Sistemas y ED de orden 2 con WxMaxima

1 Sistemas de ecuaciones diferenciales

Se pueden calcular soluciones exactas utilizando el comando "desolve".

EJEMPLO (ejercicio 7): Resolver
x' = -2x + 4y
y' = x - 2y
con dato inicial x(0)=200, y(0)=500

(%i1) kill(all);ratprint:false;

(%o0) done
(%o1) false

Paso 1: Introducir las ecuaciones

(%i2) ed1:'diff(x(t),t)=-2*x(t)+4*y(t);

$$(\%o2)\quad \frac{d}{dt}x(t)=4\,y(t)-2\,x(t)$$

(%i3) ed2:'diff(y(t),t)=x(t)-2*y(t);

$$(\%o3)\quad \frac{d}{dt}y(t)=x(t)-2\,y(t)$$

Paso 2: usar el comando "desolve" (o la pestaña "Resolver EDO con Laplace")

(%i4) desolve([ed1,ed2],[x(t),y(t)]);

$$(\%o4)\quad [x(t)=\frac{2\,y(0)+x(0)}{2}-\frac{(2\,y(0)-x(0))\,\%e^{-4\,t}}{2},y(t)=\frac{(2\,y(0)-x(0))\,\%e^{-4\,t}}{4}+\frac{2\,y(0)+x(0)}{4}]$$

NOTA: "desolve" tiene una sintaxis distinta que "ode2":
- las funciones deber aparecer como x(t), y(t),...
- si escribís "x" en lugar de "x(t)" os dará error.

Paso 3: sustituir los valores iniciales, por ejemplo con "ev"

(%i5) ev(%,x(0)=200,y(0)=500);

$$(\%o5)\quad [x(t)=600-400\,\%e^{-4\,t},y(t)=200\,\%e^{-4\,t}+300]$$

Paso 4: definir las funciones y dibujar la solución

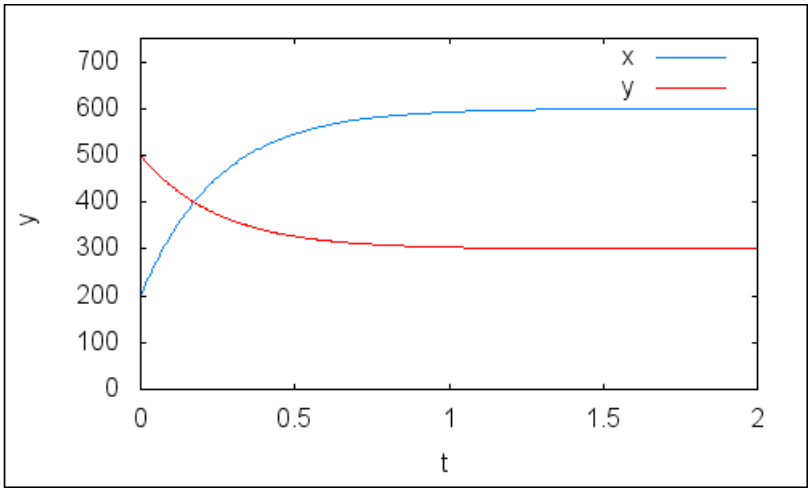
(%i6) x(t):=600-400*%e^(-4*t);y(t):=200*%e^(-4*t)+300;

$$(\%o6)\quad x(t):=600-400\,\%e^{(-4)\,t}$$

$$(\%o7)\quad y(t):=200\,\%e^{(-4)\,t}+300$$

(%i8) wxplot2d([x(t),y(t)], [t,0,2],[y,0,750],[legend,x,y])$

(%t8)



2 Ecuaciones diferenciales de orden 2

Se pueden calcular usando ode2

Ejemplo 1 (ejercicio 12a): resolver con "ode2"
x''(t) + 5x'(t) + 6x(t) = 5sin(t)
con x(0)=1, x'(0)=0

(%i9) ed1:'diff(x,t,2)+5*'diff(x,t)+6*x=5*sin(t);

$$(\%o9)\quad \frac{d^2}{d\,t^2}\,x+5\left(\frac{d}{d\,t}\,x\right)+6\,x=5\,\sin(t)$$

(%i10) ode2(ed1, x, t);

$$(\%o10)\quad x=\frac{\sin(t)-\cos(t)}{2}+\%k1\,\%e^{-2\,t}+\%k2\,\%e^{-3\,t}$$

la sustitución del valor inicial debe hacerse en la pestaña
"Problema de valor inicial(2)"

(%i11) ic2(%, t=0, x=1, 'diff(x,t)=0);

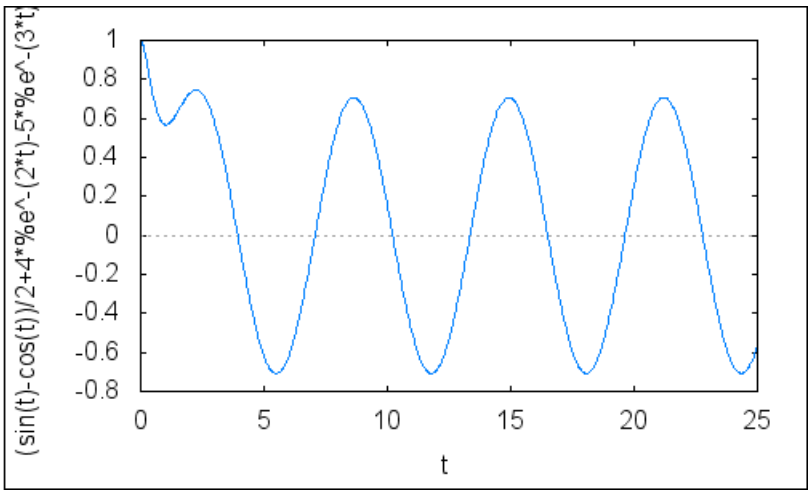
$$(\%o11)\quad x=\frac{\sin(t)-\cos(t)}{2}+4\,\%e^{-2\,t}-\frac{5\,\%e^{-3\,t}}{2}$$

(%i12) x(t):=(sin(t)-cos(t))/2+4*%e^(-2*t)-(5*%e^(-3*t))/2;

(%o12) $$x(t):=\frac{\sin(t)-\cos(t)}{2}+4\ \%e^{(-2)\,t}+\frac{-5\ \%e^{(-3)\,t}}{2}$$

(%i13) wxplot2d([x(t)], [t,0,25])$

(%t13)



a largo plazo la solución oscila entre 0.7 y -0.7, con frecuencia 2*pi

3 Soluciones numéricas con Runge-Kutta

- Runge-Kutta es el método numérico "por excelencia" para resolver casi cualquier ED...
- Los resultados son listas de números, que hay que definir y dibujar con plot
(como hicimos para dibujar recurrencias con Maxima).
- el comando se usa de la forma
rk([ed1,ed2],[x,y],[x0,y0],[t,t0,t1,h])
donde
- [ed1,ed2] son las EDs
- [x,y] son los nombres de las incógnitas
- [x0,y0] son los valores iniciales
- [t,t0,t1,h] son t = variable independiente,
t0-t1 = intervalo de tiempo que queremos calcular
"h" = paso del método numérico (típicamente ponemos h=0.01 ó menor)

- El resultado será una lista de puntos [n,x[n],y[n]] que podemos dibujar con plot2d

NOTA: Recordad que un método numérico
(i) es más exacto cuanto más pequeño sea el paso "h"
(ii) calcula la solución "paso a paso": si la queremos para t en [0,10]
y escogemos h=0.01 necesitará 1000 pasos para llegar a t=10.
Si escogemos h=10^(-5), necesitará 1 millón de pasos...

Por tanto, hay buscar un "equilibrio" entre tamaño h <--> nº pasos
para que la solución sea lo más exacta posible sin que el ordenador se cuelgue...

EJEMPLO 1: ED de orden 1 con RK

Resolver x' = -x^2 + t^2 con dato inicial x(0)=10

(%i14) kill(all);ratprint:false;

```
(%o0) done
(%o1) false
```

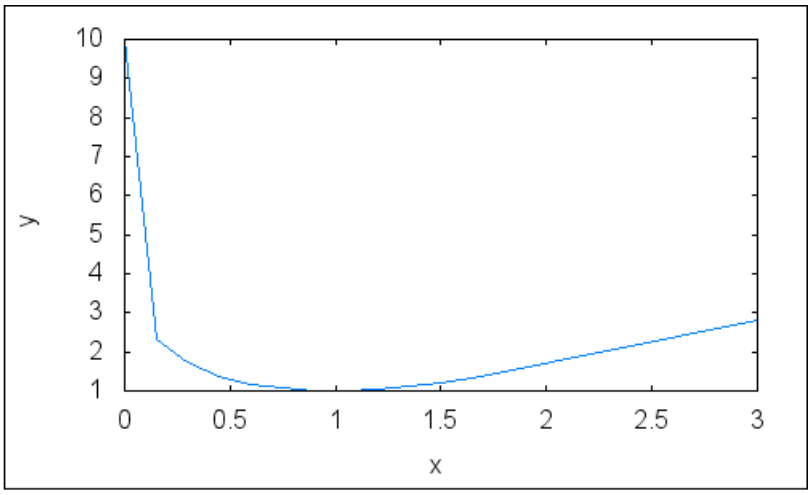
Para ilustrar el comando, tomamos primero un "paso grande" h=0.15

```
(%i2) sol1:rk([-x^2+ t^2],[x],[10],[t,0,3,0.15]);
```

```
(%o2) [[0,10],[0.15,2.31709536085083],[0.3,1.725962566131291],
[0.45,1.388834959127041],[0.6,1.18483176059432],[0.75,
1.065385474899596],[0.9,1.007946032043621],[1.05,
1.000685237968554],[1.2,1.036227404258969],[1.35,
1.10868660651974],[1.5,1.212252144059206],[1.65,
1.340716117649185],[1.8,1.487641587222746],[1.95,
1.646886879487973],[2.1,1.813180274079385],[2.25,
1.982499697262143],[2.4,2.152154443101142],[2.55,
2.320612323508032],[2.7,2.487194353381727],[2.85,
2.651757655109896],[3.0,2.814440433891635]]
```

```
(%i3) wxplot2d([discrete,sol1]);
```

(%t3)



(%o3)

veamos como funciona mejor h=0.01

```
(%i4) sol2:rk([-x^2+t^2],[x],[10],[t,0,3,0.01])$
```

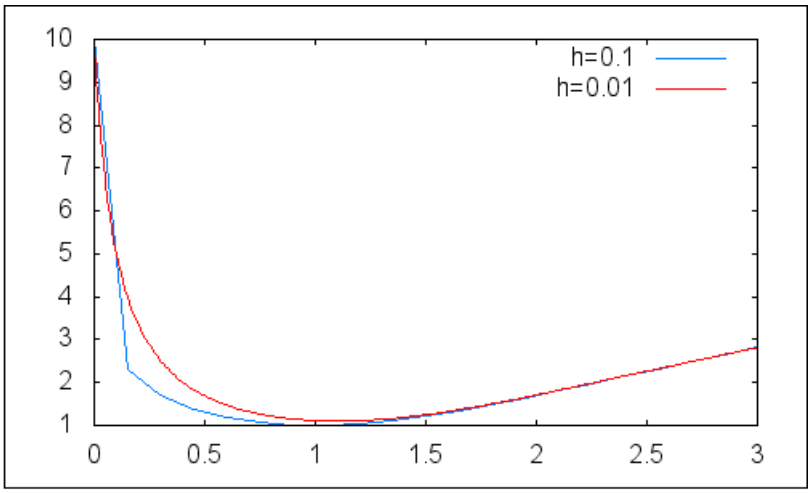
IMPORTANTE:
- al final de línea poned "$" (en lugar de ";") y no sacará en pantalla la lista de datos
  (que pueden ser miles)
- podeis calcular la longitud de la lista con "length"

```
(%i5) length(sol2);
```

```
(%o5) 301
```

```
(%i6) wxplot2d([[discrete,sol1], [discrete, sol2]],[legend, "h=0.1", "h=0.01"]);
```

(%t6)



(%o6)

Notar que la segunda solución (con paso menor) es más suave y más exacta

EJEMPLO 2: SISTEMAS de ED con RK

Resolver

x' = -2x + 4y
y' = x - 2y

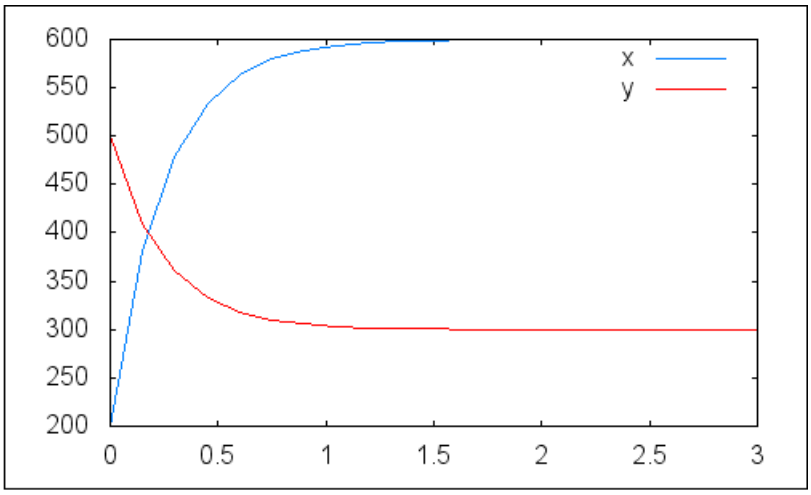
con dato inicial x(0)=200, y(0)=500

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

 (%o0) done

(%i1) sol1:rk([-2*x+ 4*y,x-2*y],[x,y],[200,500],[t,0,3,0.15]);

 (%o1) [[0,200,500],[0.15,380.24,409.88],[0.3,479.263856,
360.368072],[0.45,533.6675624864,333.1662187568],[0.6,
563.5569588300282,318.2215205849859],[0.75,579.9781931812175,
310.0109034093912],[0.9,589.0000193337609,305.4999903331196],[
1.05,593.9566106219683,303.0216946890159],[1.2,596.6797618757094
,301.6601190621453],[1.35,598.1758611745147,300.9120694127427],[
1.5,598.9978181292784,300.5010909353608],[1.65,599.4494012802255
,300.2752993598872],[1.8,599.697501063356,300.151249468322],[
1.95,599.8338070842078,300.0830964578961],[2.1,599.9086936120638
,300.0456531939681],[2.25,599.9498362704678,300.0250818647661],[
2.4,599.972440046995,300.0137799765025],[2.55,599.984858561819,
300.0075707190905],[2.7,599.9916812938634,300.0041593530683],[
2.85,599.9954297028486,300.0022851485757],[3.0,599.9974890787451
,300.0012554606275]]
```

- La solución es una lista triple [n,x[n],y[n]]
- para pintar separadamente [n,x[n]] y [n,y[n]], hay que definir estas listas dobles
- esto se puede hacer con "makelist" extrayendo las coordenadas adecuadas

```
(%i2) xsol1:makelist([sol1[i][1],sol1[i][2]],i,1,length(sol1))$
      ysol1:makelist([sol1[i][1],sol1[i][3]],i,1,length(sol1))$

(%i4) wxplot2d([[discrete,xsol1], [discrete, ysol1]],[legend, "x", "y"]);
```

(%t4)



(%o4)

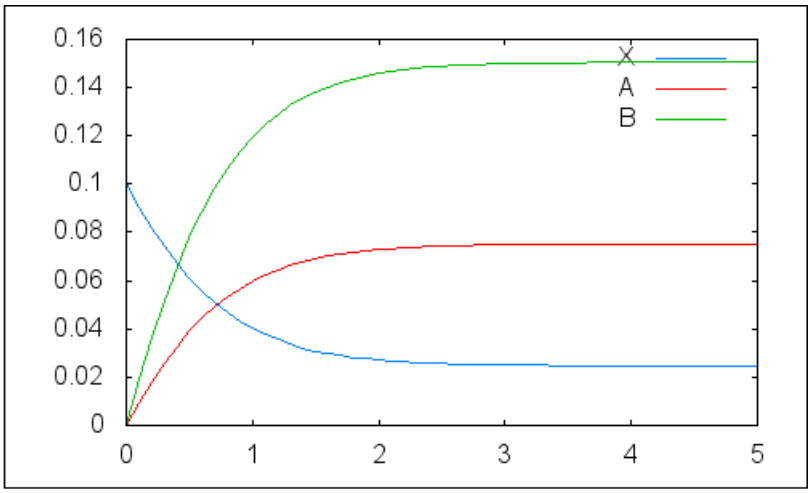
EJEMPLO 3: SISTEMAS NO LINEALES CON RK

Ejercicio 28 de la hoja 1 (petrucci) con dato inicial A(0)=B(0)=0, X(0)=0.1

- Plantea un sistema (no-lineal) de 3 ecuaciones con 3 incógnitas para los moles de A,B,X.
- Calcula la solucion para t en [0,5] con paso h=0.1
- Determina cuándo se alcanza el 90% del equilibrio

```
(%i5) kill(all);

 (%o0) done

(%i1) sol:rk ([14.5*A*B^2-X,-14.5*A*B^2+X,-2*14.5*A*B^2+2*X], [X,A,B], [0.1,0,0], [t,0,5,0.1])
      $

(%i2) length(sol);

 (%o2) 51

(%i3) Xsol:makelist([sol[i][1],sol[i][2]],i,1,length(sol))$
      Asol:makelist([sol[i][1],sol[i][3]],i,1,length(sol))$
      Bsol:makelist([sol[i][1],sol[i][4]],i,1,length(sol))$

(%i6) wxplot2d([[discrete,Xsol], [discrete, Asol],[discrete,Bsol]],[legend, "X", "A","B"]);
```

(%t6)



(%o6)

Created with wxMaxima.