Métodos Numéricos (2015-2016)
Grado en Ingeniería Informática y Matemáticas
Universidad de Granada

Métodos de Runge-Kutta

Pilar Espinosa de los Monteros Manero
Marina Estévez Almenzar
Elena Toro Pérez

April 2016

Índice

1. Introducción

Una de las herramientas matemáticas más efectivas de las que disponemos actualmente para modelizar y explicar la naturaleza son las **ecuaciones diferenciales**. Muchos problemas de las Ciencias y las Ingenierías se plantean mediante la resolución de un textbfproblema de valores iniciales, en el que las ecuaciones diferenciales juegan un papel importante. Sin embargo, una resolución exacta de dicho problema no siempre es posible, principalmente por la imposibilidad o complejidad de resolver de forma explícita las ecuaciones diferenciales involucradas en el problema.
Se quiere hallar la solución de $y(x)$, que suponemos que existe y que es única en el intervalo $[a, b]$, de un problema de valores iniciales

$$(*) \begin{cases} y'(x) = f(x, y) \\ y(x_0) = c \end{cases}$$

donde $y'(x) = f(x, y)$ puede ser una ecuación diferencial de orden 1, o un sistema de ecuaciones diferenciales de orden 1. Nos centraremos en la resolución del problema supuesto el primer caso (una ecuación diferencial de orden 1). El problema, por tanto, consiste en encontrar la función $y(x)$ que es solución única del problema.

¿Cómo?
Se usan métodos que tratan de aproximar la única solución $y(x)$ de dicho problema, entre ellos, los más usados y los que abarcan los métodos de *Runge-Kutta* son los **métodos de discretización**.

Métodos de discretización
Sea $(*)$ nuestro problema de valores iniciales. Supongamos que la función f de nuestro problema está definida en $[a, b] \times \mathbb{R}$. Un método de discretización para resolverlo es cualquier método numérico que trata de obtener valores aproximados y_n de la solución de $y(x)$ en los distintos nodos x_n con $n = 0, 1, ..., N$ obtenidos mediante la partición del intervalo $[a, b]$.

Estos métodos de discretización, además de abarcar los métodos de *Runge-Kutta* que se exponen a continuación, también abarcan otros métodos como el método de *Euler* o los métodos de *Taylor*.
Veamos una definición esquematizada del método de *Runge-Kutta* de M evaluaciones (que posteriormente desarrollaremos para distintos valores de M):
Dado el problema de valores iniciales $(*)$ se define

$$(1) \begin{cases} y_0 = y(x_0) \\ y_{k+1} = y_k + h\phi(x_k, y_k, h), k = 0, 1, ..., N - 1 \end{cases}$$

donde

- $h = \frac{a}{N}, N \in \mathbb{N}$

- $x_k = x_0 + kh, k = 0, ..., N$
- $\phi(x, y, h) = \sum_{j=1}^{M} c_j K_j(x, y, h)$ con $K_i(x, y, h) = f(x+ha_i, y+h\sum_{j=1}^{M} b_{ij} K_j(x, y, h))$ para $i = 1, ..., M$ y c_i, a_i, b_{ij} constantes

Los métodos de *Runge-Kutta* de M evaluaciones que veremos se suelen representar mediante una tabla que contiene los coeficientes que determinan el método:
la tabla de Butcher

$$\begin{array}{c|cccc} a_1 & b_{11} & \cdots & & b_{1M} \\ a_2 & b_{21} & & & b_{2M} \\ \vdots & \vdots & \vdots & \ddots & \\ a_M & b_{M1} & \cdots & \cdots & b_{MM} \\ \hline & c_1 & c_2 & \cdots & c_M \end{array}$$

Algunas definiciones previas necesarias para el estudio de los métodos

Definiciones: error de truncamiento local y global
Se define
$u_{k+1} = y(x_{k+1}) - y(x_k) - h\phi(x_k, y(x_k), h)$ con $k = 0, 1, ..., N-1$ como los *errores de truncamiento/discretización locales* del método.
A $e_k = y(x_k) - y_k$ se le llama *error de truncamiento/discretización global*, siendo x_k los nodos de la partición del intervalo, y_k los valores aproximados de la solución $y(x)$ en el nodo x_k, con $k = 0, ..., N$.
Para estudiar la convergencia de estos métodos es necesario introducir los conceptos de **estabilidad** y **consistencia**:

Definición
Un método de la forma (1) se dice **estable** si existen M_1 y M_2 tales que para cualquier par de conjuntos y_k y z_k definidos por

$$\begin{cases} y_{k+1} = y_k + h\phi(x_k, y_k, h) \text{ con } y_0 \text{ dado} \\ z_{k+1} = z_k + h\phi(x_k, z_k, h) + \varepsilon_k \text{ con } z_0 \text{ dado} \end{cases}$$

se tiene que
$\max |y_k - z_k| \leq M_1 |y_0 - z_0| + M_2 \max |\varepsilon_k|$

Es importante hacer algunas observaciones de interés:

- Las constantes M_1 y M_2 no dependen de h y por tanto la estabilidad no depende del p.v.i. planteado.
- La característica de estabilidad en un método indica que pequeñas perturbaciones en los datos de entrada originan pequeñas perturbaciones en los valores que se van obteniendo en el método.

Definición
Un método de la forma (1) se dice **consistente** con el p.v.i. $(*)$ cuando se verifica que $\lim_{h\to 0}(\text{máx}\,\frac{|u_{k+1}|}{h}) = 0$
siendo $u_{k+1} = y(x_{k+1} - y(x_k) - h\phi(x_k, y(x_k), h)$ el error de truncamiento local, con $k = 0, 1, ..., N-1$.

Un resultado importante que relaciona **estabilidad** y **consistencia** es el siguiente:

Si un método de la forma (1) es estable y consistente con un p.v.i. de la forma $(*)$ entonces dicho método converge a la solución de $(*)$

Algunas observaciones sobre los métodos de Runge-Kutta que vamos a estudiar

- Un método de Runge-Kutta se dice que es *explícito* cuando $b_{ij} = 0$ para $j \geq i$
- Si f y ϕ son continuas, entonces
  el método de Runge-Kutta es *consistente* $\iff \sum_{j=1}^{M} c_j = 1$
- Para simplificar los métodos de Runge-Kutta se exige que $a_i = \sum_{j=1}^{M} b_{ij}$

2. Métodos de Runge-Kutta

2.1. Método del punto medio

Sea f una función definida en $[x_0, x_0 + a]x\mathbb{R}$. Sea la ecuación diferencial de orden 1

$$\begin{cases} y' = f(x, y) \\ y(x_0) = \eta \end{cases}$$

un p.v.i de ecuación $y' = f(x, y)$ y condición inicial $y(x_0) = \eta$ con solución única $y(x)$ en $[x_0, x_0 + a]$.

El método del **punto medio o Euler Modificado** es el método de Runge-Kutta en el que se utiliza el valor de la función f(x,y) evaluado en el punto medio entre x_k y $x_\text{k} + h$, tomando en ese punto $x_\text{k} + \frac{h}{2}$, el valor obtenido mediante el método de Euler.

2.1.1. Descripción

El método del punto medio es el método de Runge-Kutta de 2 evaluaciones cuyo tablero de Butcher es

$$\begin{array}{c|cc} a_1 = 0 & & \\ a_2 = 1/2 & b_{21} = 1/2 & \\ \hline & c_1 = 0 & c_2 = 1 \end{array}$$

es decir:

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

tal que:

$$\Phi(x, y, h) = K_2(x, y, h)$$

siendo:

$$K_1(x, y, h) = f(x, y)$$
$$K_2(x, y, h) = f(x + \tfrac{h}{2}, y + \tfrac{h}{2} K_1(x, y, h))$$

Más concretamente, el método del punto medio para resolver dicha ecuación diferencial de orden 1, consiste en hallar $y_{\mathrm{k}} \in \mathbb{R}$ para $k = 0, 1, ..., N$, siendo $N \in \mathbb{N} - \{0\}$, mediante el siguiente proceso:

$$\mathbf{1^o)}\ y_0 = y(x_0)$$

$$\mathbf{2^o)}\ y_{\mathrm{k}+1} = y_{\mathrm{k}} + hf(x_{\mathrm{k}} + \tfrac{h}{2}, y_{\mathrm{k}} + \tfrac{h}{2} f(x_{\mathrm{k}}, y_{\mathrm{k}})), \text{ con } k = 0, ..., N - 1$$

donde $h = \frac{a}{N}$ y $x_{\mathrm{k}} = x_0 + kh$, con $k = 0, ..., N$.

A la segunda igualdad anterior se le llama **ecuación en diferencias del método del punto medio**.

2.1.2. Observación: Deducción del método del Punto Medio a partir de integración numérica

La ecuación en diferencias del método del punto medio para resolver la ecuación diferencial de orden 1 de la que estamos hablando, puede obtenerse igualando, para cada $k = 0, ..., N - 1$, las expresiones que resultan de sustituir $y(x_{\mathrm{k}+1})$ e $y(x_{\mathrm{k}})$ por $y_{\mathrm{k}+1}$ e y_{k}, respectivamente, en los dos miembros de la aproximación

$$y(x_{\mathrm{k}+1}) \simeq y(x_{\mathrm{k}}) + hf(x_{\mathrm{k}} + \tfrac{h}{2}, y_{\mathrm{k}} + \tfrac{h}{2} f(x_{\mathrm{k}}, y_{\mathrm{k}}))$$

que se deduce de la igualdad

$$y(x_{k+1}) - y(x_k) = \int_{x_k}^{x_{k+1}} y\prime(x)dx = \int_{x_k}^{x_{k+1}} f(x, y(x))dx$$

usando la fórmula del punto medio para aproximar dicha integral, y aproximando el valor $y(x_k + \frac{h}{2})$ que aparece en dicha fórmula usando el método de Euler de tamaño de paso $\frac{h}{2}$.

Es decir, usando la integración numérica:

$$\int_{x_k}^{x_{k+1}} y\prime(x)dx = \int_{x_k}^{x_{k+1}} f(x, y(x))dx$$

entonces:

$$y(x_{k+1}) - y(x_k) = \int_{x_k}^{x_{k+1}} f(x, y(x))dx$$

con $k = 0, ..., N-1$.

Supongamos $y_k \sim y(x_k)$, para $k \geq 1$ (e $y_0 = y(x_0)$ valor inicial). Sea el intervalo $[x_k, x_{k+1}]$, entonces el punto medio de dicho intervalo es $\frac{x_k + x_{k+1}}{2}$.

Sea h la longitud del intervalo $[x_k, x_{k+1}]$ y $\frac{h}{2}$ la longitud de sendos intervalos $[x_k, \frac{x_k+x_{k+1}}{2}]$ y $[\frac{x_k+x_{k+1}}{2}, x_{k+1}]$, podemos deducir:

$$\frac{x_k + x_{k+1}}{2} = x_k + \frac{h}{2}.$$

Por tanto, se obtiene:

$$y_{k+1} - y_k = (x_{k+1} - x_k) f(x_k + \tfrac{h}{2}, y_k + \tfrac{h}{2} f(x_k, y_k))$$

de donde, finalmente:

$$y_{k+1} = y_k + h f(x_k + \tfrac{h}{2}, y_k + \tfrac{h}{2} f(x_k, y_k))$$

2.1.3. Estudio del error y análisis de la convergencia

Sea la forma general de los métodos de un paso generales

$$y_{k+1} = y_k + h\Phi(x_k, y_k, h)$$

se define el **error local de discretización** en el punto x_{k+1} como la cantidad:

$$\varepsilon_{k+1}(h) = \frac{1}{h}(y(x_{k+1}) - y(x_k) - h\Phi(x_k, y(x_k), h))$$

y el **error global** como:

$$e_{k+1} = y(x_{k+1}) - y_{k+1}$$

Por construcción, el método del punto medio tiene la misma razón de convergencia que el método de Taylor de orden 2, es decir, $O(h^2)$.

Es decir, este método tiene un error local de $O(h^3)$, y global de $O(h^2)$.

Teorema 2.1. *Sea* $f : [x_0, x_0 + a]x\mathbb{R} \to \mathbb{R}$ *contínua y Lipschitziana en su segunda variable. Entonces, el método del punto medio para resolver el p.v.i.* (1) *de ecuación* $y' = f(x, y)$ *y condición inicial* $y(x_0) = \eta$*:*

- *Es estable*
- *Es consistente con el p.v.i.* $(*)$
- *Converge a la solución de* $(*)$

Además, el método del punto medio es de orden 2. Es decir, el número de veces que se evalúa la función en cada paso del método es 2.

2.1.4. Ventajas y Desventajas en comparación con los otros métodos

Dicho método, junto con los demás de Runge-Kutta, son más sencillos que los de Taylor y, además, su convergencia es rápida.

Los métodos de Taylor proporcionan una convergencia rápida, pero su implementación es complicada, ya que es necesario calcular los valores aproximados de las derivadas sucesivas de la solución. En contraposición, con los métodos de Runge-Kutta no necesitamos evaluar las derivadas sucesivas y, sin embargo, obtenemos convergencia rápida.

El método de Euler, en cambio, es muy sencillo de aplicar, pero sin embargo su convergencia es lenta. En contraposición, los métodos de Runge-Kutta son también fáciles de aplicar, pero con ellos obtenemos una convergencia rápida.

En el punto medio, el error global que se comete es de orden 2, con lo cual aunque la fórmula sea tan sencilla como la de Euler, la convergencia es mejor.

Dicho método, en comparación con los demás métodos de Runge-Kutta, podemos deducir que se usa en menos casos que, por ejemplo, Runge-Kutta de orden 4 ya que este último nos proporciona mejor orden de convergencia y menor error global y en cada paso.

2.1.5. Ejemplo y ejercicio teórico/práctico

Ejercicio 2: Sea

$$\begin{cases} y' = 1 - x + 4y \\ y(0) = 1 \end{cases}$$

queremos aproximar la solución en $x = 1$ con distintos tamaños de paso: $h = 0{,}1$ y $h = 0{,}05$.

Como comenzamos con $x_0 = 0$ y queremos llegar hasta $x = 1$, serán necesarias $\frac{|x - x_0|}{h=0{,}1} = 10$ iteraciones en este caso.

Realizando las sucesivas iteraciones con $h = 0{,}1$ y el método del punto medio obtenemos:

- $x_0 = 0$ e $y_0 = 1$
- $x_1 = 0{,}1$ e $y_1 = 1{,}595$
- $x_2 = 0{,}2$ e $y_2 = 2{,}463600000000001$
- $x_3 = 0{,}3$ e $y_3 = 3{,}737128000000001$
- $x_4 = 0{,}4$ e $y_4 = 5{,}609949440000002$
- $x_5 = 0{,}5$ e $y_5 = 8{,}369725171200003$
- $x_6 = 0{,}6$ e $y_6 = 12{,}442193253376$
- $x_7 = 0{,}7$ e $y_7 = 18{,}45744601499649$
- $x_8 = 0{,}8$ e $y_8 = 27{,}3480201021948$
- $x_9 = 0{,}9$ e $y_9 = 40{,}4940697512483$
- $x_{10} = 1$ e $y_{10} = 59{,}93822323184749$

La solución exacta es $64,897803$. Por tanto, el **error global** obtenido con $h = 0{,}1$ es: $4{,}95958$.

Para $h = 0{,}05$, serán necesarias $\frac{|x - x_0|}{h=0{,}05} = 20$ iteraciones en este caso.

Realizando las sucesivas iteraciones con $h = 0{,}05$ y el método del punto medio obtenemos:

- $x_0 = 0$ e $y_0 = 1$
- $x_1 = 0{,}05$ e $y_1 = 1{,}27375$
- $x_2 = 0{,}1$ e $y_2 = 1{,}604975$

- $x_3 = 0{,}15$ e $y_3 = 2{,}0063195$
- $x_4 = 0{,}2$ e $y_4 = 2{,}49320979$
- $x_5 = 0{,}25$ e $y_5 = 3{,}0844659438$
- $x_6 = 0{,}3$ e $y_6 = 3{,}803048451436$
- $x_7 = 0{,}35$ e $y_7 = 4{,}676969110751919$
- $x_8 = 0{,}4$ e $y_8 = 5{,}740402315117342$
- $x_9 = 0{,}45$ e $y_9 = 7{,}035040824443157$
- $x_{10} = 0{,}5$ e $y_{10} = 8{,}611749805820651$
- $x_{11} = 0{,}55$ e $y_{11} = 10{,}5325847631012$
- $x_{12} = 0{,}6$ e $y_{12} = 12{,}87325341098346$
- $x_{13} = 0{,}65$ e $y_{13} = 15{,}72611916139982$
- $x_{14} = 0{,}7$ e $y_{14} = 19{,}20386537690778$
- $x_{15} = 0{,}75$ e $y_{15} = 23{,}44396575982749$
- $x_{16} = 0{,}8$ e $y_{16} = 28{,}61413822698954$
- $x_{17} = 0{,}85$ e $y_{17} = 34{,}91899863692724$
- $x_{18} = 0{,}9$ e $y_{18} = 42{,}60817833705123$
- $x_{19} = 0{,}95$ e $y_{19} = 51{,}9862275712025$
- $x_{20} = 1$ e $y_{20} = 63{,}42469763686705$

La solución exacta es $64,897803$. Por tanto, el **error global** obtenido con $h = 0{,}05$ es: $1,473105$.

2.1.6. Método programado en Maxima

PRIMERA IMPLEMENTACIÓN DEL MÉTODO

Método del Punto Medio

```
(%i1) kill(all);
(%o0) done
```

```
Programa en Maxima del método del punto medio para y' = f(x,y) con argumento una lista formada
por xi, yi, f(y,x), h:

y(i+1) = y(i) + h*f(xi + h/2, yi + h/2 * f(xi, yi))

- La función "endcons" añade a la "listapm" los valores "valor[1]" y "valor[2]".
```

```
(%i1) PuntoMedio(argumentos):=block([xi:argumentos[1], yi:argumentos[2],
      f:argumentos[3], h:argumentos[4], K1],
          K1: ev(f, y=yi, x=xi, numer),
          [xi + h, yi + h*ev(f, y = yi + h/2 * K1, x = xi + h/2, numer), f, h])$
```

```
(%i2) valor:[0.0, 1.0, y+x, 0.1];
      listapm:[[0.0, 1.0]];
(%o2) [0.0,1.0,y+x,0.1]
(%o3) [[0.0,1.0]]
```

Figura 1: Punto Medio

```
(%i4) for i:1 thru 10 do(
          valor:PuntoMedio(valor),
          listapm:endcons([valor[1], valor[2]], listapm));
(%o4) done
```

```
(%i5) listapm;
(%o5) [[0.0,1.0],[0.1,1.11],[0.2,1.24205],[0.3,1.39846525],[0.4,1.58180410125],[0.5,1.79489353188125],[0.6,
2.040857352728782],[0.7,2.323147374765304],[0.8,2.645577849115661],[0.9,3.012363523272805],[1.0,3.42816169321645]]
```

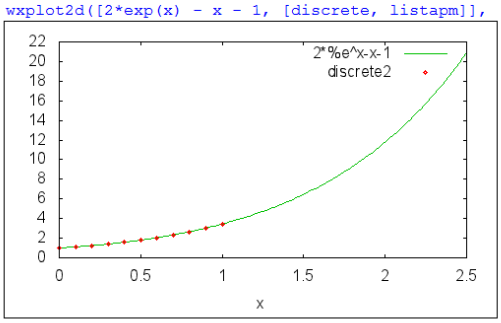
```
Devuelve una lista con los valores de (xo, yo), (x1, y1), (x2, y2), ..., (x5, y5)
por el método del punto medio para la ecuación y' = y + x preparados para representarlos en Maxima.

Si queremos dibujar los puntos tenemos que guardar en la lista sólo los pares [xi, yi] y no los
valores de f, h.
```

Figura 2: Punto Medio

```
Por ejemplo, la solución de y' = y + x con y(0) = 1 y h = 0.1 es:
y = 2exp(x) - x - 1.
```

```
(%i6) wxplot2d([2*exp(x) - x - 1, [discrete, listapm]], [x, 0, 2.5], [style, [lines, 1, 3], [points, 1, 2]]);
```




```
(%t6)

(%o6)
```

Figura 3: Punto Medio

SEGUNDA IMPLEMENTACIÓN DEL MÉTODO

```
Otra forma: guardar en la variable "h" el valor del paso y en la variable "f"
            la función "f(x,y)" antes de empezar y transmitir sólo los valores
            de xi, yi.
```

```
(%i7) kill(all);
(%o0) done
```

```
(%i1) PuntoMedio2(argumentos):=block([xi:argumentos[1], yi:argumentos[2],K1],
                K1:ev(f, y=yi, x=xi, numer),
                [xi + h, yi + h * ev(f, y = yi + h/2 * K1, x = xi + h/2, numer)])$
```

Figura 4: Punto Medio

```
Mismo ejemplo con h = 0.05
```

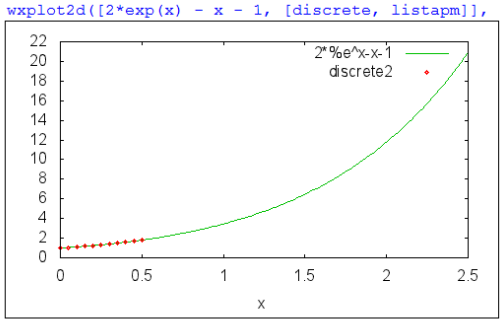
```
(%i2) h: 0.05;
      f: y+x;
      valor: [0.0,1.0];
      listapm: [[0.0,1.0]];
(%o2) 0.05
(%o3) y+x
(%o4) [0.0,1.0]
(%o5) [[0.0,1.0]]
```

```
(%i6) for i:1 thru 10 do(
          valor:PuntoMedio2(valor),
          listapm:endcons(valor,listapm));
(%o6) done
```

```
(%i7) listapm;
(%o7) [[0.0,1.0],[0.05,1.0525],[0.1,1.110253125],[0.15,1.17352859765625],[0.2,1.242609438286133],[0.25,1.317793171998297]
,[0.3,1.39939257206321],[0.35,1.487736441381449],[0.4,1.583170434002249],[0.45,1.686057918744864],[0.5,1.796780887080538]]
```

Figura 5: Punto Medio

```
(%i8) wxplot2d([2*exp(x) - x - 1, [discrete, listapm]], [x, 0, 2.5], [style, [lines, 1, 3], [points, 1, 2]]);
```




```
(%t8)
(%o8)
```

Figura 6: Punto Medio

TERCERA IMPLEMENTACIÓN DEL MÉTODO

```
Otra forma: calculando K1 y K2 por separado.
```

```
(%i9) kill(all);
(%o0) done
```

```
(%i1) PuntoMedio3(x0, b, h, y0, expre, x, y):=(
          local(f),
          define(f(x,y), expre),
          N: abs(b-x0)/h,
          puntos:makelist([0,0], k, 0, N),
          xn:x0,
          yn:y0,
          puntos[1]:[x0,y0],
          for k:1 thru N do(
              K1: f(xn, yn),
              K2: f(xn + (1/2)*h, yn + (h/2)*K1),
              yn: yn + h * (K2),
              xn: xn + h,
              puntos[k+1]:[xn, yn]
          ),
          puntos
      )$
```

Figura 7: Punto Medio

```
Ejemplo: Aplicar el método del Punto Medio a:
            y' = 1 - x + 4y
            y(0) = 1
para aproximar la solución en x = 1 con h = 0.1 y h = 0.05.
```

```
(%i2) PuntoMedio3(0, 1, 0.1, 1, 1 - x + 4*y, x, y);
(%o2) [[0,1],[0.1,1.595],[0.2,2.463600000000001],[0.3,3.737128000000001],[0.4,5.609949440000002],[0.5,8.369725171200003],
[0.6,12.442193253376],[0.7,18.45744601499649],[0.8,27.3480201021948],[0.9,40.4940697512483],[1.0,59.93822323184749]]
```

```
(%i3) PuntoMedio3(0, 1, 0.05, 1, 1 - x + 4*y, x, y);
(%o3) [[0,1],[0.05,1.27375],[0.1,1.604975],[0.15,2.0063195],[0.2,2.49320979],[0.25,3.0844659438],[0.3,3.803048451436],[
0.35,4.676969110751919],[0.4,5.740402315117342],[0.45,7.035040824443157],[0.5,8.611749805820651],[0.55,10.5325847631012],[
0.6,12.87325341098346],[0.65,15.72611916139982],[0.7,19.20386537690778],[0.75,23.44396575982749],[0.8,28.61413822698954],[
0.85,34.91899863692724],[0.9,42.60817833705123],[0.95,51.9862275712025],[1.0,63.42469763686705]]
```

Figura 8: Punto Medio

```
Primer ejemplo con esta última implementación del método.
Por ejemplo, para aproximar la solución en x = 2.
```

```
(%i4) PuntoMedio3(0, 2, 0.1, 1, y+x, x, y);
(%o4) [[0,1],[0.1,1.11],[0.2,1.24205],[0.3,1.39846525],[0.4,1.58180410125],[0.5,1.79489353188125],[0.6,2.040857352728782
],[0.7,2.323147374765304],[0.8,2.645577849115661],[0.9,3.012363523272805],[1.0,3.42816169321645],[1.1,3.898118671004177],[
1.2,4.427921131459615],[1.3,5.023852850262875],[1.4,5.692857399540477],[1.5,6.442607426492227],[1.6,7.281581206273911],[1.7
,8.219147232932672],[1.800000000000001,9.265657692390603],[1.900000000000001,10.43255175009162],[2.0,11.73246968385124]]
```

```
(%i5) PuntoMedio3(0, 2, 0.05, 1, y+x, x, y);
(%o5) [[0,1],[0.05,1.0525],[0.1,1.110253125],[0.15,1.17352859765625],[0.2,1.242609438286133],[0.25,1.317793171998297],[
0.3,1.39939257206321],[0.35,1.487736441381449],[0.4,1.583170434002249],[0.45,1.686057918744864],[0.5,1.796780887080538],[
0.55,1.915740907543416],[0.6,2.043360129055016],[0.65,2.180082335669086],[0.7,2.326374055372126],[0.75,2.482725725709948],[
0.8,2.649652919152582],[0.85,2.827697631259152],[0.9,3.017429634861184],[0.95,3.21944790364782],[1.0,3.434382108709771],[
1.05,3.662894191781147],[1.1,3.905680019109931],[1.15,4.163471120089314],[1.2,4.437036514993891],[1.25,4.727184636387328],[
1.300000000000001,5.034765349002178],[1.350000000000001,5.360672073138541],[1.400000000000001,5.705844016886891],[
1.450000000000001,6.071268522752344],[1.500000000000001,6.457983534543401],[1.550000000000001,6.867080190688751],[
1.600000000000001,7.299705550461551],[1.650000000000001,7.757065459922704],[1.700000000000001,8.240427564743744],[
1.750000000000001,8.751124477436861],[1.800000000000001,9.290557106905499],[1.850000000000001,9.860198158634406],[
1.900000000000001,10.46159581426442],[1.950000000000001,11.09637759974547],[2.000000000000001,11.76625445173243]]
```

Figura 9: Punto Medio

2.2. Método de Runge-Kutta de orden 2

El método de **Runge-Kutta de orden 2** es una categoría que abarca diferentes métodos todos ellos haciendo uso de dos evaluaciones.

2.2.1. Método de Euler modificado

Intenta refinar la aproximación a la pendiente de la función mediante la aplicación de dos derivadas en el intervalo, una en el punto inicial y otra en punto final. La aproximación mejorada de la pendiente será el promedio de las dos derivadas.
Presenta la siguiente **tabla de Butcher**

0		
1	1	
	1/2	1/2

Definimos el método como sigue:

$$[RK2] \begin{cases} y_0 = y(x_0) \\ y_{k+1} = y_k + \frac{h}{2}(f(x_k, y_k) + f(x_k + h, y_k + hf(x_k, y_k))), k = 0, 1, ..., N-1 \end{cases}$$

donde:

- $h = \frac{a}{N}, N \in \mathbb{N}$
- $x_k = x_0 + kh, k = 0, ..., N$
- $\phi(x, y, h) = \frac{1}{2}(K_1 + K_2)(x, y, h)$ con
  - $K_1(x, y, h) = f(x, y)$
  - $K_2(x, y, h) = f(x + h, y + hK_1(x, y, h))$

2.2.2. Método de Heun

Este es también un método de segundo orden, que en las condiciones anteriores, tiene la forma:

$$[RK2,2] \begin{cases} y_0 = y(x_0) \\ y_{k+1} = y_k + \frac{h}{4}(f(x_k, y_k) + 3f(x_k + \frac{2}{3}h, y_k + \frac{2}{3}hf(x_k, y_k))), k = 0, 1, ..., N-1 \end{cases}$$

donde:

- $h = \frac{a}{N}, N \in \mathbb{N}$
- $x_k = x_0 + kh, k = 0, ..., N$
- $\phi(x, y, h) = \frac{1}{4}(K_1 + 3K_2)(x, y, h)$ con
  - $K_1(x, y, h) = f(x, y)$
  - $K_2(x, y, h) = f(x + \frac{2}{3}h, y + \frac{2}{3}hK_1(x, y, h))$

2.2.3. Análisis de la convergencia y estudio del error

Estudio de la convergencia

Teorema 2.2. *Sea $f : [x_0, x_0 + a] \times \mathbb{R} \to \mathbb{R}$ contínua, y lipschitziana en su segunda variable. Entonces el método de Runge-Kutta de orden 2 para resolver el p.v.i.$(*)$ de ecuación $y' = f(x, y)$ y condición inicial $y(x_0) = \eta$ es*

- *Es estable*
- *Es consistente con $(*)$*
- *Converge a la solución de $(*)$*

Además, el método $[RK2]$ es de orden 2.

Análisis del error
Sean y_k el valor exacto de la función y $\overline{y_k}$ el valor aproximado conseguido con $k = 1, ..., N$. Entonces:

- Se define el *error de truncamiento del paso j* como $e_j = y_j - \overline{y_j}$
- Se define el *error de truncamiento global* como $e_n = y_n - \overline{y_n}$

En cuanto al error podemos decir que por su definición el método RK2 tiene la misma razón de convergencia que el método de Taylor de orden 2, es decir, $O(h^2)$.

Es decir, este método tiene un error local de $O(h^3)$, y global de $O(h^2)$.

Observaciones De nuevo este método presenta una ventaja sobre los métodos de *Taylor*, evitando el cálculo de las derivadas y la evaluación de las mismas. Es decir conlleva un cálculo computacionalmente menos pesado.

2.2.4. Programación [RK2] y ejercicio numérico

Pseudocódigo para el método de Euler modificado.

$RK2(a, b, h, y(a), f)$
$N \leftarrow (b-a)/h$
$x_0 \leftarrow a$
$y_0 \leftarrow y(a)$
$for(i = 0; i < N; i++)\{$
$K_1 \leftarrow f(x_i, y_i)$
$p \leftarrow y_i + h * K_1$
$x_{i+1} \leftarrow a + (i+1) * h$
$K_2 \leftarrow f(x_{i+1}, p)$
$y_{i+1} \leftarrow y_i + \frac{h}{2}(K_1 + K_2)$
$\}$

Por lo tanto el código queda como sigue:

```
--> rk2(a,b,N,y0,f):=block([],
    h:float((b-a)/N),
    x[0]:a,
    y[0]:y0,
    for i:0 thru N do(
        K1:f(x[i],y[i]),
        p:y[i] + h*K1,
        x[i+1]:a + (i+1)*h,
        K2:f(x[i+1],p),
        y[i+1]:y[i]+h/2*float(K1+K2),
        print("Iteracion",i,"intervalo ",x[i+1],"resultado ",y[i+1])
    )
    );
```

Figura 10: Metodo de Euler modificado

Pseudocódigo para el método de Heun

$RK2(a, b, h, y(a), f)$
$N \leftarrow (b-a)/h$
$x_0 \leftarrow a$
$y_0 \leftarrow y(a)$
$for(i = 0; i < N; i++)\{$
$K_1 \leftarrow f(x_i, y_i)$
$p \leftarrow y_i + \frac{2}{3}h * K_1$
$x_{i+1} \leftarrow a + (i+1) * h$
$K_2 \leftarrow 3f(x_i + \frac{2}{3}h, p)$
$y_{i+1} \leftarrow y_i + \frac{h}{4}(K_1 + K_2)$
$\}$

```
Heun(a,b,N,y0,m):=block([],
fpprec:10,
h:abs(b-a)/N,
x[0]:a,
y[0]:y0,
for i:0 thru N do(
    K1:m(x[i],y[i]),
    x[i+1]:x[0]+h*(i+1),
    p:y[i]+(2/3)*h*K1,
    K2:m(x[i]+(2/3)*h,p),
    y[i+1]:y[i]+h/4*float((K1+3*K2)),
    print("Iteracion",i,"punto ",x[i],"resultado ",y[i])
)
);
```

Figura 11: Metodo de Heun

Ejemplo: veamos un simple ejemplo que muestra la convergencia del método de Euler modificado.

Dado el siguiente p.v.i.:

$$\begin{cases} y'(x) = f(x,y) = y + 2x - x^2 \\ y(0) = -1 \end{cases}$$

Aproximar el valor de $y(1)$ fijado un paso $h = 0{,}2$

Ejecutando el método obtenemos los siguientes valores:

x_n	x_n	$y_n(x_n)$	$\lvert y_n - \overline{y_n}\rvert$
0	-1.0000	-1.000	0
0.2000	-1.1840	-1.1814	0.0026
0.4000	-1.3373	-1.3318	0.0055
0.6000	-1.4707	-1.4621	0.0086
0.8000	-1.5974	-1.5855	0.0119
1.0000	-1.7337	-1.7183	0.0154

Otro ejemplo:

Dada el siguiente p.v.i.:

$$\begin{cases} y'(x) = 2xy \\ y(1) = 1 \end{cases}$$

Aproximar el valor de $y(1{,}5)$ fijado un paso $h = 0{,}1$

```
(%i47) f(x,y):=2*x*y$
       rk2(1,1.5,5,1,f);
Iteracion 0 intervalo 1 resultado 1
Iteracion 1 intervalo 1.1 resultado 1.232
Iteracion 2 intervalo 1.2 resultado 1.5478848
Iteracion 3 intervalo 1.3 resultado 1.98315000576
Iteracion 4 intervalo 1.4 resultado 2.590787167524864
Iteracion 5 intervalo 1.5 resultado 3.450928507143119
(%o48) done
```

Figura 12: Ejecución RK2

2.3. Método de Runge-Kutta de orden 4

El **Método de Runge-Kutta de orden 4** (también llamado método de *Runge-Kutta clásico* o *Runge-Kutta de 4 evaluaciones*) viene dado por la siguiente **tabla de Butcher**

$$\begin{array}{c|cccc} 0 & & & & \\ 1/2 & 1/2 & & & \\ 1/2 & 0 & 1/2 & & \\ 1 & 0 & 0 & 1 & \\ \hline & 1/6 & 2/6 & 2/6 & 1/6 \end{array}$$

por tanto, el método se define como:

$$[RK4] \begin{cases} y_0 = y(x_0) \\ y_{k+1} = y_k + \frac{h}{6}\phi(x_k, y_k, h), k = 0, 1, ..., N-1 \end{cases}$$

donde

- $h = \frac{a}{N}, N \in \mathbb{N}$
- $x_k = x_0 + kh, k = 0, ..., N$
- $\phi(x, y, h) = (K_1 + 2K_2 + 2K_3 + K_4)(x, y, h)$ con
  - $K_1(x, y, h) = f(x, y)$

- $K_2(x,y,h) = f(x+\frac{h}{2}, y+\frac{h}{2}K_1(x,y,h))$
- $K_3(x,y,h) = f(x+\frac{h}{2}, y+\frac{h}{2}K_2(x,y,h))$
- $K_4(x,y,h) = f(x+h, y+hK_3(x,y,h))$

2.3.1. Análisis de convergencia y estudio del error

Estudio de la convergencia
En nuestro caso tenemos que, dado el método

$$[RK4] \begin{cases} y_0 = y(x_0) \\ y_{k+1} = y_k + \frac{h}{6}\phi(x_k, y_k, h), k = 0, 1, ..., N-1 \end{cases}$$

donde

- $h = \frac{a}{N}, N \in \mathbb{N}$
- $x_k = x_0 + kh, k = 0, ..., N$
- $\phi(x,y,h) = (K_1 + 2K_2 + 2K_3 + K_4)(x,y,h)$ con
  - $K_1(x,y,h) = f(x,y)$
  - $K_2(x,y,h) = f(x+\frac{h}{2}, y+\frac{h}{2}K_1(x,y,h))$
  - $K_3(x,y,h) = f(x+\frac{h}{2}, y+\frac{h}{2}K_2(x,y,h))$
  - $K_4(x,y,h) = f(x+h, y+hK_3(x,y,h))$

Si f es *Lipschitziana* en su segunda variable, entonces el método $[RK4]$ para resolver el p.v.i. $(*)$:

- Es estable
- Es consistente con $(*)$
- Converge a la solución de $(*)$

Además, el método $[RK4]$ es de orden 4.

Estudio del error
Se cumple que, dado el método

$$[RK4] \begin{cases} y_0 = y(x_0) \\ y_{k+1} = y_k + \frac{h}{6}\phi(x_k, y_k, h), k = 0, 1, ..., N-1 \end{cases}$$

donde

- $h = \frac{a}{N}, N \in \mathbb{N}$
- $x_k = x_0 + kh, k = 0, ..., N$
- $\phi(x, y, h) = (K_1 + 2K_2 + 2K_3 + K_4)(x, y, h)$ con
  - $K_1(x, y, h) = f(x, y)$
  - $K_2(x, y, h) = f(x + \frac{h}{2}, y + \frac{h}{2}K_1(x, y, h))$
  - $K_3(x, y, h) = f(x + \frac{h}{2}, y + \frac{h}{2}K_2(x, y, h))$
  - $K_4(x, y, h) = f(x + h, y + hK_3(x, y, h))$

Llamemos ε_0 al error de redondeo cometido en el dato inicial y, para $k = 0, ..., N$, llamemos ε_k al error de redondeo cometido en el paso k, es decir:

- $z_0 = y_0 + \varepsilon_0$
- $z_{k+1} = z_k + \frac{h}{6}\phi(x_k, z_k, h) + \varepsilon_k$

Se cumple que si ϕ es *Lipschitziana* en su segunda variable con constante de Lipschitz igual a M se verifica que

$|E_k| \leq e^{x_k - x_0}M|\varepsilon_0| + \frac{e^{(x_k - x_0)M} - 1}{M}(\tau + \frac{\varepsilon}{h})$

donde $E_k = z_k - y(x_k)$, $\tau = \text{máx}_{k=0,1,...,N-1} \frac{|u_{k+1}|}{h}$, $\varepsilon = \text{máx}_{k=0,1,...,N-1} |\varepsilon_k|$

2.3.2. Ventajas y desventajas

Ventajas:
En comparación con los métodos de *Taylor*, el método $[RK4]$ evita el cálculo de las derivadas, por lo que resulta computacionalmente menos pesado.
En comparación con los demás métodos de *Runge-Kutta* se tiene que el error por paso es del orden de $O(h^5)$, mientras que el error acumulado es del orden de $O(h^4)$. Así, el orden de convergencia es de $O(h^4)$. Esta es la razón por la que $[RK4]$ es el método computacional más usado.

2.3.3. Ejercicio teórico-práctico

Dada el siguiente p.v.i.:

$$\begin{cases} x'(t) = -x + (t-1)y \\ y'(t) = x - ty \end{cases}$$

Con condiciones iniciales:

$$\begin{cases} x(0) = 0'483941 \\ y(0) = 0'682689 \end{cases}$$

Utilizamos el método de Runge-Kutta se orden 4 para hallar los valores de $x(t)$ e $y(t)$ para $t = 2\sqrt{2} - 1 = 1'82843$

Por simplificar, vamos a calcular dicha aproximación en un solo paso, de forma que, como las condiciones iniciales son dadas para un tiempo $t_0 = 0$, el valor de h debe ser $h = 2\sqrt{2} - 1 = 1'82843$. Nuestro método por tanto consiste en aplicar la siguiente iteración:

$\binom{x_{i+1}}{y_{i+1}} = \binom{x_i}{y_i} + \frac{h}{6}\binom{r_1+2r_2+2r_3+r_4}{k_1+2k_2+2k_3+k_4}$

donde

$r_1 = f(t_y, x_i, y_i)$
$k_1 = g(t_i, x_i, y_i)$
$r_2 = f(t_i + \frac{h}{2}, x_i + \frac{h}{2}r_1, y_i + \frac{h}{2}k_1)$
$k_2 = g(t_i + \frac{h}{2}, x_i + \frac{h}{2}r_1, y_i + \frac{h}{2}k_1)$
$r_3 = f(t_i + \frac{h}{2}, x_i + \frac{h}{2}r_2, y_i + \frac{h}{2}k_2)$
$k_3 = g(t_i + \frac{h}{2}, x_i + \frac{h}{2}r_2, y_i + \frac{h}{2}k_2)$
$r_4 = f(t_i + h, x_i + hr_3, y_i + hk_3)$
$k_4 = g(t_i + h, x_i + hr_3, y_i + hk_3)$

De esta forma, sustituyendo $h = 2\sqrt{2} - 1 = 1'82843$ nos queda que:

$r_1 = 0'198748$
$k_1 = 0'483941$
$r_2 = 1'48807$
$k_2 = -0'362958$
$r_3 = -1'17272$
$k_3 = 1'52359$
$r_4 = 11'4706$
$k_4 = -8'00215$

Y sustituyendo en la iteración dada por el método de Runge-Kutta se obtiene:

$x(2\sqrt{2}-1) = x(1'82843) \approx x_1 = 4'23224$
$y(2\sqrt{2}-1) = y(1'82843) \approx y_1 = 0'90102$

2.3.4. Programación de [RK4] y ejercicio numérico

Veamos a continuación el pseudocódigo del método:

$RK4(a, b, h, y(a), f)$
$N \leftarrow (b-a)/h$
$x_0 \leftarrow a$
$y_0 \leftarrow y(a)$
$for(i = 0; i < N; i++)\{$
$x_i \leftarrow a + i * h$
$K_1 \leftarrow h * f(x_i, y_i)$
$K_2 \leftarrow h * f(x_i + \frac{1}{2}h, y_i + \frac{1}{2}K_1)$
$K_3 \leftarrow h * f(x_i + \frac{1}{2}h, y_i + \frac{1}{2}K_2)$
$K_4 \leftarrow h * f(x_i + h, y_i + K_3)$
$y_{i+1} \leftarrow y_i + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4)$
$\}$

Visto esto, construimos el método en *Maxima* para más tarde resolver un p.v.i. sencillo a modo de ejemplo. La construcción se encuentra en la Figura 13.

```
RK4(x0,b,h,y0,expre,x,y):=(
    local(f),
    define(f(x,y),expre),
    N:abs(b-x0)/h,
    puntos:makelist([0,0],k,0,N),
    xn:x0,
    yn:y0,
    puntos[1]:[x0,y0],
    for k:1 thru N do(
        K1:h*f(xn,yn),
        K2:h*f(xn+(1/2)*h,yn+(1/2)*K1),
        K3:h*f(xn+(1/2)*h,yn+(1/2)*K2),
        K4:h*f(xn+h,yn+K3),
        yn:yn+(1/6)*(K1+2*K2+2*K3+K4),
        xn:xn+h,
        puntos[k+1]:[xn,yn]
    ),
    puntos
)$
```

Figura 13: RK4

Resolvamos a continuación un sencillo p.v.i.:

Dada el siguiente p.v.i.:

$$\begin{cases} y'(x) = 2xy \\ y(1) = 1 \end{cases}$$

Aproximar el valor de $y(1{,}5)$ fijado un paso $h = 0{,}1$

Tras la ejecución en *Maxima* obtenemos:
$y(1) = 1,0000$ (solución exacta: 1,0000)
$y(1{,}1) = 1,2337$ (solución exacta: 1,2337)
$y(1{,}2) = 1,5527$ (solución exacta: 1,5527)
$y(1{,}3) = 1,9937$ (solución exacta: 1,9937)
$y(1{,}4) = 2,6116$ (solución exacta: 2,6117)
$y(1{,}5) = 3,4902$ (solución exacta: 3,4904)

Con este mismo p.v.i. podemos ver una **comparativa** de los tres métodos estudiados, expresada en la siguiente tabla:

Valor	h	*Pto medio*	*Euler2*	*Heun*	*RK4*	*Err MPM*	*Err Euler*	*ErrHeun*	*ErrRK4*
1,0000	1,0	1,0000	1.0000	1,0000	1,0000	0	0	0	0
1,2337	1,1	1.231	1,232	1.2313	1,2337	0,0027	0.0017	0.004	0
1,5527	1,2	1,5453	1,5479	1.5461	1,5527	0,0074	0.0048	0,0066	0
1,9937	1,3	1,9779	1,9832	1.9797	1,9937	0.0158	0.0105	0,014	0
2,6117	1,4	2,5814	2,5908	2.5845	2,6116	0,303	0,0209	0,0272	0,0001
3,4904	1,5	3,4348	3,4509	3.4402	3,4902	0,0556	0,0395	0.0502	0.0002

2.4. Articulo

El articulo *"Zero-finder methods derived using Runge–Kutta techniques"* hace un estudio sobre la posibilidad de integrar los métodos de Runge-Kutta en un problema iterativo donde se busca el cero de la función. Runge-Kutta entra en juego cuándo se están tratando las derivadas de la función. Por lo tanto, se intenta simplificar el problema creando uno que ya conocemos como es el problema del valor inicial. La idea consiste en resolver una función diferencial que es a su vez inversa de una función donde al computar el cero obtenemos el valor inicial.

En consecuencia el problema toma la siguiente forma. Sea f una función con raíz simple α, $f(\alpha) = 0$ teniendo que $f\prime(\alpha) \neq 0$ en un entorno I_α de α. Entonces existe la inversa de la función $x = g(y)$ en $J_0 \subseteq f(I_\alpha)$ con $g(0) = \alpha$. Si derivamos la función $x(y)$ respecto de la y se obtiene:

$\frac{dx}{dy} = g\prime(y) = \frac{1}{f\prime(x)}$

Puediendo expresarlo de la forma: $x\prime(y) = F(x)$ siendo $F(x) = \frac{1}{f\prime(x)}$

Esto supone que, la ecuación $f(x) = 0$ con una aproximación inicial x_0 puede resolverse como un problema de valor inicial con $x\prime(y) = F(x)$ siendo la condición inicial $y = y_0 = f(x_0)$. El intervalo de trabajo tiene extremos y_0 y 0 ya que $g(y_0) = x_0$ y $g(0) = \alpha$.

En el estudio presentado se utilizan métodos de runge-Kutta de ordenes 1, 2 y 3 que combinándolo con el método iterativo de Newton mejoran el error cometido. Cabe destacar que a la hora de aplicar los diferentes métodos de Runge-Kutta se exige siempre que la raíz sea simple y que no se anule la derivada hasta un cierto grado, recordemos que tratamos con la inversa de una función diferencial, en ningún caso puede ser cero.

En resumen, el articulo propone una herramienta más para solucionar problemas que tratan con ecuaciones no lineales por medio de ecuaciones diferenciales. Confluyen así, en cierta medida, dos de los temas más importantes tratados en la asignatura.

3. Referencias

Referencias

[1] Análisis Matemático para Ingeniería. M. Molero; A. Salvador; T. Menarguez; L. Garmendia. Capitulo 13, pag. 36-53.
http://www2.caminos.upm.es/Departamentos/matematicas/Fdistancia/PIE/Analisis%20matematico/Temas/C13_Metodo_Numerico.pdf

[2] Apuntes del Tema 3 de la asignatura Métodos Numérico II de los profesores Teresa E. Pérez y Miguel Piñar. Curso 14/15.

[3] Apuntes del Tema 3 de la asignatura Métodos Numérico II de la profesora M. Carmen Serrano Pérez. Curso 13/14.

[4] Métodos numéricos con software libre: Maxima. María Santos Bruzón Gallego, José Ramírez Labrador. Capítulo 8, pág 156-159.

[5] Facultad Regional de San Nicolás: Métodos Numéricos para EDOs. Problemas de valor inicial y problemas de frontera.
http://www.frsn.utn.edu.ar/gie/an/mnedo/34_RK.html

[6] Métodos numéricos: problemas resueltos y prácticas. Isaac A. García, Susanna Maza.

[7] Bibligrafía recomendada en la hoja del trabajo.

[8] Zero-finder methods derived using Runge–Kutta techniques. Miquel Grau-Sánchez , José Luis Díaz-Barrero. Technical Universidad de Cataluña, Departmento de Matemáticas Aplicadas.
http://www.sciencedirect.com/science/article/pii/S0096300310011926