1} \ldots\ldots\ldots a_{ss} \end{bmatrix}$$

Todo esto nos deja un sistema implicito, por lo que para resolver esto con mayor facilidad, se igualará el triángulo superior de la matriz "A" a cero (con substituciones hacia atrás), con lo que nos queda un sistema de ecuaciones fácil de resolver . Esto es:

$$R(t_k,y_k) = b_1 f(t_k,y_k) +$$

$$b_2\Big[f(t_k,y_k + ha_{21}f(t_k,y_k)) + a_{21}hf_t(t_k,y_k + ha_{21}f(t_k,y_k)) +$$

$$((a_{21}h)^2/2)f_{tt}(t_k,y_k + ha_{21}f(t_k,y_k)) + O(h^3)\Big] =$$

$$b_1 f(t_k,y_k) + b_2\Big[f(t_k,y_k) + a_{21}hf(t_k,y_k)f_y(t_k,y_k) +$$

$$((a_{21}h)^2/2)f(t_k,y_k)^2 f_{yy}(t_k,y_k) + O(h^3) +$$

$$a_{21}h\Big[f_t(t_k,y_k) + a_{21}h\, f(t_k,y_k)f_{ty}(t_k,y_k) + O(h^3)\Big] +$$

$$((a_{21}h)^2/2)\Big[f_{tt}(t_k,y_k) + O(h^3)\Big]\Big] = (b_1 + b_2)f(t_k,y_k) +$$

$$b_2h\Big[a_{21}f(t_k,y_k)f_y(t_k,y_k) + a_{21}f_t(t_k,y_k)\Big] +$$

$$(b_2h^2/2)\Big[(a_{2k}f(t_k,y_k))^2 f_{yy}(t_k,y_k) + 2a_{21}^2 f(t_k,y_k)f_{ty}(t_k,y_k)$$

$$+ a_{21}^2 f_{tt}(t_k,y_k)\Big] + O(h^3).$$

A continuación, igualamos el desarrollo anterior con el siguiente desarrollo de Taylor:

$$R_t = f(t_k,y_k) + (h/2)f_t(t_k,y_k) + (h^2/6)f_{tt}(t_k,y_k) + O(h^3) =$$

$$f(t_k,y_k) + (h/2)(f_t + f_y f) + (h^2/6)\Big[f(t_k,y_k)^2 f_{yy}(t_k,y_k) +$$

Para definir un método de 2 etapas con parámetros numéricos daremos los valores que cumplen el sistema de ecuaciones anterior de la siguiente manera, $a_{21} = 1$, $b_1 = b_2 = 1/2$, para obtener finalmente:

$$R(t_k, y_k) = (1/2)(f(t_k, y_k) + f(t_k + h\ ,\ y_k + hf(t_k, y_k)))$$

Es claro ver que el metodo de Euler es menos preciso que el método anterior, ya que solo se esta utilizando un punto para estimar la pendiente promedio, mientras que para Runge-Kutta se utilizan varios puntos a lo largo del paso, los cuales tienen distintas ponderaciónes. Para que el método de Euler sea confiable requerimos que se cumpla al menos una de las siguientes dos condiciones:

a) Que la función "f" que representa la pendiente de "y" mantenga un crecimiento moderado, para obtener una "precisión" mejor.

b) Que la h sea lo suficientemente "pequeña" para prevenir cambios bruscos de $f(t, y)$.

Mediante Euler buscamos acercarnos al valor deseado y_n mediante la construcción de una serie de rectas construidas con iteraciones. Desafortunadamente no conocemos el comportamiento de la función $f(t, y)$ ni que tan rápido se incrementa, y al buscar acercarnos a este conocimiento por el método de Euler implica poca precision con respecto de otros.

$$
\begin{array}{c|cccccc}
0 & & & & & & \\
c_2 & a_{21} & & & & & \\
c_3 & a_{31} & a_{32} & & & & \\
\vdots & \vdots & \vdots & & & & \\
c_s & a_{s1} & \cdots & a_{s2} & \cdots & a_{s,s-1} & 0 \\
\hline
1 & a_{s+1,1} & a_{s+1,2} & \cdots & a_{s+1,s-1} & a_{s+1,s} & \\
\hline
 & b_1 & b_2 & & b_{s-1} & b_s & b_{s+1} \\
 & [d_1 & d_2 & & d_{s-1} & d_s & d_{s+1}]
\end{array}
\qquad \left\{ c_i = \sum_{j=1}^{s} a_{ij} \right\}
$$

y donde $d_i = b_i - a_{s+1,i}$

Aquí estamos combinando 2 métodos Runge-Kutta, y donde $a_{s+1,1} = b_1'$, $a_{s+1,2} = b_2'$, ..., $a_{s+1,s} = b_s'$, son los b_i's para el método Runge-Kutta de "s" etapas y b_1, b_2,..... b_{s+1} son los de RK de s+1 etapas, es decir, que para el método de mayor número de etapas utilizamos los parámetros del método de menos etapas. Todo esto con el objeto de ahorrarnos el cálculo de algunos parámetros; y por supuesto cumpliendo con las condiciones del sistema de ecuaciones.

Cuando representamos 2 métodos en una gráfica significa que comparten valores de parámetros, entonces:

RK-12 encajado queda como:

$$
\begin{array}{c|cc}
0 & & \\
1 & 1 & 0 \\
\hline
 & 1/2 & 1/2 \\
 & [-1/2) & [-1/2]
\end{array}
$$

Esto significa que tomamos $b_1' = a_{21}$

Con lo anterior nos ahorramos muchos pasos ya que a_{ij} es igual para RK de "s" etapas y para RK de "s+1" etapas solo requerimos calcular $f(Y_i)$ (donde i= 1, 2, ... s) y esto incluso es útil en la ecuación $f(Y_{s+1})$ por estar incluidas ahi.

Esto significa un gran ahorro de evaluaciones del lado derecho de el método numérico, principalmente cuando los métodos Runge-Kutta son de más de 2 etapas.

2.2 METODOS RUNGE-KUTA EXPLICITOS DE 1 Y 2 ETAPAS.

Como ya mencionamos, los métodos que utilizaremos en este trabajo son los de Runge-Kutta de etapas 1 y 2 explicitos. Del primero de ellos y de acuerdo a la formula general de Runge-Kutta es de la forma:

$$q_k = q_{k-1} + hb_1 f(Y_1) \quad \text{con } Y_1 = (t_{k-1}, q_{k-1}) \qquad (R = b_1 f)$$

e igualando "R" con la expansión de Taylor (de hecho solo con el primer término el cual es (t_{k-1}, q_{k-1})) llegamos a que $b_1 = 1$ con lo que obtenemos la ecuación:

$q_k = q_{k-1} + hf(t_{k-1}, q_{k-1})$ la cual es el conocido método de Euler.

decir, la pendiente estimada en el punto (t_k, y_k), y la pendiente estimada en el punto (t_{k+1}, y_{k+1}) también por Euler (figura 2.4). Este método al cálcular la pendiente mediante el promedio de dos pendientes estimadas con base en un mismo número de puntos dentro de la trayectoria de $y(t)$ nos permite reducir un posible error por cambios impredecibles en la pendiente al "repartir la responsabilidad" entre más de uno. Conforme utilizamos un método Runge-Kutta de más etapas obtenemos una estimación que fué obtenida mediante el promedio de más pendientes en distintos puntos de $y(t)$ y con ponderaciones distintas

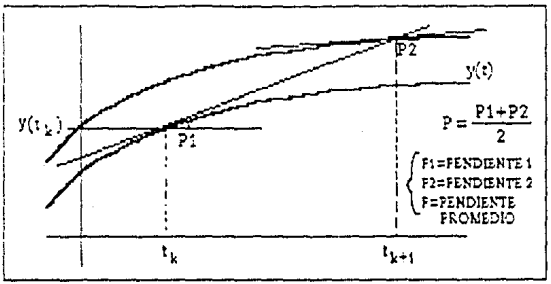

Figura 2.4.- Estimación de un punto mediante Runge-Kutta de 2 etapas.

2.3.1 ORDEN.

Si tomamos el error $E_{k+1} = y(t_{k+1}) - y_{k+1}$ donde

$y(t_{k+1})$ es el valor real y

[2]. En el trancurso de este trabajo el error global nos servirá para ver el comportamiento de la ecuación de prueba.

2.3.3 ERROR LOCAL.

Cuando avanzamos un paso "h", a partir de un punto inicial $y(t_k)$ conocido a $y_{k+1} = y(t_k) + h*R(t_k, y(t_k))$, estamos generando una aproximación que debida a distintos errores como truncamiento o algún otro tipo de limitante no será igual al valor real; el error generado, $(|y_{k+1} - y(t_{k+1})|)$ se llama error local. Si generamos el siguiente paso y_{k+2}, el error resultante $(|y_{k+2} - y(t_{k+2})|)$ ya no sería local, ya que esta contando el error local anterior, el cual se acumula e interfiere para incrementar el error no solo aditivamente ya que puede influir de otras formas. Entonces, para obtener el error local en este segundo paso requerimos en lugar de $|y_{k+1} - y(t_{k+1})|$ a $(|y_{k+1} - u(t_{k+1})|)$ donde la funcion "u" es tal que:

$$u' = f(t,u)$$

$$u(t_{k+1}) = y_{k+1}$$

Es decir, esta última función "u" es la solución a la ecuación diferencial con punto inicial $u(t_{k+1}) = y_{k+1}$. Esto se puede ver más claramente en la figura 2.5.

error local, el cual puede llegar a ser muy pequeño, tenemos que el global sigue aumentando rápidamente.

En el caso de que nuestras curvas solución se acerquen cuando "t" crece y si el error local es pequeño, estaremos controlando el error global, ya que el cambiar de una curva solución a otra curva solución no evitará que se llegue al valor correcto con precisión satisfactoria.

Debido a lo anterior, no importa que tan eficaz sea un método numérico, ya que como siempre hay pequeñas imprecisiones, estos brincos aunque sean relativamente chicos nos llevan a ecuaciones vecinas que se distanciarán irremediablemente cuando la ecuación sea inestable, por lo que al encontrarnos con que la ecuación diferencial presenta esta característica, debemos desistir de estimar su solución o al menos restarle credibilidad al resultado obtenido. En cambio nuestra metodología será util y valdrá la pena el esfuerzo por mejorar la estimación por cualquier recurso cuando la ecuación sea estable, y es aquí donde nuestra metodología va a entrar en acción.

En las siguientes dos figuras (2.6 y 2.7) se muestran las tendencias que siguen una ecuación inestable y otra estable respectivamente, de lo cual redundaremos un poco más cuando al presentar resultados en el capítulo 5 se muestre el comportamiento de los errores global y local para estos dos tipos de ecuaciones al utilizar la ecuación de "prueba".

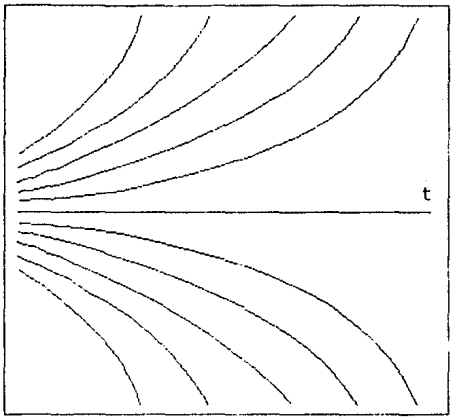

Figura 2.6.- Divergencia en una ecuación inestable.

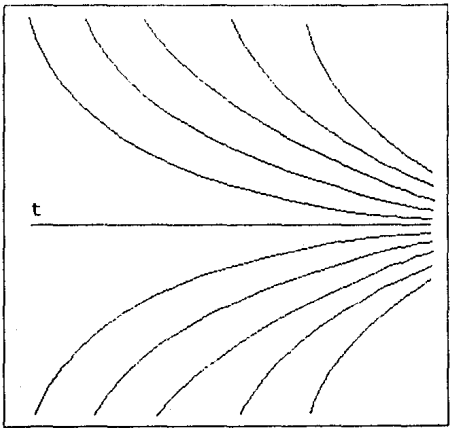

Figura 2.7.- Convergencia en una ecuación estable.

2.3.5 ECUACION DE PRUEBA.

Para obtener el punto $y(t_k)$ de la ecuación diferencial $y' = f(t,y)$ con valor inicial $y(t_0) = y_0$ por el método de Euler, construimos la recta $y(t)$ mediante pequeños segmentos de recta, los cuales vamos uniendo (figura 2.8) para seguir la supuesta forma de la funcion solución. Esto implica a su vez que $f(t,y)$ también está siendo representada linealmente por segmentos, ya que:

$$(y_n - y_{n-1})/h = f(t_{n-1}, y_{n-1})$$

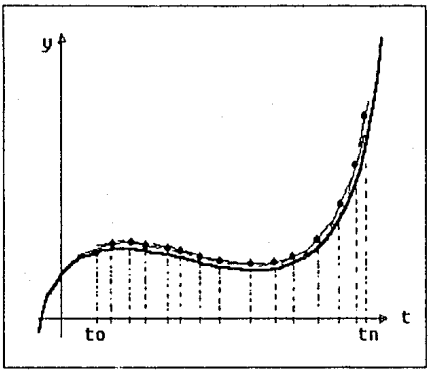

Figura 2.8.- Aproximacion numérica a $y(t)$ mediante segmentos de recta.

Los siguientes ejemplos nos muestran como utilizaremos este resultado:

Tomando la ecuación de prueba:

$$y' = \lambda y = f \Rightarrow f_y = \lambda \quad y \quad f_t = 0$$

$$f(y) = f(y_0) + f_y(y_0)(y - y_0) = \lambda y_0 + \lambda(y - y_0)$$

Por lo tanto λ juega el papel de estabilidad antes mencionado (positivo implica inestabilidad y negativo estabilidad).

Para la ecuación diferencial:

$$y' = y^2 = f \Rightarrow f_y = 2y \quad y \quad f_t = 0$$

$$y' = f(y_0) + f_y(y_0)(y - y_0) = y_0^2 + 2y_0(y - y_0)$$

$$\Rightarrow \quad 2y_0 = \lambda$$

$\Rightarrow$ Para $y_0 < 0$ es estable

Para $y_0 > 0$ es inestable

En los 2 ejemplos anteriores, la ecuación diferencial es estable en un rango e inestable en otro. Esto no siempre sucede como en el siguiente ejemplo:

$$y = \log(t)$$

$$y' = 1/t = 1/e^{y} = f(y)$$

$$f_y = -1/e^{y} \quad y \quad f_t = 0$$

$$f(y) = f(y_0) + f_y(y_0)(y - y_0) = 1/e^{y0} - 1/e^{y0}(y - y_0) =$$

$$((1 + y_0)/e^{y0}) - (1/e^{y0})y$$

$$\Rightarrow \lambda = -1/e^{y0} \Rightarrow \lambda < 0 \quad \therefore \text{siempre hay estabilidad.}$$

En los ejemplos anteriores $f_t = 0$, pero en lo que respecta a una función con $f_t \neq 0$ y $f_y \neq 0$ para averiguar si es o no estable requerimos de una ecuación que será resuelta en el siguiente tema y que resulta más general que lo anterior.

2.3.6 REGION DE ESTABILIDAD ABSOLUTA.

En cada paso de nuestra ecuación $y_{k+1} = y_k + hR(t_k, y_k)$ se introducen los errores de truncamiento y redondeo, lo cual perturbará

nuestro resultado final; la estabilidad absoluta de una ecuación diferencial con una "h" determinada, consiste en que dada una perturbación "γ" en una "y_i" donde $i \in \{1, 2, \ldots, n\}$, el cambio provocado no será mayor de esa misma "γ" para los subsecuentes valores de esa y_i.

Como ejemplo obtendremos el intervalo de estabilidad absoluta del método de Euler para la ecuación de prueba:

Utilizando $y' = f(t,y) = \lambda y$

y sea $y_{k+1} = y_k + hf(t_k, y_k) \Rightarrow y_{k+1} = y_k + h\lambda y_k = (1 + \lambda h)y_k$

$\therefore y_{k+1} = (1 + \lambda h)y_k$

Entonces, como:

$y_{k+1} = (1 + \lambda h)y_k$ sea $y_k + \xi = \tilde{y}_k$ = Perturbación en "y_k".

Sea ξ una perturbación tal que $\xi \in \mathbb{R}$ y $\xi \neq 0$.

$\Rightarrow \tilde{y}_{k+1} = (1 + \lambda h)\tilde{y}_k = (1 + \lambda h)(y_k + \xi)$ = Perturbación en y_{k+1} debida a la perturbación en y_k.

$\therefore |y_{k+1} - \tilde{y}_{k+1}| \leq |y_k - \tilde{y}_k|$

$$\Rightarrow |(1+\lambda h)(y_k + \xi) - (1 + \lambda h)y_k| \leq y_k - (y_k + \xi)|$$

y por definición de estabilidad absoluta:

$$|(1 + \lambda h)\left[y_k + \xi - y_k\right]| \leq |-\xi|$$

$$\Rightarrow |(1 + \lambda h)\xi| \leq |\xi|$$

$$\therefore | 1 + \lambda h ||\xi| \leq |\xi|$$

$$\Rightarrow | 1 + \lambda h| \leq 1 = q(z)$$

Y esta es la región de estabilidad absoluta de Euler, lo que significa que para los valores de λ y h que cumplan la desigualdad tendremos estabilidad en la ecuación diferencial.

Para nuestro caso particular $y' = \lambda y$ tenemos:

$$|1 + \lambda h| \leq 1$$

Por lo que:

$$-1 \leq 1 + h\lambda \leq 1$$

$$-2 \leq h\lambda \leq 0$$

Modificando y_{k-1} llegamos a $\tilde{y}_{k-1} = y_{k-1} + \xi$ donde ξ es nuestra "perturbación" de y_{k-1}; esto provoca otra perturbación, ahora en y_k, a la cual llamaremos $\tilde{y}_k$ y definiremos como:

$$\tilde{y}_k = \tilde{y}_{k-1} + hf(t_{k-1}, \tilde{y}_{k-1}) = y_{k-1} + \xi + hf(t_{k-1}, y_{k-1} + \xi)$$

Es decir, $\tilde{y}_k$ es el resultado de aplicar una perturbación a y_{k-1} ; como requerimos que $|\tilde{y}_k - y_k| \leq |\tilde{y}_{k-1} - y_{k-1}|$ entonces :

$$|\tilde{y}_k - y_k| =$$

$$|y_{k-1} + \xi + h\, f(t_{k-1}, y_{k-1} + \xi) - (y_{k-1} + hf(t_{k-1}, y_{k-1}))| =$$

$$|\xi + h\left[f(t_{k-1}, y_{k-1} + \xi) - f(t_{k-1}, y_{k-1})\right]| \leq$$

$$|\tilde{y}_{k-1} - y_{k-1}| = |y_{k-1} + \xi - y_{k-1}| = |\xi|$$

$$\therefore |\xi + h\left[f(t_{k-1}, y_{k-1} + \xi) - f(t_{k-1}, y_{k-1})\right]| \leq |\xi|$$

Por el teorema del valor medio tenemos que :

$$|f(t_{k-1}, y_{k-1} + \xi) - f(t_{k-1}, y_{k-1})| =$$

$$|\xi \partial f(\beta)/\partial y| = |\xi f_y(t_{k-1}, \beta)|$$

donde $y_{k-1} \leq \beta \leq y_{k-1} + \xi$

$$|\xi\left[1+hf_y(t_{k-1},\beta)\right]| \leq |\xi| \Rightarrow \boxed{|1+hf_y(t_{k-1},\beta)| \leq 1}$$

Este resultado será utilizado en el capitulo 5 cuando usemos el intervalo de estabilidad absoluta del método de Euler en relación con las ecuaciones diferenciales a las que aplicaremos el programa.

3.3.7 METODO NUMERICO PARA SISTEMAS DE ECUACIONES.

El hecho de que no nos hemos metido más que con ecuaciones de grado uno es debido, como se sabe, a que toda ecuación diferencial de grado "n" puede ser representada como "n" ecuaciones diferenciales de grado uno.

A continuación, además de mostrar lo anterior, se verá el método numérico para resolver este sistema de "n" ecuaciones de grado 1.

Sea $y^{(r)}$ = r-ésima derivada de "y".

Dada la ecuación $y^{(n)} = f(t, y^{(1)}, y^{(2)}, y^{(3)}, \ldots, y^{(n-1)})$ con valores iniciales:

$$X' = A*X + B = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 1 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 1 & 0 & \dots & 0 \\ \vdots & & & & & & \\ \beta 0 & \beta 1 & \beta 2 & \beta 3 & \beta 4 & \dots & \beta(n-1) \end{bmatrix} * \begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ \vdots \\ X_{n-1} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ -q(t) \end{bmatrix}$$

Donde $\beta 0 = -a_0/a_n \quad \beta 1 = -a_1/a_n \quad \dots \quad \beta(n-1) = -a_{n-1}/a_n$

Ya que $y^{(n)} = \partial X_{n-1}/\partial t =$

$$\{-a_{n-1}X_{n-1} - a_{n-2}X_{n-2} - \dots - a_1X_1 - ay_0 - q(t)\}/a_n$$

Donde $a_n \partial^n y/\partial t + a_{n-1}\partial^{n-1} y/\partial t + \dots + a_0 y + q(t) = 0$

y suponiendo que $q(t)=0 \Rightarrow X' = A*X$

Mientras que en el caso no lineal nos quedan por resolver las "n" ecuaciones diferenciales de grado 1:

$$\begin{bmatrix} X_k^1 \\ X_k^2 \\ \vdots \\ X_k^n \end{bmatrix} = \begin{bmatrix} X_{k-1}^1 \\ X_{k-1}^2 \\ \vdots \\ X_{k-1}^n \end{bmatrix} + h \begin{bmatrix} f_1(t_{k-1}, X_{k-1}^1, X_{k-1}^2, \ldots, X_{k-1}^n) \\ f_2(t_{k-1}, X_{k-1}^1, X_{k-1}^2, \ldots, X_{k-1}^n) \\ \vdots \\ f_n(t_{k-1}, X_{k-1}^1, X_{k-1}^2, \ldots, X_{k-1}^n) \end{bmatrix}$$

Entonces:

$$X_k^1 = X_{k-1}^1 + hf_1(t_{k-1}, X_{k-1}^1, \ldots, X_{k-1}^n)$$

$$\vdots$$

$$X_k^n = X_{k-1}^n + hf_n(t_{k-1}, X_{k-1}^1, \ldots, X_{k-1}^n)$$

Por necesidades del sistema de ecuaciones diferenciales, se resuelven de esta manera, una iteración para cada uno "n" veces. Entonces tenemos finalmente que nuestro afán de buscar y mejorar soluciones numéricas de ecuaciones diferenciales de grado 1, no esta limitado a estas, ya que también estamos abarcando las de grado "n" no lineales y lineales.

EULER EXPLICITO CON PASO VARIABLE

3.1 ESTIMACION DEL ERROR LOCAL.

Ahora, utilizando lo visto anteriormente obtendremos lo que se podria considerar la base de los métodos de estimación local, los cuales requieren del uso de 2 métodos numéricos de distinto orden (en nuestro caso estos son de órdenes uno y dos). Veamos ahora que sucede si utilizamos estos 2 métodos numéricos para estimar el mismo y_{k+1}.

Dado que $R = R(t_k, y_k)$ y $\tilde{R} = R(t_k, \tilde{y}_k)$ son las metodologías para estimar la pendiente promedio de $y(t)$ entre los puntos t_k y t_{k+1} por los métodos 1 y 2 respectivamente y sin considerar otro error más que el de truncamiento:

$$u(t_{k+1}) = y_k + hR + h\mu_k = y_{k+1} + h\mu_k \quad \text{por el método 1 de } O(h^1)$$

$$u(t_{k+1}) = \tilde{y}_k + h\tilde{R} + h\tilde{\mu}_k = \tilde{y}_{k+1} + h\tilde{\mu}_k \quad \text{por el método 2 de } O(h^2)$$

$$\therefore \text{Error local} = \tilde{y}_{k+1} - y_{k+1} = -h\tilde{\mu}_k + u(t_{k+1}) - [-h\mu_k + u(t_{k+1})] =$$

$$h\mu_k - h\tilde{\mu}_k = h\mu_k + O(h^2)$$

Ahora consideramos que, como el orden de $\tilde{\mu}_k$ es mayor que el de μ_k, tiende más rápido a cero (esto es más claro cuando el orden es grande) por lo que finalmente:

$$\tilde{y}_{k+1} - y_{k+1} \approx h\mu_k$$

Lo que nos dice que el error local o la diferencia de 2 estimaciones de $y(t_{k+1})$ por métodos con ordenes uno y dos, es aproximadamente igual a "h" veces el error de truncación del método de orden menor en el punto t_{k+1}.

Con esto tenemos una estimación del error local lo cual nos invita a mejorar el método numérico añadiendo esta aproximación; esto es llamado extrapolación local y se realiza mediante el simple uso de 2 métodos numéricos, uno como base y el otro para que conjuntamente con el primero nos de el parámetro (error local) estimado.

El siguiente metodo, desarrollado por Roland England, ha resultado muy importante en la busqueda de resultados numéricos debido a varios factores que considera paso a paso; entre ellos el hecho de que cada función requiere de distinto número de pasos ya que unas son más "suaves" que otras. Las 2 funciones que utilizaremos para desarrollar el metodo que considera el error son las de RK-1 y RK2 (Donde RK-1 es tambien llamado Euler).

que depende de $|y|$ y por lo tanto es deseable que la magnitud de $|e_L|/|y|$ sea muy chico (sin importar la magnitud de $|e_L|$). Para esto es deseable tomar una referencia "γ" tal que $|e_L|/|y| \leq \gamma$. Esta referencia es llamada tolerancia (Tol). Es recomendable que $\gamma \leq 0.01$; este valor ha sido obtenido por experimentación. La figura (3.1) ilustra lo anterior.

De esta forma si $|e_L|/|y| > \gamma$ entonces rechazamos "h" y recalculamos y_n mediante una "h" ajustada.

Existe una pequeña dificultad en cuanto a la obtención de "y", ya que no podemos obtenerla con su real magnitud, sino solo una aproximación; sin embargo podemos tratar de mejorar esa aproximación, tomando "$\tilde{y}_n$" y "y_n" e incluso "$y_{n/2}$" y "$\tilde{y}_{n/2}$", podemos entonces tomar $y \approx wt = (1/2)(y_n+(1/2)(y_{n/2}+\tilde{y}_{n/2}))$. De esta manera si $wt = 0 \Rightarrow y_n$, $y_{n/2}$, $y_{n/2}$ y $\tilde{y}_{n/2}$ son cero. Aún así, especial cuidado será tomado en caso de que $wt = 0$.

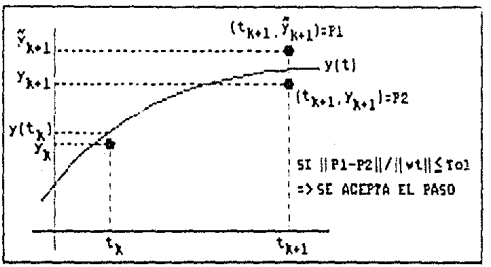

Figura 3.1.- Rechazo o aceptación del tamaño de paso.

y de aquí tenemos que $|H^2\,\Phi(t_k,y_k)|/|wt| \approx$

$$|\alpha^2 h^2 \Phi(t_k,y_k)|/|wt| \approx |\alpha^2(\tilde{y}_k - y_k)|/|wt| \approx \gamma$$

$$\therefore |H^2/h^2| \approx \gamma/((\tilde{y}_k - y_k)/wt) \quad \Rightarrow \quad H \approx h(\gamma/((\tilde{y}_k - y_k)/wt))^{(1/2)}$$

En general:

$$H = h\left(\sqrt[\theta]{\frac{\gamma}{(\,(\,\tilde{y}_k - y_k)/wt)}}\right) \qquad \text{donde} \quad \theta = \text{orden del error}$$

De aquí, tenemos que el tamaño de paso "H" es supuestamente el mayor paso posible, sin embargo, por efectos de aproximación tal vez se pase de ser el tamaño ideal a ser un paso mayor del requerido, por ser una "H" calculada aproximadamente y no saberse si se encuentra en la frontera o de que lado de ella por haberse calculado el límite superior de "H". Mediante la práctica que se ha determinado que el elemento γ sea multiplicado por .81, lo que equivaldría a obtener 9/10 de la supuesta "H" óptima; el significado de tomar una proporción del valor obtenido de "H" es el de no creerle del todo (no se ha podido obtener lo que sería la proporción óptima de "H", sin embargo en la práctica esta proporción de 9/10 ha funcionado).

3.3 DESCRIPCION DEL ALGORITMO DE PASO VARIABLE.

A continuación se dará una descripción del algoritmo que desarrollé basado en lo visto anteriormente. Esta descripción pretende explicar el proceso que se llevará a cabo ordenadamente a partir de los datos iniciales proporcionados por el usuario; después se dará el listado del programa para continuar con una descripción de cada procedimiento; es decir, indicar que parte del algoritmo se encargará de cada proceso.

El algoritmo inicia con los siguientes puntos indispensables:

- Presentación del programa.
- Limpia de todas la variables.
- Reconocimiento de las condiciones del equipo en uso.
- Requerimiento al usuario de los parámetros iniciales t_0, $y(t_0)$, H inicial, límite superior para "H" (si se desea), número de particiones (si se desean). Las particiones consisten en la división del intervalo $t-t_0$ en "N" partes y presentar el valor $y(t_0+k((t-t_0)/n))$ para k = 1, 2, ..., N-1, N; esto con la finalidad de mostrar los movimientos que $y(t)$ hizo durante el proceso.

Después de los anteriores movimientos, se inicia un proceso que construye "$y(t)$"; este proceso se repetirá mientras el usuario desee pedir más $y(t)$'s para distintos valores de "t"; es así que el proceso llama al procedimiento que pide los elementos iniciales al usuario y

continúa así con la construcción del nuevo $y(t)$ estimado.

El procedimiento de construcción de $y(t)$ continúa en funcionamiento mientras no se cumplan alguna de 2 condiciones:

- El tamaño de paso requerido es muy chico; tanto, que es considerado cero por el proceso.
- Se terminó de generar $y(t)$.

El procedimiento requiere de otros procedimientos a los cuales solicita conforme los va requiriendo. El proceso completo de creación y prueba del paso es considerada una iteración; en ejemplos ejecutados se requirieron desde cientos hasta millones de iteraciones. Factores importantes que implican el número de iteraciones son la magnitud $|t-t_0|$ y la suavidad de la función $y(t)$ o alguna de las funciones del sistema. El factor principal es la suavidad de la función, ya que en muchas ocaciones la "H" manipulada durante el proceso es un número real muy chico. Por lo general este tipo de funciones es no estable.

El algoritmo funciona de la siguiente manera:

A) Dados los valores actuales t_L, $y(t_L)$ y "H" (la primera vez que se llama a esta opción el valor t_L es igual a t_0). Para cada función del sistema, se genera $f_L(t_L,y_L)$ mediante la simple substitución de los valores t_L y y_L en la función. A partir de cada $f_L(t_L,y_L)$ y con las funciones Runge Kutta de órdenes 1 y 2, obtenemos los respectivos

valores aproximados de "y" y "y'" restando $y'-y$ obtenemos el vector de errores locales.

B) Obtenemos $\|\omega\tau\|$ en base a un promedio de magnitudes de "y" y "y'" para paso completo y medio paso. Este promedio se definió como:

$\|\omega\tau\|$=(1/8)(Abs(YH2)+Abs(YTILDEH2)+3(Abs(Y)+Abs(YTILDE)))

Este promedio da mayor ponderación a Y y Ytilde, los cuales son de paso completo, mientras que YH2 y YtildeH2 representan a "y" y "y'" pero con medio paso.

C) Ahora generamos otro arreglo, el de errores relativos; mediante (Error Local/$\omega\tau$) para cada elemento del sistema. Enseguida obtenemos la norma infinito de este arreglo; esta norma nos da el mayor elemento (considerando para todos su valor absoluto), y a este valor lo definimos Erl = error local. Mediante una comparación con la tolerancia decidimos si el paso es o no aceptado, esto se hace de la siguiente manera:

Si Erl ≤ Tol ⇒ se acepta el tamaño de paso.

Si Erl > Tol ⇒ se rechaza el tamaño de paso.

D) Si se acepto el tamaño de paso, entonces incrementamos t_L de la forma $t_L := t_L + H$ y se guardan los valores obtenidos de "y" y "H" en el punto aceptado.

E) Independientemente de si "H" fué o no aceptada se genera una nueva "H". La generación de "H" es un proceso que dependiendo del valor (Err/Tol) alarga o encoge "H" automáticamente. Más adelante se explicará este proceso detalladamente cuando se vea el listado del programa.

F) Si $t_L < t$ (t es el punto en el eje T del cual se solicita $y(t)$) entonces regresamos al inciso "A".

G) Si $t_L \neq t$ interpolamos para obtener $y(t)$, ya que solo tenemos $y(t_L)$ y $y(t_L - h)$ donde $t_{L-H} < t < t_L$

H) Termina el procedimiento y se imprimen los resultados.

Todo el mecanismo que se dió requiere de varios procedimientos individuales, los cuales desarrollan su parte con base en elementos ya vistos y que ya han sido explicados. Estos procedimientos serán explicados más a detalle después de presentarse el listado del programa; cada uno de esos procesos proviene de distintos materiales y aunque el algoritmo anterior parece muy simple, requiere en cada proceso de detalles y consideraciones a diversas problemáticas.

Dentro del desarrollo de este programa, existe una gran cantidad de contadores y variables que se encargan de dar estados del

funcionamiento durante y después de la generación de $y(t)$. Algunos de estos son:

- Contadores de pasos aceptados y rechazados.
- Variable que registra el mayor tamaño de paso aceptado.
- Promedio de paso "H" (solo para pasos aceptados).
- Tolerancia utilizada.
- Malla de datos durante el paso de t.

El último punto citado, el de la malla, consiste en presentar para valores del eje "Y" los puntos separados a distancias iguales Y(L) donde $t_0 \leq L \leq t$. El usuario decide cuantos puntos desea.

A continuación se presenta el diagrama de flujo del programa que se acaba de describir y que se presenta listado en su totalidad en el siguiente capítulo. Las funciones presentadas son las principales.

PROGRAMA DE EULER EXPLICITO CON PASO VARIABLE

A continuación se presenta un listado del programa, con una explicación del mismo abajo de cada procedimiento de importancia, con el fin de explicar el funcionamiento de cada parte del programa; después se explicara en forma general el funcionamiento para que aunado a la explicación individual de las partes se comprenda el total del mismo.

Dentro del programa se manejó un formato distinto al que hemos venido utilizando, esto es, dada la ecuación:

$y' = f(t, y_1, y_2, \ldots, y_n)$ en el caso de un sistema, y por razones de simplificación de formato en lugar de representar a "t" como una variable y a "y" como un arreglo, incluiremos a "t" dentro del arreglo que contiene a las "y's". Esto, en lugar de significar que representamos vectorialmente a "y'" implica un formato para simplificar el desarrollo del programa.

4.1 AMBIENTE DEL PROGRAMA.

El programa se elaboró en lenguaje Pascal, principalmente por razones de precisión, ya que las versiones actuales permiten utilizar 20 dígitos utilizando el correspondiente co-procesador matemático

8088. Durante la ejecución de este programa se utilizaron hasta 11 digitos ya que no se contó con el co-procesador, sin embargo una pequeña modificación a las declaraciones del programa y la instalación del co-procesador permitirán mejorar la precisión si se requiere. Dentro del ambiente necesario para ejecutar el programa están la memoria, de la cual el mínimo es de 256k, computadora XT ó AT con sistema operativo MS-DOS versión 3.0 o mayor y Turbo Pascal versión 4.0 o mayor.

4.2 PROGRAMA RK-12 CON PASO VARIABLE.

```
PROGRAM PASO_VARIABLE;
USES CRT,PRINTER;
CONST
  ORDEN=2;    { PARAMETRO QUE DETERMINA EL ORDEN DEL ERROR }
  N=1;        { TAMAÑO DEL SISTEMA DE ECUACIONES DIFERENCIALES }
  TOL=0.000001;    { TOLERANCIA UTILIZADA }
TYPE
  NODO=RECORD
   Y0,Y1,FY1,FY2,YTILDE,ERROR:REAL;
  END;
  AP=^NODO2;
  NODO2=RECORD
   MALLA:ARRAY[0..N] OF REAL;
   COLA:AP;
  END;
```

```
ARREGLO=ARRAY[0..N] OF NODO;
ARREGLO2=ARRAY[0..N] OF REAL;
ARREGLO3=ARRAY[1..N] OF REAL;

 VAR
  GUARDAMALLA            :AP;
  ERL,H,HINICIAL,
  BETA,T,MATCHEPS,
  HSUPERIOR,
  HMAYORUTILIZADA,
  TAMANIODELINTERVALO,
  PUNTOAESPERAR          :REAL;
  I,PASOSACEPTADOS,
  PASOSRECHAZADOS        :LONGINT;
  VECTOR                 :ARREGLO;
  ERLARRAY,
  YH2,YTILDEH2           :ARREGLO3;
  FY                     :ARRAY[1..N] OF REAL;
  YOANT,VECTORARMADO,
  GUARDADATOS            :ARREGLO2;
  TERMINA,HMUYCHICA,
  SELLEGOALTOPE          :BOOLEAN;
  FIN,PREG,PREG2         :CHAR;
  SIGNO                  :INTEGER;
```

```
{ LOS 2 PROCEDIMIENTOS SIGUIENTES REALIZAN CAMBIOS EN EL VIDEO }

  PROCEDURE VIDEONORMAL;
  BEGIN
   TEXTCOLOR(15);
   TEXTBACKGROUND(0);
  END;


  PROCEDURE VIDEOINVERSO;
  BEGIN
   TEXTCOLOR(0);
   TEXTBACKGROUND(15);
  END;


  PROCEDURE LEETYH;
  VAR
     PARTICIONES  :INTEGER;
   BEGIN
    CLRSCR;
    SIGNO:=1;
    HMUYCRICA:=FALSE;
    NEW(GUARDAMALLA);
    GUARDAMALLA^.MALLA[0]:=0;
    GUARDAMALLA^.COLA:=NIL;
    SELLEGOALTOPE:=FALSE;
```

```
T:=VECTOR[0].YO;
FOR I:=1 TO 3 DO
WRITELN(' ');
WHILE T=VECTOR[0].YO DO
 BEGIN
  VIDEONORMAL;
  GOTOXY(WHEREX,WHEREY-1);
  WRITE('LIMITE SUPERIOR DE T : ');
  VIDEOINVERSO;
  WRITE('   ');
  GOTOXY(WHEREX-3,WHEREY);
  READLN(T);
 END;
IF T<VECTOR[0].YO THEN
SIGNO:=-1;
H:=-1;
WRITELN(' ');
WHILE H<=0 DO
BEGIN
 GOTOXY(WHEREX,WHEREY-1);
 VIDEONORMAL;
 WRITE('H INICIAL : ');
 VIDEOINVERSO;
 WRITE('   ');
 GOTOXY(WHEREX-3,WHEREY);
 READLN(H);
END;
```

```
PREG2:='N';
HMAYORUTILIZADA:=0;
VIDEONORMAL;
WRITELN(' ');
WRITE('DESEA LIMITE SUPERIOR PARA "H" ? (S/N) ');
VIDEOINVERSO;
WRITE(' ');
GOTOXY(WHEREX-1,WHEREY);
PREG2:=READKEY;
VIDEONORMAL;
HSUPERIOR:=0;
WRITELN(' ');
WRITELN(' ');
IF UPCASE(PREG2)='S' THEN
WHILE HSUPERIOR<=H DO
BEGIN
 VIDEONORMAL;
 GOTOXY(WHEREX,WHEREY-1);
 WRITE('DEME LIMITE SUPERIOR DE "H" : ');
 WRITE('                    ');
 GOTOXY(WHEREX-20,WHEREY);
 VIDEOINVERSO;
 WRITE('   ');
 GOTOXY(WHEREX-3,WHEREY);
 READLN(HSUPERIOR);
END
ELSE
```

```
HSUPERIOR:=1.0E+38;
HINICIAL:=H;
VIDEONORMAL;
WRITELN(' ');
PREG:='N';
WRITE('DESEA PARTICIONES ? (S\N) :');
VIDEOINVERSO;
WRITE(' ');
GOTOXY(WHEREX-1,WHEREY);
PREG:=READKEY;
PREG:=UPCASE(PREG);
VIDEONORMAL;
WRITELN(' ');
IF PREG='S' THEN
 BEGIN
   WRITE('PARTICIONES DEL INTERVALO :');
   VIDEOINVERSO;
   WRITE('  ');
   GOTOXY(WHEREX-2,WHEREY);
   READLN(PARTICIONES);
   IF PARTICIONES<=1 THEN
   PREG:='N'
   ELSE
    BEGIN
      TAMANIODELINTERVALO:=ABS(T-VECTOR[0].Y0)/PARTICIONES;
      PUNTOAESPERAR:=VECTOR[0].Y0+SIGNO*TAMANIODELINTERVALO;
    END;
```

```
FOR I:=0 TO N DO
GUARDADATOS[I]:=VECTOR[I].YO;
   {                GUARDADATOS                }
   { Guarda los parámetros iniciales de t0,    }
   { y1(t0), ..., yn(t0) por si son reque-     }
   { ridos nuevamente, esto es en el caso      }
   { de que se de otro punto "t" para cal-     }
   { cular y(t)                                }

WRITELN(' ');
LEETYH;
END;


PROCEDURE FDY(Y:ARREGLO2);
{ Y[0]=T0, Y[1]=Y1, ..., Y[N]=YN }

 BEGIN

  FY[1]:=COS(Y[0])+10*(Y[1]-SIN(Y[0]));   { AQUI VA EL SISTEMA }

 END;


PROCEDURE RK12;
BEGIN
```

```
FDY(VECTORARMADO);
     { Genera la función con los parámetros conte-   }
     { nidos en vectorarmado, f(tk+h,yk+h*f(tk,yk)); }
     { Toma el valor de la función y lo pone en FY2  }

FOR I:=1 TO N DO
VECTOR[I].FY2:=FY[I];
FOR I:=1 TO N DO
WITH VECTOR[I] DO
 BEGIN
  YTILDEN2[I]:=YO+SIGNO*(H/4)*(FY1+FY2);
      { Runge-Kutta de orden 2 y medio paso}

  YTILDE:=YO+SIGNO*(H/2)*(FY1+FY2);
      { Runge-Kutta de orden 2 }
  ERROR:=YTILDE-Y1;
      { calculo del error local }
 END;
END; (* DEL PROCEDIMIENTO *)



FUNCTION NORMAINF:REAL;
VAR
 NORMA:REAL;
 AR:ARRAY[1..N] OF REAL;
 BEGIN
   NORMA:=ABS(ERLARRAY[1]);
```

```
   IF N>1 THEN
   FOR I:=2 TO N DO
   IF ABS(ERLARRAY[I])>NORMA' THEN
   NORMA:=ABS(ERLARRAY[I]);
   NORMAINF:=NORMA;

 END; (* FIN DEL PRECEDIMIENTO NORMAINF *)



FUNCTION FUNCIONAELPASO:BOOLEAN;
VAR
 WT:ARREGLO3;
       { En la variable WT calculamos la magnitud apro-    }
       { ximada de "y" real más que con exactitud, con la  }
       { intención de determinar su tamaño y usarlo en la  }
       { del error relativo, usando "y" y "y'" con pasos   }
       { h y h/2                                           }

BEGIN
  FOR I:=1 TO N DO
  WITH VECTOR[I] DO
  BEGIN
   WT[I]:=(1/8)*(ABS(YTILDEH2[I])+ABS(YH2[I])+
   3*(ABS(YTILDE)+ABS(Y1)));
   IF WT[I]=0 THEN ERLARRAY[I]:=ERROR
   ELSE
    ERLARRAY[I]:=ERROR/WT[I];
```

```
    END;
    ERL:=NORMAINF;
    IF ERL<=TOL THEN FUNCIONAELPASO:=TRUE
    ELSE
    BEGIN
      FUNCIONAELPASO:=FALSE;
      INC(PASOSRECHAZADOS);
         { No funciona el paso, entonces incrementamos }
         { el contador de pasos rechazados             }
    END;
  END;


PROCEDURE GENERAH;
 BEGIN
    IF ERL=0 THEN
    BETA:=EXP((1/ORDEN)*LN(10*(TOL/MATCHEPS)))
    ELSE
    BETA:=EXP((1/ORDEN)*LN(TOL/ERL));
    H:=(9/10)*H*BETA;
    IF H=0 THEN
    HMUYCHICA:=TRUE;
    IF H>HSUPERIOR THEN
     BEGIN
       H:=HSUPERIOR;
       SELLEGOALTOPE:=TRUE;
```

```
    GUARDAMALLA^.MALLA[I])
   END
   ELSE
   IF DATO=2 THEN
    BEGIN
     FOR I:=1 TO N DO
    IF GUARDAMALLA^.MALLA[0]<>0 THEN
     IF (GUARDAMALLA^.MALLA[I]>0.00000009) OR
     (GUARDAMALLA^.MALLA[I]<-0.00000009) THEN
     BEGIN
      WRITE(LST,'Y',I:2,'(',GUARDAMALLA^.MALLA[0]:11:8,'): ');
      WRITELN(GUARDAMALLA^.MALLA[I]:11:8);
     END
     ELSE
     BEGIN
       WRITE(LST,'Y',I:2,'(',GUARDAMALLA^.MALLA[0]:11:8,'): ');
       WRITELN(GUARDAMALLA^.MALLA[I])
     END;
    END;
  IF GUARDAMALLA^.COLA<>NIL THEN
  IMPRIMEMALLA(GUARDAMALLA^.COLA,DATO);
END;



PROCEDURE IMPRIMERESULTADOS;
 VAR
  TERM,RESPUESTA:CHAR;
```

```
IF PREG='S' THEN
IMPRIMEMALLA(GUARDAMALLA,1)
ELSE
FOR I:=1 TO N DO
BEGIN
    IF (VECTOR[I].Y1>0.00000009) OR
    (VECTOR[I].Y1<-0.00000009) THEN
    WRITE('Y (',I,'): ',VECTOR[I].Y1:11:8)
    ELSE WRITE('Y (',I,'): ',VECTOR[I].Y1);
    IF I<N THEN WRITELN(',')  ELSE WRITELN(' ');
END;
WRITELN('H INICIAL : ',HINICIAL:11:8);

WRITELN('H MAYOR UTILIZADA : ',HMAYORUTILIZADA:11:8);
IF UPCASE(PREG2)='S' THEN
IF SELLEGOALTOPE THEN
 WRITELN('SE LLEGO A LA H TOPE : ',HSUPERIOR:11:8)
ELSE
 WRITELN('NO SE REQUIRIO H TOPE ',HSUPERIOR:11:8);
IF (((T-GUARDADATOS[0])/PASOSACEPTADOS)>0.00000009)
OR (((T-GUARDADATOS[0])/PASOSACEPTADOS)<-0.00000009) THEN
BEGIN
  WRITE('PROMEDIO DE PASO H : ');
  WRITELN((ABS(T-GUARDADATOS[0])/PASOSACEPTADOS):11:8);
END
ELSE
```

```
  BEGIN
  WRITE('PROMEDIO DE PASO H : ');
  WRITELN(ABS(T-GUARDADATOS[0])/PASOSACEPTADOS);
  END;
  WRITELN('TOLERANCIA : ',TOL:11:8);
  WRITELN('# DE PASOS ACEPTADOS : ',PASOSACEPTADOS);
  WRITELN('# DE PASOS RECHAZADOS : ',PASOSRECHAZADOS);
  WRITE('# TOTAL DE PASOS : ');
  WRITELN(PASOSACEPTADOS+PASOSRECHAZADOS);
  WRITELN('==========================================');
  WRITELN(' ');
 END;

IF UPCASE(RESPUESTA)='I' THEN
 BEGIN
  FOR I:=1 TO 3 DO
  WRITELN(LST,' ');
  WRITELN(LST,'T INICIAL : ',GUARDADATOS[0]);
  WRITELN(LST,'CON VALORES DE Y :');
  FOR I:=1 TO N DO
  WRITE(LST,'Y',I:2,'(',GUARDADATOS[0]:11:8,') : ');
  WRITE(LST,GUARDADATOS[1]);
  WRITELN(LST,'EN EL PUNTO T= ',T:11:8);
  IF PREG='S' THEN
  IMPRIMEMALLA(GUARDAMALLA,2)
  ELSE
```

```
FOR I:=1 TO N DO
BEGIN
    IF (VECTOR[I].Y1>0.00000009) OR
    (VECTOR[I].Y1<-0.00000009) THEN
    WRITE(LST,'Y (',I,'): ',VECTOR[I].Y1:11:8)
    ELSE WRITE(LST,'Y (',I,'): ',VECTOR[I].Y1);
    IF I<N THEN WRITELN(LST,',')
    ELSE
    WRITELN(LST,' ');
END;
WRITELN(LST,'H INICIAL : ',HINICIAL:11:8);

WRITELN(LST,'H MAYOR UTILIZADA : ',HMAYORUTILIZADA:11:8);
IF UPCASE(PREG2)='S' THEN
IF SELLEGOALTOPE THEN
 WRITELN(LST,'SE LLEGO A LA H TOPE : ',HSUPERIOR:11:8)
ELSE
 WRITELN(LST,'NO SE REQUIRIO H TOPE ',HSUPERIOR:11:8);
IF (((T-GUARDADATOS[0])/PASOSACEPTADOS)>0.00000009)
OR (((T-GUARDADATOS[0])/PASOSACEPTADOS)<-0.00000009) THEN
BEGIN
 WRITE(LST,'PROMEDIO DE PASO H : ');
 WRITELN(LST,(ABS(T-GUARDADATOS[0])/PASOSACEPTADOS):11:8);
END
ELSE
BEGIN
 WRITE(LST,'PROMEDIO DE PASO H : ');
```

```
        WRITELN(ABS(T-GUARDADATOS[0])/PASOSACEPTADOS);
      END;
      WRITELN(LST,'TOLERANCIA : ',TOL:11:8);
      WRITELN(LST,'# DE PASOS ACEPTADOS : ',PASOSACEPTADOS);
      WRITELN(LST,'# DE PASOS RECHAZADOS : ',PASOSRECHAZADOS);
      WRITE(LST,'# TOTAL DE PASOS : ');
      WRITELN(LST,PASOSACEPTADOS+PASOSRECHAZADOS);
      WRITELN(LST,'========================================');
      WRITELN(LST,' ');
      WRITELN(LST,' ');
      WRITELN(' ');
      WRITELN(' ');
      WRITE('PULSE CUALQUIER TECLA PARA TERMINAR ...');
      TERM:=READKEY;
    END;

  END;  (* FIN DEL PROCEDIMIENTO IMPRIMERESULTADOS*)


PROCEDURE RED(GUARDAIMPRESION:AP);
  BEGIN
    GUARDAIMPRESION^.MALLA[0]:=PUNTOAESPERAR;
    FOR I:=1 TO N DO
    GUARDAIMPRESION^.MALLA[I]:=INTERPOLA(YOANT[0],
    VECTOR[0].YO,YOANT[I],VECTOR[I].Y1,PUNTOAESPERAR);
    PUNTOAESPERAR:=PUNTOAESPERAR+SIGNO*TAMANIODELINTERVALO;
    NEW(GUARDAIMPRESION^.COLA);
```

```
    IF FUNCIONAELPASO THEN
     BEGIN
       IF HMAYORUTILIZADA<H THEN
       HMAYORUTILIZADA:=H;
       FOR I:=0 TO N DO
       YOANT[I]:=VECTOR[I].YO;
       VECTOR[0].YO:=VECTOR[0].YO+SIGNO*H;
       FOR I:=1 TO N DO
       VECTOR[I].YO:=VECTOR[I].Y1;
       INC(PASOSACEPTADOS);
         { Si funciona el paso, entonces incremen- }
         { tamos el contador de pasos aceptados    }

     END;
   GENERAH;
  IF (PREG='S') AND (ABS(PUNTOAESPERAR)<=ABS(VECTOR[0].YO)) OR
  (ABS(VECTOR[0].YO) >= ABS(T) THEN
  MANDANODOSIGUENTEALARED(GUARDAMALLA);
   END;
IF PREG = 'S' THEN
WHILE ABS(PUNTOAESPERAR) < = ABS(T) DO
MANDANODOSIGUIENTEALARED(GUARDAMALLA);
IF (T<>VECTOR[0].YO) AND (NOT(HMUYCHICA)) AND (PREG='N') THEN
FOR I:=1 TO N DO
VECTOR[I].Y1:=INTERPOLA(YOANT[0],
VECTOR[0].YO,YOANT[I],VECTOR[I].Y1,T);
```

```
  IF NOT(HMUYCHICA) THEN
  IMPRIMERESULTADOS
   ELSE
    BEGIN
     CLRSCR;
     GOTOXY(1,10);
     SOUND(1500);
     VIDEOINVERSO;
     WRITE(' Lo siento, proceso interrumpido ');
     VIDEONORMAL;
     WRITELN;
     VIDEOINVERSO;
     WRITE(' por requerirse H<',matcheps);
     WRITE(' (epsilon de la maquina) ');
     VIDEONORMAL;
     DELAY(1500);
     NOSOUND;
     READLN;
    END;

  END; (* DEL PROCEDIMIENTO PRINCIPAL *)


PROCEDURE EPSILON;
 VAR
  EPSPOS,EPSANT:REAL;
```

```
BEGIN
 EPSPOS:=1;
 WHILE EPSPOS<>0 DO
  BEGIN
    EPSANT:=EPSPOS;
    EPSPOS:=EPSPOS/10;
    IF EPSPOS =0 THEN
    MATCHEPS:=EPSANT;
  END;
END;     { FIN DEL PROCEDIMIENTO EPSILON }


BEGIN     (* PROGRAMA PRINCIPAL *)

 INICIAL;
 TERMINA:=FALSE;
 EPSILON;
 WHILE NOT(TERMINA) DO
  BEGIN
    PASOSACEPTADOS:=0;
      { inicializamos en cero los contadores de }
    PASOSRECHAZADOS:=0;
      { pasos aceptados y pasos rechazados       }
    PRINCIPAL;
    WRITELN(' ');
    TERMINA:=TRUE;
    WRITE('DESEAS OBTENER OTRO PUNTO ? (S/N): ');
```

menor que t_0. Gracias a esto, a partir de $y(t_0)$ podemos buscar $y(t_j)$ y $y(t_i)$ donde $t_i \leq t_0 \leq t_j$.

Al principio, con la declaración de las constantes, se encuentra la línea Tol = .000001 la cual es la ya mencionada tolerancia; esto es, cuando calculamos $|e_L|/|y|$ para saber si debemos aceptar el tamaño del paso requerimos que la magnitud obtenida sea menor a una tolerancia predeterminada.

A continuación se dará una explicación general del funcionamiento del programa, para finalizar con una explicación más detallada de cada procedimiento del mismo, donde se explica el porque de cada función y como se resuelven ciertos problemas.

El programa esta estructurado en procedimientos individuales, los cuales realizan actividades específicas; unas de estas actividades son de recopilación de datos del usuario, como tamaño inicial del paso, punto donde se desea calcular la aproximación, número de particiones requeridas del intervalo, etc.

Otros procedimientos se encargan de controlar el video, colocar los elementos ordenados en la pantalla y de reconocer las condiciones en las que se trabaja (Matcheps). El programa principal se encarga de administrar cada procedimiento y dentro de cada uno de estos se llaman unos a otros de acuerdo a las necesidades y situaciones.

El papel primordial del programa es, mediante datos preestablecidos y que solo pueden ser modificados reescribiendo en las respectivas lineas del programa, calcular la aproximación de $y(t)$ dado un punto del eje t. Uno de los puntos preestablecidos anteriores es la linea donde se encuentra la funcion (procedimiento FDY). Esta función es una ecuación diferencial o sistema de ecuaciones diferenciales; más adelante cuando se explique en detalle el funcionamiento de este procedimiento, se explicará la forma en que se deberá de modificar.

Durante el proceso se hacen diversos calculos, en los que podria haber dificultades, por lo que se han agregado procesos que buscan evitar estos problemas que podrian consistir en divisiones sobre cero o pérdida de avance en los pasos por limitantes en la cantidad de digitos por la computadora. Por otra parte, se eliminó el pasar valores de variables por parámetro a los procedimientos, ya que esto provocaba un incremento excesivo en el tiempo de ejecucion. Es por esta razón que muchas variable son globales.

Un punto muy importante consiste en la presentación de datos obtenidos durante y al final del proceso respecto al resultado lo cual nos da una visión muy importante de la obtención de $y(t)$, estos datos son, el número de pasos aceptados y rechazados, tolerancia utilizada, tamaño de paso más grande aceptado, promedio de paso, etc; además de la presentación (si se requirió al principio del programa) de una malla de datos, es decir, el avance registrado de $y(t)$ para puntos anteriores a "t" (el número de puntos, separados a distancias iguales se solicita al principio del programa si se acepta la opción).

Si recordamos cuando obtuvimos la región de estabilidad, entonces nos daremos cuenta que, para que la función se encuentre dentro de la región de estabilidad, es necesario que el tamaño de paso no salga de un cierto rango; para esto el programa presenta una opción en la que no permite que, dado el valor de una "h", los valores de esta no sean mayores y en caso de que el programa tuviera la oportunidad de hacerlo, un proceso mantendría el tamaño de la "h" en el tope solicitado mientras ese tamaño de "h" fuera aceptado como tamaño de paso.

La parte primordial del programa, considera 2 puntos importantes:

1) Ajuste del tamaño de paso.

2) Cálculo del error de paso.

Ambos puntos ya han sido explicados y se encuentran ligados, ya que con el cálculo del error local (error de paso) obtenemos el error relativo y mediante la tolerancia decidimos si se acepta o no el paso y se crea el nuevo paso sin importar si fue o no aceptado (este paso se ajusta automáticamente respecto al resultado anterior). El proceso de alargamiento y encogimiento del paso es un proceso que se explicará cuando tratemos el procedimiento "Generall".

El programa principal se encarga de mantener el programa funcionando mientras el usuario tenga más datos que pedir; Además de

entrada poner a funcionar el procedimiento épsilon, el cual se encarga de averiguar cual es el número más pequeño manejable por la computadora que se está usando. Este último lo hace mediante el sencillo proceso de encoger un número inicial hasta que este desaparezca; de esta manera el número anterior a este desvanecimiento será el más pequeño. Más adelante un proceso se encargará de evitar que el programa continúe si la "h" requerida es más pequeña que el mínimo disponible llamado Matcheps.

Otro proceso importante del programa principal se encarga de guardar los datos iniciales (valores t_0, $y_i(t_0)$ para toda "i" del sistema) para volverlos a utilizar sin solicitarlos otra vez si son requeridos para calcular otro punto "t".

El procedimiento "Leetyh" pide los datos principales requeridos como tamaño inicial de "h", punto "t" (del cual se obtendrá $y(t)$), límite superior para "h" (si se desea), número de particiones (si se desea). Este procedimiento vuelve a utilizarse cada vez que se desea pedir otro $y(t)$, a diferencia del procedimiento "inicial" el cual pide una sola vez "t_0" y $y(t_0)$ ya sea para ecuación diferencial o sistema de ecuaciones diferenciales.

El procedimiento "FDY" requiere parámetros de entrada los cuales son los valores "t" y "y" de las "n" dimensiones del sistema y que servirán a este procedimiento para construir $f(t, y)$.

Suponiendo que tenemos un sistema con funciones:

adelante, en otro procedimiento se hara uso de estas "y's" adicionales; la razón principal es la de evitar caer en magnitudes de cero al tratar de obtener una $\|y\|_n$ que más que de precisión se requiere en magnitud.

El último paso consiste en calcular el error mediante la diferencia de los 2 métodos "y'-y". Esta diferencia, como ya se vió en los cálculos anteriores es una aproximación del error local.

Una opción importante que se puede tomar es la de modificar las líneas donde se calculan "y" y "y'"; estas actualmente son las aproximaciones por Runge-Kutta 1 y Runge-Kutta 2, pero podrían ser RK-23 ó RK-34, etc u otros métodos distintos de Runge-Kutta pero de grados distintos. En este trabajo se utilizaron solamente RK-12 para mostrar que con el RK más sencillo se obtienen buenos resultados por la estructura del proceso de este programa, ya que esto no es una búsqueda de una aproximación mediante un método RK de "x" etapas, sino que va más allá; es una metodología para calcular esa aproximación tomando en cuenta limitantes como suavidad de la función y precisión del método para calcular las "y's", llámese Runge-Kutta, Taylor o cual fuere. Si se llegaran a modificar las funciones "$y$tilde" ó "y", es necesario cambiar la constante "Global=2" a el valor de la función de mayor orden ya sea "$y$tilde" ó "y" (global es el orden del error).

"Normainf" es un procedimiento que solamente da la norma infinito de todos los valores que se encuentren en el arreglo "Erlarray". Este procedimiento que en el lenguaje de programación utilizado se llama

realmente función, devuelve la magnitud del valor más grande de los valores de ese arreglo. Esta función es de utilidad cuando tratamos el error relativo para sistemas de ecuaciones diferenciales (N > 1).

La función "funcionaelpaso" es un proceso que nos devuelve un valor booleano, el cual depende de si es aceptada o no la "H" propuesta. Esto se realiza de la siguiente manera; en el arreglo $\omega\tau$, calculamos "y" (como ya se mencionó) más que con precisión, con la intención de determinar una magnitud, ya que con esta y el error local, obtenemos el error relativo. En caso de que la magnitud de $\omega\tau$ sea cero tomaremos como error relativo al error local ya que siendo $\|\omega\tau\| = 0$ el error local nos dice mucho sobre la precisión obtenida.

Realmente es muy complicado que $\|\omega\tau\|$ sea cero, ya que para aproximarlo hemos utilizado un "promedio" en el que requerimos "y" y "$y$tilde" de paso completo y además de medio paso. Para obtener una magnitud de estos cuatro, se le dió mayor peso a los de paso completo, siendo este peso 3 veces mayor que el que se le dió a los de medio paso, quedando $\|\omega\tau\|$ de la siguiente forma:

$$\|\omega\tau\| = (1/8)[Abs(ym') + Abs(ym) + Abs(3y') + Abs(3y)]$$

Donde ym y ym' son y y y' pero con medio paso.

Este función termina comparando tolerancia con el error relativo más grande y revisa que esté en el rango para aceptar el paso.

El siguiente procedimiento, "GeneraH" es quien, a partir de si se aceptó o no el tamaño de "H" propuesto por última vez, alarga o encoge "H". Como ya habíamos visto, la fórmula de "H" implica una división, (Tol/Erl) donde Erl = Error relativo; si no se aceptó el paso anterior, implica:

Erl > Tol ⇒ (Tol/Erl) < 1 ∴ El paso siguiente crecerá.

Pero si Erl ≤ Tol ⇒ (Tol/Erl) ≤ 1 ∴ El paso se reduce.

Todo esto es debido a que Beta = Sqrt(Tol/Erl) y como H:=(9/10)*Beta entonces el alargamiento o encogimiento de "H" es automático dependiendo del valor de (Tol/Erl).

En caso de que el error relativo (Erl) sea cero, se da a Erl el valor mínimo manejable por la computadora y despues de la división (Tol/Erl) se multiplica por "10" para compensar lo que se dió a Erl.

Aunque ya se señaló que es conveniente tomar (9/10) del valor de "H" por razones de frontera, cabe tambien señalar que si tomamos el tamaño completo de "H", podría ser que aumentara el número de pasos rechazados, y aunque al final obtendríamos el mismo resultado de "$y(t)$", el costo sería más alto.

Este proceso realiza la generación de la nueva "H" dependiendo de el orden del error, el cual se dió como valor a la constante "Global=2"; Esto es debido a que la fórmula para desarrollar "H"

requiere de la raiz n-ésima. Es necesario que la constante "Global" tenga el valor del orden que se planea generar.

Al dar una "H" inicial, si esta no es de tamaño adecuado (es decir, se rechaza como tamaño de paso), entonces inmediatamente es ajustada al tamaño necesario; la diferencia entre dar una "H" inicial de tamaño adecuado y otro incorrecto, implica solo un incremento en el contador de pasos rechazados; el proceso que genera "H" no solo encoge "H", sino que lo hace a una proporción adecuada, de tal manera que es muy dificil que el proceso de rechazo del paso se repita 2 veces seguidas.

Un procedimiento se encargará de interpolar linealmente; esta es la función "Interpola", ya que el tamaño de "H" no esta predeterminado, es casi imposible terminar exactamente en el punto "t", por lo que esta función se encargará de determinar $y(t)$ mediante los valores de los 4 puntos, $t-\alpha$, $t+\kappa$, $y(t-\alpha)$, $y(t+\kappa)$, donde $t-\alpha \leq t \leq t+\kappa$. El resultado obtenido es el único valor que devuelve. Este procedimiento se encarga de interpolar para cada ecuación del sistema.

"Imprimemalla" se encarga de imprimir la "malla" de resultados del intervalo $(y(t_0), y(t))$. Otro procedimiento que imprime es "Imprimeresultados", que imprime además de los resultados obtenidos, una serie de datos (ya mencionados) que sirven para señalar el trabajo que se requirió durante el proceso. Una opción al inicio de este

proceso, pregunta por el destino solicitado de estos datos (pantalla o impresora).

Otros 2 procesos, "red" y "mandanodosiguientealared" son procesos que se encargan de obtener los datos que imprimirá "Imprimemalla". Estos son procesos que no interfieren con resultados relevantes; sin embargo, los elementos de la malla de datos nos muestran la forma de crecimiento de $y(t)$ calculado.

El procedimiento final se llama "Principal" y hace el papel de administrar los procedimientos anteriores; esto es, trabaja mientras no se llegue a "t", checando que la "H" requerida no sea cero, checa que funcione el paso, incrementa diversos contadores, vigila que "H" ha sido la más grande utilizada; arma los elementos de la malla (lo que equivale a generar $y(t)$ para varios puntos "t"), e interpola si no se cae en el punto exacto "t" del intervalo. Si hay algun problema o se termina de generar "$y(t)$" termina el proceso. Un proceso adaptado a este procedimiento es el de recorrimiento de la "t"; este proceso visualiza en la pantalla en que punto del eje "t" se encuentra el programa, para que de esta manera, el usuario se de cuenta de la velocidad a la que se esta construyendo el punto.

Esta ha sido una explicación sobre los procesos individuales del programa que sin duda puede mejorar modificando las funciones generadoras de "$y(t)$", al orden que se desee, para obtener tambien una mejor aproximación del error local.

Un punto importante del ajuste de paso, consiste en que una función puede ser suave en algunas partes, mientras que en otras puede volverse poco estable; con esto el ajuste automático de paso permite adaptar el tamaño de "H" a las necesidades de la función; con algunas funciones avanzaremos mucho más rapido y con otras el proceso será lento o muy lento (dependiendo además de las caracteristicas de la computadora en uso).

A continuación se da un breviario de las lineas que deberán ser cambiadas en caso de que se requiera resolver sistemas de ecuaciones distintos.

Dentro de las primeras lineas del listado tenemos las constantes "Orden", "N" y "Tol". "Tol" es la tolerancia que se va a requerir respecto del error relativo para condicionar la aceptación o rechazo del tamaño de paso "H". "Orden" es el orden del error que se va a estimar, es decir, es el máximo del orden de los 2 métodos de estimación en uso (q' y q). "N" es la dimensión del sistema de ecuaciones diferenciales que se está usando.

En el procedimiento FDY está la linea FY donde va el sistema requerido (ya se explicó como substituirla). Finalmente, en el procedimiento RK-12 están los 2 métodos "q" y "q'" con los que se van a calcular $y(t)$ y el error local. Estas ecuaciones se encuentran en las lineas que empiezan con:

```
Y1:=                    (y con paso completo)

YH2[I]:=                (y con medio paso)

YTILDEH2[I]:=           (y tilde con medio paso)

YTILDE:=                (y tilde con paso completo)
```

en un punto de una ecuación diferencial se debe a que no se tiene la solución analítica). Es por esto que los teoremas de existencia y unicidad y el intervalo de estabilidad nos serán útiles para saber que podemos esperar de nuestros resultados.

Con estas pruebas, además de los resultados obtenidos del calculo de las ecuaciones diferenciales, nos daremos cuenta de la confiabilidad del programa y de que tanto confiar en los resultados obtenidos numéricamente.

En este trabajo no se presentan resultados con sistemas de ecuaciones diferenciales, ya que implica el mismo trabajo que se realiza para las que estamos utilizando; sin embargo se realizaron pruebas con sistemas (cuyos resultados no presentaremos aquí por no tratarse de esto el presente trabajo) y se llegó a resultados similares. El proceso obviamente es más tardado, pero igual de preciso.

Para las 3 ecuaciones diferenciales de este trabajo, se calculó $y(1)$, pero modificando en las 2 primeras ecuaciones el parámetro variable "λ", el cual dependiendo de su signo implica estabilidad o inestabilidad en la funcion. Con esto veremos gráficamente, al terminar la siguiente prueba, que comportamiento siguen los errores local y global tanto de la ecuación estable como de la inestable, a fin de ver que esperamos al obtener estimaciones de $y(t)$ para "t" tal que $|t-t_0|$ es grande. A continuación se muestran los desarrollos que nos permitirán interpretar los resultados que nos dará el programa.

5.2 REGION DE ESTABILIDAD, EXISTENCIA Y UNICIDAD PARA LAS ECUACIONES DIFERENCIALES DE EJEMPLO.

Empezando con existencia y unicidad, tenemos que para la ecuación $y'=\lambda y$ y mediante Lipschitz:

$$|\lambda y - \lambda \bar{y}| \leq L|y - \bar{y}|$$

$$\lambda|y - \bar{y}| \leq L|y - \bar{y}| \quad \Rightarrow \quad \lambda \leq L$$

Lo anterior nos dice que dado el valor de λ, basta con que L sea cualquier real mayor o igual. Con esto demostramos que esta ecuación diferencial tiene solución para todo $\lambda \in \mathbb{R}$ (independientemente de que sea estable o no).

En el caso de la ecuación:

$y' = \cos(t) + \lambda(y - \text{sen}(t))$, esta es continua para todo $t \in \mathbb{R}$.

Ahora, tenemos utilizando el segundo teorema:

$$|f_y| = |\lambda| \leq M$$

Y aquí se cumple lo mismo que en el problema anterior, esto es, $\lambda \leq M$.

Entonces la ecuación diferencial tiene solución y esta es única.

El último caso es el de $y' = e^{-t^2}$, de donde tenemos, dado que esta función es continua respecto a "t" $\forall\ t \in \mathbb{R}$:

$|f_y| = 0 \leq M$ $\therefore$ M es cualquier real positivo, con lo que se cumple el segundo teorema significando esto que la solución existe y es única para todo $t \in \mathbb{R}$.

Ahora que sabemos que las 3 ecuaciones diferenciales anteriores tienen solución, se procederá a obtener su región de estabilidad absoluta.

En nuestros 2 casos particulares :

$y' = \lambda y = f(t,y)$ tenemos $df/dy = \lambda$

$y' = \cos(t) + \lambda(y - \text{sen}(t)) = f(t,y)$ y de aquí tenemos:

$df/dy = \lambda$ $\therefore$ para ambas se requiere $|1 + \lambda h| \leq 1$

Cuando utilizamos $\lambda = 1$ tenemos :

$|\ 1 + h\ | \leq 1$ es decir $-1 \leq 1 + h \leq 1$

$-1 \leq 1 + h \Rightarrow -2 \leq h$ y $h \leq 0$

$\therefore -2 \leq h \leq 0$

Pero si utilizamos $\lambda = -1$ tenemos:

$$|1 - h| \leq 1 \qquad \begin{array}{l|l} -1 \leq 1 - h \leq 1 & 1 - h \leq 1 \\ h - 1 \leq 1 & 1 \leq 1 + h \\ h \leq 2 & 0 \leq h \end{array}$$

$$0 \leq h \leq 2$$

Para el caso en que $\lambda = -1$ tenemos que el rango requerido $[0,2]$ es relativamente amplio, mientras que para $\lambda = -1$ nos encontramos ante una situación difícil, ya que, para mantener el método numérico en la región de estabilidad, "h" debe ser negativa o nula, lo cual es imposible. Esto implica que la ecuación $y'=y$ siempre se encontrará (sin importar el tamaño de "h") fuera de la región de estabilidad.

Cuando incrementamos el parámetro "λ" nuestro intervalo de estabilidad disminuye, es decir, para $y' = -10y$ tenemos:

$$\begin{array}{l|l} |1-10h| \leq 1 & -2 \leq -10h \leq 0 \\ -1 \leq 1-10h \leq 1 & 1/5 \geq h \geq 0 \end{array}$$

En este caso la región de estabilidad se reduce, pero aún es lo sufucientemente grande, ya que para $0 \leq h \leq 1/5$ es muy difícil salirnos del rango.

Este rango es el mismo para $y' = \cos(t) + \lambda(y - \text{sen}(t))$, ya que para:

$|1-hf_y(t,\beta)| \leq 1$ donde $f(t,\beta) = \lambda$

$\therefore |1 - hf_y(t,\beta)| = |1 - h\lambda| \leq 1$

En general:

Sea $y' = \lambda y$ $\quad |1 + \lambda h| \leq 1$ para $\lambda > 0$ tenemos que:

$1 + \lambda h \leq 1 \qquad \lambda h \leq 0 \qquad \Rightarrow \; ! \; h \leq 0 \; !$

Finalmente para $y' = e^{-t^2}$ obtendremos su región de estabilidad:

$|1 - hf_y(t,\beta)| \leq 1 \qquad f(t,y) = e^{-t^2}$

$\therefore \partial f/\partial y = 0 \Rightarrow |1 + 0h| \leq 1 \therefore$ es estable $\forall h \in \mathbb{R}$

5.3 RESULTADOS OBTENIDOS Y CONCLUSIONES.

Para la función $y' = \lambda y$ con $y(0)=0$ se calculó $y(1)$ y se utilizó $\lambda = 1, -1, 10, -10, 22$ y -22 y los resultados obtenidos fueron los siguientes:

FUNCION: Y[1]

T INICIAL : 0

CON VALORES DE Y :

Y1(0) : 1

EN EL PUNTO T= 1

Y1(0.1): 1.10510093

Y1(0.2): 1.22124771

Y1(0.3): 1.34960156

Y1(0.4): 1.49144556

Y1(0.5): 1.64819730

Y1(0.6): 1.82142397

Y1(0.7): 2.01285653

Y1(0.8): 2.22440922

Y1(0.9): 2.45819564

Y1(1.0): 2.71655389

H INICIAL : 0.001

H MAYOR UTILIZADA : 0.0012735

PROMEDIO DE PASO H : 0.00127226

TOLERANCIA : 0.000001

DE PASOS ACEPTADOS : 786

DE PASOS RECHAZADOS : 0

TOTAL DE PASOS : 786

==========

Valor Real = 2.7182818

Error = 0.001728

Er1 = 0.0006357

FUNCION: -Y[1]

T INICIAL : 0

CON VALORES DE Y :

Y1(0) : 1

EN EL PUNTO T= 1

Y1(0.1): 0.90478005

Y1(0.2): 0.81862680

Y1(0.3): 0.74067688

Y1(0.4): 0.67014960

Y1(0.5): 0.60633782

Y1(0.6): 0.54860218

Y1(0.7): 0.49636424

Y1(0.8): 0.44910029

Y1(0.9): 0.40633692

Y1(1.0): 0.36764545

H INICIAL : 0.001

H MAYOR UTILIZADA : 0.00127224

PROMEDIO DE PASO H : 0.00127065

TOLERANCIA : 0.000001

DE PASOS ACEPTADOS : 787

DE PASOS RECHAZADOS : 0

TOTAL DE PASOS : 787

==========

Valor Real = 0.3678794

Error = 0.000234

Er1 = 0.0006361

```
FUNCION: 10Y(1)

T INICIAL : 0

CON VALORES DE Y :

Y1(0) : 1

EN EL PUNTO T=  1

Y1( 0.1):  2.71655336

Y1( 0.2):  7.37966155

Y1( 0.3): 20.04724089

Y1( 0.4): 54.45938519

Y1( 0.5): 147.94181626

Y1( 0.6): 401.89181270

Y1( 0.7): 1091.76038620

Y1( 0.8): 2965.82462050

Y1( 0.9): 8056.81978660

Y1( 1.0): 21886.77993300

H INICIAL :  0.001

H MAYOR UTILIZADA :  0.00012784

PROMEDIO DE PASO H :  0.00012734

TOLERANCIA :  0.000001

# DE PASOS ACEPTADOS : 7853

# DE PASOS RECHAZADOS : 1

# TOTAL DE PASOS : 7854

================================================

Valor Real = 22026.466

Error = 139.687

Erl = 0.0063418
```

```
FUNCION: 22Y[1]
T INICIAL : 0
CON VALORES DE Y :
Y1(0) : 1
EN EL PUNTO T=  1
Y1( 0.1):  9.01239211
Y1( 0.2): 81.22318137
Y1( 0.3): 732.01514452
Y1( 0.4): 6597.2052724
Y1( 0.5): 54456.59747
Y1( 0.6): 535846.00557
Y1( 0.7): 4829253.8791
Y1( 0.8): 43523117.572
Y1( 0.9): 392247352.46
Y1( 1.0): 3535086086.3
H INICIAL :  0.001
H MAYOR UTILIZADA :  0.00005841
PROMEDIO DE PASO H :  0.00005788
TOLERANCIA :  0.000001
# DE PASOS ACEPTADOS : 17276
# DE PASOS RECHAZADOS : 1
# TOTAL DE PASOS : 17277
================================================
Valor Real = 3,584,912,846
Error = 49,826,760
Erl = 0.013899
```

De acuerdo a los teoremas de existencia y unicidad utilizados, esta ecuación diferencial tiene solución; situación que conociamos de antemano por conocer la solución analitica. La única razón por la que se utilizó esta prueba fué la de ejemplificar.

Mediante el intervalo de estabilidad nos damos cuenta que para $\lambda > 0$ encontraremos desestabilidad, principalmente para valores grandes de λ; mientras que para $\lambda < 0$ tendremos estabilidad si no damos valores $|\lambda|$ muy altos.

Como es requerido $0 \leq h \leq 2$ para $\lambda = -1$, tenemos una amplia libertad para mover "h", esto es, no requerimos poner un tope porque la "h" mayor utilizada nunca fué ni de cerca parecida al limite aceptado (de hecho nunca se utilizó en el programa la opción que evita que se rebase este limite ya que no fué necesario).

En los casos para $\lambda = -1$, -10 y -22 se esperan buenos resultados. Mientras λ es mayor, el rango de "h" es menor; sin embargo, para estos tres valores de "λ" no hay problema por no reducirse este rango rapidamente.

De los resultados obtenidos podemos observar que los errores relativos son muy chicos, sin embargo el error resultado de la diferencia entre el valor real y el estimado nos da una diferencia muy marcada entre los "λ" positivos y los negativos, aunque se requirió aproximadamente igual número de pasos para las funciones $y' = \lambda_1 y$ y $y' = \lambda_2 y$ cuando $|\lambda_1| = |\lambda_2|$.

Una cantidad muy grande de pasos requeridos implica mayor dificultad para obtener el mínimo error local requerido, lo que sería observado si modificaramos la tolerancia, ya que, a mayor exigencia en la precisión el tamaño de paso promedio sería menor (independientemente de si este fuera rechazado o no).

Hagamos ahora una suposición con respecto al número de pasos requeridos y la tolerancia; tomando el supuesto de que el error en cada uno de los "n" pasos fue de la magnitud de la tolerancia, tenemos así que el error local acumulado debido al error máximo en la totalidad de pasos efectuados es = (# de pasos aceptados multiplicado por la tolerancia). En el caso de esta primera función $y' = \lambda y$, se observa que el error local obtenido, para toda "λ" utilizada, no es mayor a esa cantidad, aunque no sea estable. Esto nos habla de un control en el error local por medio de la tolerancia. Podemos mejorar, entonces, el resultado, pero solo hasta un cierto grado, ya que al ser más estrictos en la tolerancia estamos propiciando que el tamaño de paso promedio sea menor y por lo tanto habrá mayor acumulación de otro tipo de errores como el de redondeo, ya que con el tamaño de paso menor, estamos realizando más estimaciones y con cada una de estas, efectuamos muchas operaciones, y con estos errores se acumulan otros posibles paso a paso. Durante algunas pruebas se notó que al dar una tolerancia muy pequeña, se propició un aumento en el error, en comparación con tolerancias menos estrictas.

Debido a los resultados obtenidos por el intervalo de estabilidad, tenemos que para $\lambda < 0$ los resultados obtenidos son confiables, pero lo son más mientras la magnitud de λ no sea muy grande. Aquí se obtuvo aún para $\lambda = -22$ un buen resultado, y con un numero de pasos necesarios relativamente corto.

Es importante darnos cuenta de varios resultados en conjunto, nunca individualmente, para considerar si nuestro resultado es confiable. Si consideramos solamente, por ejemplo, el número de pasos aceptados, la tolerancia o el promedio de paso, podemos estar en un error, ya que como puede verse para $y' = \lambda_1 y$ y $y = \lambda_2 y$ donde $\lambda_1 \neq \lambda_2$ y $|\lambda_1| = |\lambda_2|$ tenemos que el número de pasos, el error local, la tolerancia y la "h" máxima utilizada son practicamente los mismos, mientras que nuestros resultados son por una parte confiables y por otra no confiables. Un punto que hemos de considerar, y que es más importante, consiste en el resultado obtenido de el intervalo de estabilidad, lo cual nos dice que para $\lambda > 0$ no hay estabilidad, mientras que para $\lambda < 0$ si la hay. El hecho de que para $\lambda < 0$, $|\lambda|$ sea muy grande, va a ir deteriorando la precisión. Es justamente este último resultado el que nos aclara en definitiva si nuestro resultado es o no confiable.

Entonces, si la "h" máxima utilizada nunca fué mayor que la "h" mayor autorizada por el intervalo de estabilidad, entonces podemos considerar que nuestro resultado es confiable.

Lo siguiente es un análisis comparativo del comportamiento de los errores de la ecuación de prueba $y'=\lambda y$ para los valores $\lambda = 1, -1$. El tener los resultados de las estimaciones de $y(1)$ y $y(-1)$ con errores locales chicos no nos dice mucho y en cambio podría confundirnos al hacernos pensar que para valores pequeños de λ si $\lambda>0$ o $\lambda<0$ se tiene estabilidad para ambos solo porque el error local para ambos es muy pequeño o por que el resultado es preciso; esto no es así, ya que el error local de la ecuación de prueba para una ecuación estable es decreciente (figura 5.1), mientras que para una inestable es creciente (figura 5.2); esto es, a cada paso en una ecuación diferencial estable el error local es menor por lo que, teniendo un valor inicial preciso y_0 y dando un punto inicial de "h" ideal desde el principio, podremos obtener mejores resultados.

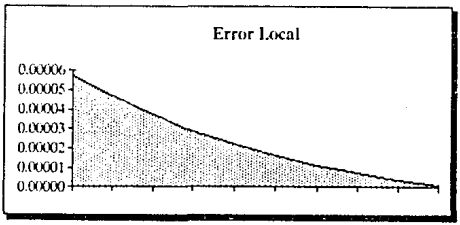

Figura 5.1.- Error para $y'=-y$

Por punto ideal de "h" se hace referencia a uno que no se aleje de la "h" promedio utilizada, ya que (y esperando que $y(t)$ sea igualmente suave en todo el rango) tenemos que el tamaño promedio es un tamaño estandar de acuerdo al comportamiento de $y'(t)$. Esto permitirá evitar errores locales grandes desde el principio, que tengan peso en el error global.

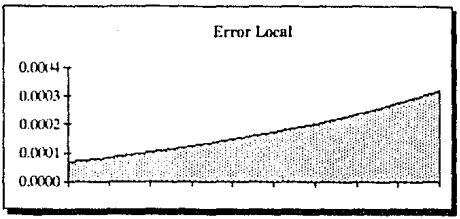

Figura 5.2.- Error para $y'=y$

En base al comportamiento de los anteriores errores, los errores globales tiene comportamientos muy distintos (figuras 5.3 y 5.4). El primero da la apariencia de un incremento que tiende a desaparecer, con una pendiente que se acerca cada vez más a cero. Caso contrario de $y'=\lambda y$ con $\lambda > 0$ que muestra un incremento acelerado y creciente. Entonces, aunque los resultados real y estimado de $y'=\lambda y$ sean similares tanto para un ejemplo estable como para uno inestable, es a futuro, es decir, para valores "t" tales que $|t-t_0|$ es grande, cuando $y(t)$ estimada tenderá a fallar más visiblemente para este caso.

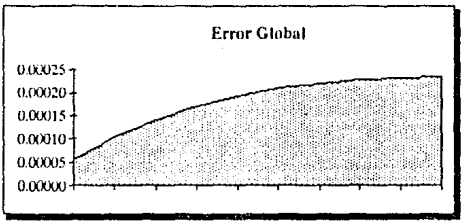

Figura 5.3.- Error para $y'=-y$

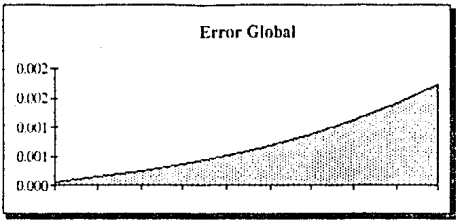

Figura 5.4.- Error para $y'=y$

Para la función $y' = \cos(t) + \lambda(y(t)-\text{sen}(t))$ con $y(0) = 1$ se calculó $y(1)$ y se utilizó $\lambda = 1, -1, -10, 10, 22$ y -22. Los resultados se presentan a continuación:

```
FUNCION: COS(Y[0])+(Y[1]-SIN(Y[0]));
T INICIAL : 0
CON VALORES DE Y :
Y1(0) :  1
EN EL PUNTO T=  1
Y1( 0.1):  1.20493451
Y1( 0.2):  1.41991800
Y1( 0.3):  1.64512429
Y1( 0.4):  1.88086981
Y1( 0.5):  2.12763361
Y1( 0.6):  2.38608340
Y1( 0.7):  2.65709988
Y1( 0.8):  2.94180056
Y1( 0.9):  3.24157065
Y1( 1.0):  3.55808595
H INICIAL :  0.001
H MAYOR UTILIZADA :  0.00177977
PROMEDIO DE PASO H :  0.00163132
TOLERANCIA :  0.000001
# DE PASOS ACEPTADOS : 613
# DE PASOS RECHAZADOS : 0
# TOTAL DE PASOS : 613
```

Valor Real = 3.5597

Error = 0.001615

Er1 = 0.0004537

FUNCION: COS(Y[0])-1(Y[1]-SIN(Y[0]));

T INICIAL : 0

CON VALORES DE Y :

Y1(0) : 1

EN EL PUNTO T= 1

Y1(0.1): 1.00461362

Y1(0.2): 1.01729709

Y1(0.3): 1.03620082

Y1(0.4): 1.05957669

Y1(0.5): 1.08578227

Y1(0.6): 1.11328423

Y1(0.7): 1.14066737

Y1(0.8): 1.16659224

Y1(0.9): 1.18985008

Y1(1.0): 1.20935174

H INICIAL : 0.001

H MAYOR UTILIZADA : 0.01221234

PROMEDIO DE PASO H : 0.00228311

TOLERANCIA : 0.000001

DE PASOS ACEPTADOS : 438

DE PASOS RECHAZADOS : 4

TOTAL DE PASOS : 442

==========

Valor Real = 1.2093504

Error = 0.0000013

Er1 = 0.00000107

FUNCION: COS(Y[0])-10(Y[1]-SIN(Y[0]));

T INICIAL : 0

CON VALORES DE Y :

Y1(0) : 1

EN EL PUNTO T= 1

Y1(0.1): 0.46746772

Y1(0.2): 0.33380204

Y1(0.3): 0.34516949

Y1(0.4): 0.40764466

Y1(0.5): 0.48611047

Y1(0.6): 0.56711046

Y1(0.7): 0.64514711

Y1(0.8): 0.71772453

Y1(0.9): 0.78349252

Y1(1.0): 0.84156457

H INICIAL : 0.001

H MAYOR UTILIZADA : 0.00486849

PROMEDIO DE PASO H : 0.00041771

TOLERANCIA : 0.000001

DE PASOS ACEPTADOS : 2394

DE PASOS RECHAZADOS : 5

TOTAL DE PASOS : 2399

==========

Valor Real = 0.8415163

Error = 0.0000482

Er1 = 0.0000573

FUNCION: COS(Y[0])+10(Y[1]-SIN(Y[0]));

T INICIAL : 0

CON VALORES DE Y :

Y1(0) : 1

EN EL PUNTO T= 1

Y1(0.1): 2.81636400

Y1(0.2): 7.57819433

Y1(0.3): 20.34226117

Y1(0.4): 54.84726963

Y1(0.5): 148.41682085

Y1(0.6): 402.44419452

Y1(0.7): 1092.37095520

Y1(0.8): 2966.45057520

Y1(0.9): 8057.35441390

Y1(1.0): 21886.94204000

H INICIAL : 0.001

H MAYOR UTILIZADA : 0.00012969

PROMEDIO DE PASO H : 0.00012798

TOLERANCIA : 0.000001

DE PASOS ACEPTADOS : 7814

DE PASOS RECHAZADOS : 1

TOTAL DE PASOS : 7815

==========

Valor Real = 22,027.307

Error = 140.365

Er1 = 0.006373

FUNCION: COS(Y[0])+22(Y[1]-SIN(Y[0]));

T INICIAL : 0

CON VALORES DE Y :

Y1(0) : 1

EN EL PUNTO T= 1

Y1(0.1): 9.11214069

Y1(0.2): 81.42077478

Y1(0.3): 732.30010796

Y1(0.4): 6597.5008868

Y1(0.5): 59456.216371

Y1(0.6): 535838.92743

Y1(0.7): 4829184.5359

Y1(0.8): 43522496.694

Y1(0.9): 392241661.52

Y1(1.0): 3535035534.3

H INICIAL : 0.001

H MAYOR UTILIZADA : 0.00005844

PROMEDIO DE PASO H : 0.00005794

TOLERANCIA : 0.000001

DE PASOS ACEPTADOS : 17258

DE PASOS RECHAZADOS : 1

TOTAL DE PASOS : 17259

==========

Valor Real = 3,584,912,846

Error = 49,877,380

Er1 = 0.0139131

Es probable que tomando una "h" máxima como tope, se podría mejorar la precisión, ya que como se ha visto, para $\lambda < 0$ el tamaño de paso "h" crece rapidamente y propicia que se pierda precisión, aunque debido a la forma en que se genera "h" es precisamente por el tamaño tan pequeño de error local que permite el crecimiento tan acelerado de "h". Talvez dando un tope a "h" daría una mejoría muy pequeña, ya que la ecuación diferencial es estable y los errores ya son muy pequeños.

Los resultados para $\lambda<0$ como es de esperarse son confiables y con escaso error, además de que el error local no es ni cercano al máximo error posible por acumulación de errores locales, mientras que para $\lambda>0$ los resultados no son malos. Con esto nos podemos dar cuenta que, el hecho de que si se requieren más pasos para suavizar la función, no implica que el resultado vaya a ser satisfactorio; en cambio, esto podría implicar un esfuerzo mayor pero inutil por parte del programa para explicar una función inestable.

Finalmente, para la función $y' = e^{-t^2}$ con $y(0) = 0$ se calcularon $y(1)$, $y(5)$, $y(-.5)$ y $y(-1)$; no se utilizó parámetro variable. Lo que se obtuvo fué:

FUNCION: EXP(-SQR(Y[0]));

T INICIAL : 0

CON VALORES DE Y :

Y1(0) : 0

EN EL PUNTO T= 1

Y1(0.1): 0.09967214

Y1(0.2): 0.19738279

Y1(0.3): 0.29127715

Y1(0.4): 0.37972108

Y1(0.5): 0.46138466

Y1(0.6): 0.53529759

Y1(0.7): 0.60087395

Y1(0.8): 0.65790485

Y1(0.9): 0.70652416

Y1(1.0): 0.74715439

H INICIAL : 0.001

H MAYOR UTILIZADA : 0.00128194

PROMEDIO DE PASO H : 0.00100503

TOLERANCIA : 0.000001

DE PASOS ACEPTADOS : 995

DE PASOS RECHAZADOS : 1

TOTAL DE PASOS : 996

==========

Valor Real = 0.7457245

Error = 0.0014298

Er1 = 0.001917

```
FUNCION: EXP(-SQR(Y[0]));

T INICIAL : 0

CON VALORES DE Y :

Y1(0) :  0

EN EL PUNTO T=  5

Y1( 0.5):  0.46138466

Y1( 1.0):  0.74715439

Y1( 1.5):  0.85672435

Y1( 2.0):  0.88273826

Y1( 2.5):  0.88657479

Y1( 3.0):  0.88693236

Y1( 3.5):  0.88695502

Y1( 4.0):  0.88695627

Y1( 4.5):  0.88695638

Y1( 5.0):  0.88695638

H INICIAL :  0.001

H MAYOR UTILIZADA :  1.37083338

PROMEDIO DE PASO H : 0.00314268

TOLERANCIA :  0.000001

# DE PASOS ACEPTADOS : 1591

# DE PASOS RECHAZADOS : 1

# TOTAL DE PASOS : 1592

================================================

Valor Real = 0.8862092

Error = .0007471

Er1 = 0.000843
```

FUNCION: EXP(-SQR(Y[0]));

T INICIAL : 0

CON VALORES DE Y :

Y 1(0) : 0

EN EL PUNTO T= -0.5

Y 1(-0.05000000): -0.04995948

Y 1(-0.10000000): -0.09967214

Y 1(-0.15000000): -0.14889259

Y 1(-0.20000000): -0.19738279

Y 1(-0.25000000): -0.24491539

Y 1(-0.30000000): -0.29127715

Y 1(-0.35000000): -0.33627147

Y 1(-0.40000000): -0.37972108

Y 1(-0.45000000): -0.42146989

Y 1(-0.50000000): -0.46138466

H INICIAL : 0.001

H MAYOR UTILIZADA : 0.00101818

PROMEDIO DE PASO H : 0.00092593

TOLERANCIA : 0.000001

DE PASOS ACEPTADOS : 540

DE PASOS RECHAZADOS : 1

TOTAL DE PASOS : 541

==========

Valor Real = -0.4628763

Error = 0.0014917

Er1 = 0.0003222

```
FUNCION: EXP(-SQR(Y[0]));
T INICIAL :  0
CON VALORES DE Y :
Y 1(0) :  0
EN EL PUNTO T= -1
Y 1(-0.10000000): -0.09967214
Y 1(-0.20000000): -0.19738279
Y 1(-0.30000000): -0.29127715
Y 1(-0.40000000): -0.37972108
Y 1(-0.50000000): -0.46138466
Y 1(-0.60000000): -0.53529759
Y 1(-0.70000000): -0.60087395
Y 1(-0.80000000): -0.65790485
Y 1(-0.90000000): -0.70652416
Y 1(-1.00000000): -0.74715439
H INICIAL :  0.001
H MAYOR UTILIZADA :  0.00128194
PROMEDIO DE PASO H : 0.00100503
TOLERANCIA :  0.000001
# DE PASOS ACEPTADOS : 995
# DE PASOS RECHAZADOS : 1
# TOTAL DE PASOS : 996
==============================================
Valor Real = -0.7457245
Error = .00142
Er1 = 0.001917
```

Para esta última ecuación diferencial tenemos que los 4 puntos estimados son confiables, ya que el intervalo de estabilidad abarca todo valor de "h"; pero es claro que al estimar un punto y_1 tal que $y_1 > y_r$ será de esperarse que será menos preciso que y_r, ya que para calcular el primero es necesario pasar antes por este último y es lógico pensar que habrá acumulación en el error global. Sin embargo en los resultados obtenidos podemos ver que hay una buena precisión para todos ellos, además de que el número de pasos fué corto, en todos se rechazó un paso, lo que puede deberse a que el tamaño de paso crece rapidamente y de hecho es grande y tiende a crecer, incluso nos damos cuenta que para la estimación de $y(5)$ se requirió de un paso muy grande, de magnutud 1.37 lo cual aunado al hecho de que para el punto más cercano a cero (-0.5) se utilizó el promedio de paso más chico mientras que para el punto $y(5)$ se requirió el más grande. Esto nos indica que nuestra función a partir del origen es más suave y requiere de pasos cada vez vez más grandes, por lo que llega a darlos tan grandes como el citado.

Sin ver los resultados reales nos podemos dar cuenta de la precisión confiable de los resultados debido a que según nuestro intervalo de estabilidad la misma es $\forall$ t, y porque según el teorema de existencia y unicidad nuestra ecuación existe tambien para todo "t". Además podemos darnos cuenta que al incrementarse rápidamente el tamaño de paso estamos reduciendo el error local considerablemente. Este último puede quedar más patente de la siguiente forma, considerando que para calcularse $y(-0.5)$ se requirieron 540 pasos, es de esperarse que para $y(1)$ se utilicen aproximadamente 1080, sin

embargo se requirieron solo 995; ahora, tomando este último resultado, para calcular $y(5)$ se requerirían 4975 pasos, pero el programa requirió de tan solo 1591. Entonces el mismo prgrama nos da un parámetro para indicarnos que tan rápido reduce el error, ya que, de acuerdo a que tan abajo se está de la tolerancia, es como se ajusta el siguiente tamaño de paso.

Considerando la opción de tamaño maximo para el paso, esta nos puede ser util cuando el tamaño de paso crece demasiado rápido lo que provoca que cada punto de la malla tenga que ser calculado por interpolación con puntos muy lejanos del requerido, lo que provoca pérdida de precisión. Esto pudo haber sucedido en nuestro último ejemplo, ya que el crecimiento del paso fué rápido.

La forma como se pudieron conocer los valores reales de $y(t)$ donde $y'(t)=e^{-t^2}$ fué a partir de la función normal y una tabla de la misma; esto es de la siguiente forma:

Para obtener $y(p)$ donde $y(t)=e^{-t^2}$ nos basaremos en lo siguiente:

Función estadistica de la normal $= \int_{-\infty}^{z} \sqrt{(2\pi)}\ e^{(-z^2/2)}\ dt$

requerimos la función $\int_{-\infty}^{p} e^{-t^2} dt$

Transformando la segunda llegaremos a una proporción de la primera, esto es

Sea $z=\sqrt{2}\ t \Rightarrow dz/dt = \sqrt{2} \therefore dz = \sqrt{2}\ dt$

$$\Rightarrow \int_0^p e^{-t^2}\,dt = 1/\sqrt{2} \int_0^{\sqrt{2}\,p} e^{(-z^2/2)}\,dt =$$

$$\sqrt{2\pi}/\sqrt{2} \left[\int_0^{\sqrt{2}\,p} 1/\sqrt{2\pi}\ e^{(-z^2/2)}\,dt \right] =$$

$$\sqrt{\pi} \left[\int_{-\infty}^{\sqrt{2}\,p} 1/\sqrt{2\pi}\ e^{(-z^2/2)}\,dt - \int_{-\infty}^{0} 1/\sqrt{2\pi}\ e^{(-z^2/2)}\,dt \right]$$

Como $= e^{-t^2}$ tiene $t = 0$ y $\psi(t) = 0 \Rightarrow$

$$\int_0^p \psi'\,dt = \int_0^p e^{-t^2}\,dt \quad \Rightarrow \quad \psi(p) - \psi(0) = \int_0^p e^{-t^2}\,dt$$

$$\psi(p) = \psi(0) + \int_0^p e^{-t^2}\,dt = \int_0^p e^{-t^2}\,dt$$

⇒ Para obtener $\psi(p)$, requerimos solo de una tabla de distribución "normal" (de cualquier libro de estadistica y obtenemos la normal en "p", le restamos 0.5 (la normal en cero) y al resultado de esta operación lo multiplicamos por $\sqrt{\pi}$.

5.4 CONCLUSIONES GENERALES.

Los resultados obtenidos con los métodos de estos órdenes son satifactorios, es decir, los parámetros que utilizamos nos sirvieron para saber que esperar de los resultados numéricos. El cambiar de orden a los métodos de estimación y error podrá aumentar la precisión, pero en base a lo ya visto sabemos de que funciones e intervalos podremos esperar mayor precisión y cuando será inutil obtener mejoria y confiabilidad.

Ahora estamos en una situación diferente a la de realizar programas de estimación de puntos, ya que ahora y antes de correr el programa podremos saber si nuestro resultado será confiable, si existe la solución, si es única, y en que regiones no habrá resultados satisfactorios.

Si la computadora es rápida (16 Mhz. o más) podremos decrementar el tamaño de la tolerancia, es decir, ser tan exigentes en la precisión como nos lo permita el costo de tiempo en el cálculo de la estimación. Además, si contamos con co-procesador matemático podremos usar 20 digitos en lugar de 11. Entonces a partir de este programa estaremos en condiciones de resolver ecuaciones diferenciales con alta precisión.

A partir de este trabajo se puede continuar con el estudio de métodos de mayor orden e incluso con Euler implicito. Existe una metodologia que nos da, en base a árboles, un sistema de ecuaciones

BIBLIOGRAFIA

1) Modern numerical methods for ordinary differential ecuations.
G. Hall & J. M. Watt

2) Fundamentals of numerical computing.
Richard Allen, Steven Pruess, Lawrence Shampine

3) Elementary numerical analysis, an algorithmic approach.
Samuel D. Conte, Carl De Boor

4) Numerical initial value problems in ordinary differential equations.
C. William Gear

5) Computational methods in ordinary differential equations.
J. D. Lambert

6) Differential equations.
Martin Braum

7) Error estimates for Runge-Kutta type solutions to systems of ordinary differential equations.
Computer Journal, 12 (1969), pp 166-170

8) The numerical analysis of ordinary differential equations.
Butcher J.C.

9) Introducción a las ecuaciones diferenciales ordinarias.
Earl A. Coddington