1 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

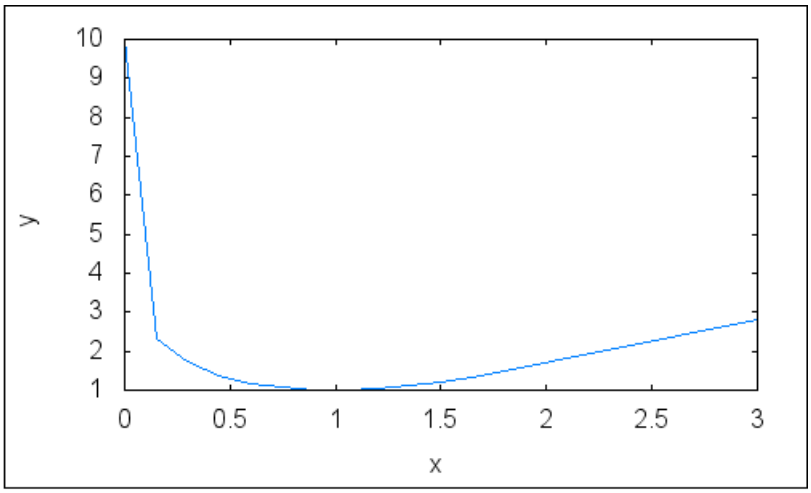
```
--> kill(all);ratprint:false;

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

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

```
--> 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]]
```

```
--> wxplot2d([discrete,sol1]);
```

(%t3)



(%o3)

veamos como funciona mejor h=0.01

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

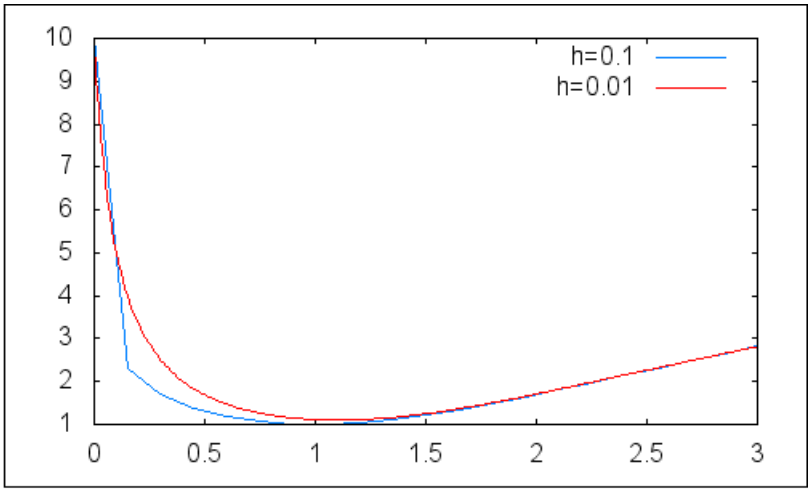
IMPORTANTE:
- al final de línea poned "$" (en lugar de ";") y no sacará en pantalla la lista de datos
  (que pueden ser miles)
- podeis calcular la longitud de la lista con "length"

--> length(sol2);

(%o5) 301

--> 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

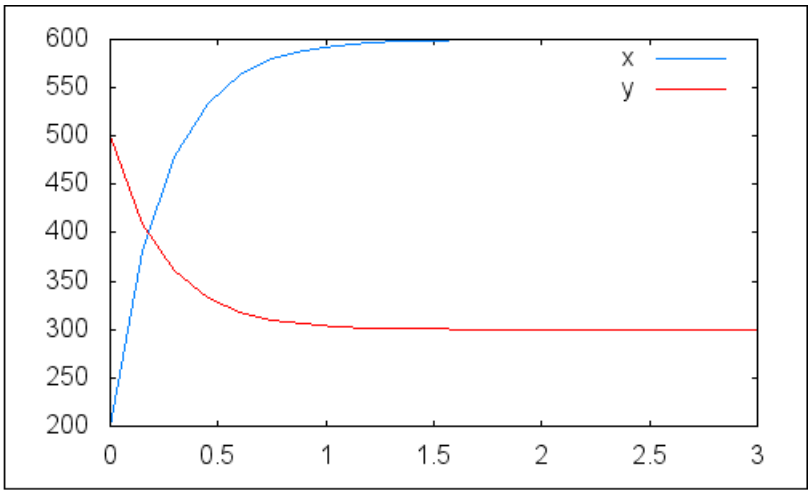
EJEMPLO 2: SISTEMAS de ED con RK
Resolver
x' = -2x + 4y
y' = x - 2y
con dato inicial x(0)=200, y(0)=500

```
--> kill(all);

 (%o0) done

--> 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

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

--> wxplot2d([[discrete,xsol1], [discrete, ysol1]],[legend, "x", "y"]);
```

(%t4)



(%o4)

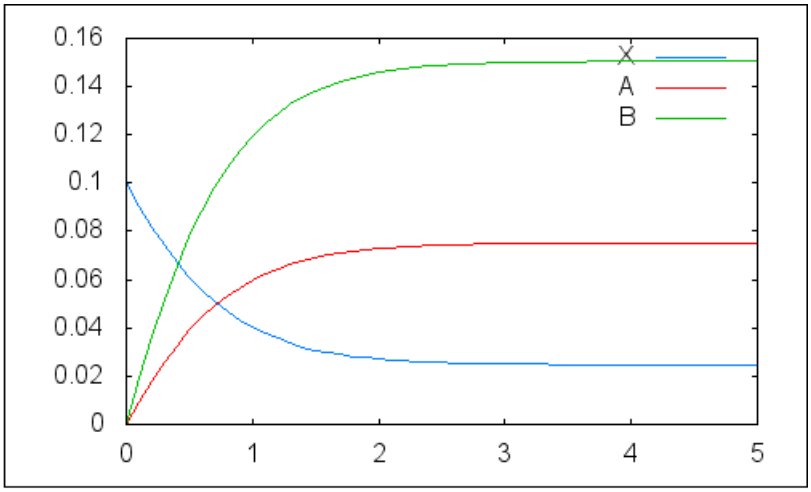
EJEMPLO 3: SISTEMAS NO LINEALES CON RK
Ejercicio 28 de la hoja 1 (petrucci) con dato inicial A(0)=B(0)=0, X(0)=0.1
- Plantea un sistema (no-lineal) de 3 ecuaciones con 3 incógnitas para los moles de A,B,X.
- Calcula la solucion para t en [0,5] con paso h=0.1
- Determina cuándo se alcanza el 90% del equilibrio

```
--> kill(all);
```

```
(%o0)  done
```

```
--> 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])
    $

--> length(sol);

(%o2)  51

--> 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))$

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

(%t6)



(%o6)

2 Gráficas con plotdf

- Plotdf es otro comando sencillo para visualizar soluciones de sistemas 2x2,
lineales o no-lineales (y que ya usamos en algunos ejercicios de la hoja 1).
- la principal ventaja es el uso de "sliders" para visualizar el efecto de los parámetros

OBSERVACIONES:
- Plotdf dibuja DIAGRAMAS XY (llamados "diagramas de fases")
- Para dibujar las curvas x(t), y(t) se usa la opción "versus_t" (1=true, 0=false)
- Pinchando en el plano XY se dibuja la trayectoria que seguiria la solucion
con ese valor inicial.
- [direction,forward] dibuja solo la trayectoria con t>0

EJEMPLO: Ejercicio 16c
- Dibujar la solucion de la ED de la reaccion quimica "brusselator" con a=1, b=2.5, y x(0)=y(0)=1
- Dibujar las soluciones variando el parametro b entre 1 y 4

```
--> plotdf([1-3.5*x+x^2*y,2.5*x-x^2*y],[x,0,5],[y,0,5],[trajectory_at,1,1],
    [direction,forward],[nsteps,1000],[versus_t,1]);

(%o7)  0
```

Notas:
- observar en el diagrama de fases que distintos datos iniciales tienden a largo plazo a la misma solucion periodica
- los periodos se repiten aprox cada 6.5 seg
- X oscila entre 0.5-2.5, Y entre 1-3.6

repetimos el dibujo con sliders...

```
--> plotdf([a-(b+1)*x+x^2*y,b*x-x^2*y],[x,0,5],[y,0,8],[trajectory_at,1,1],[tstep,0.01],
    [direction,forward],[nsteps,1500],[parameters,"a=1,b=2.5"],[sliders,"b=1:4"],[versus_t,1]);

(%o8)  0
```

Notas:
- cuando b aumenta, las cantidades de producto aumentan y los periodos tambien
- si b=4 tenemos periodos de 9 seg y oscilaciones de X entre 0-6.5 , y de Y entre 0-7
- cuando b<2 la solucion de equilibrio deja de ser periodica y pasa a ser cte, independientemente del dato inicial

Figura 1: C:\verso\curso_UM\biotec2014\practicas\Pr6fig1.jpg

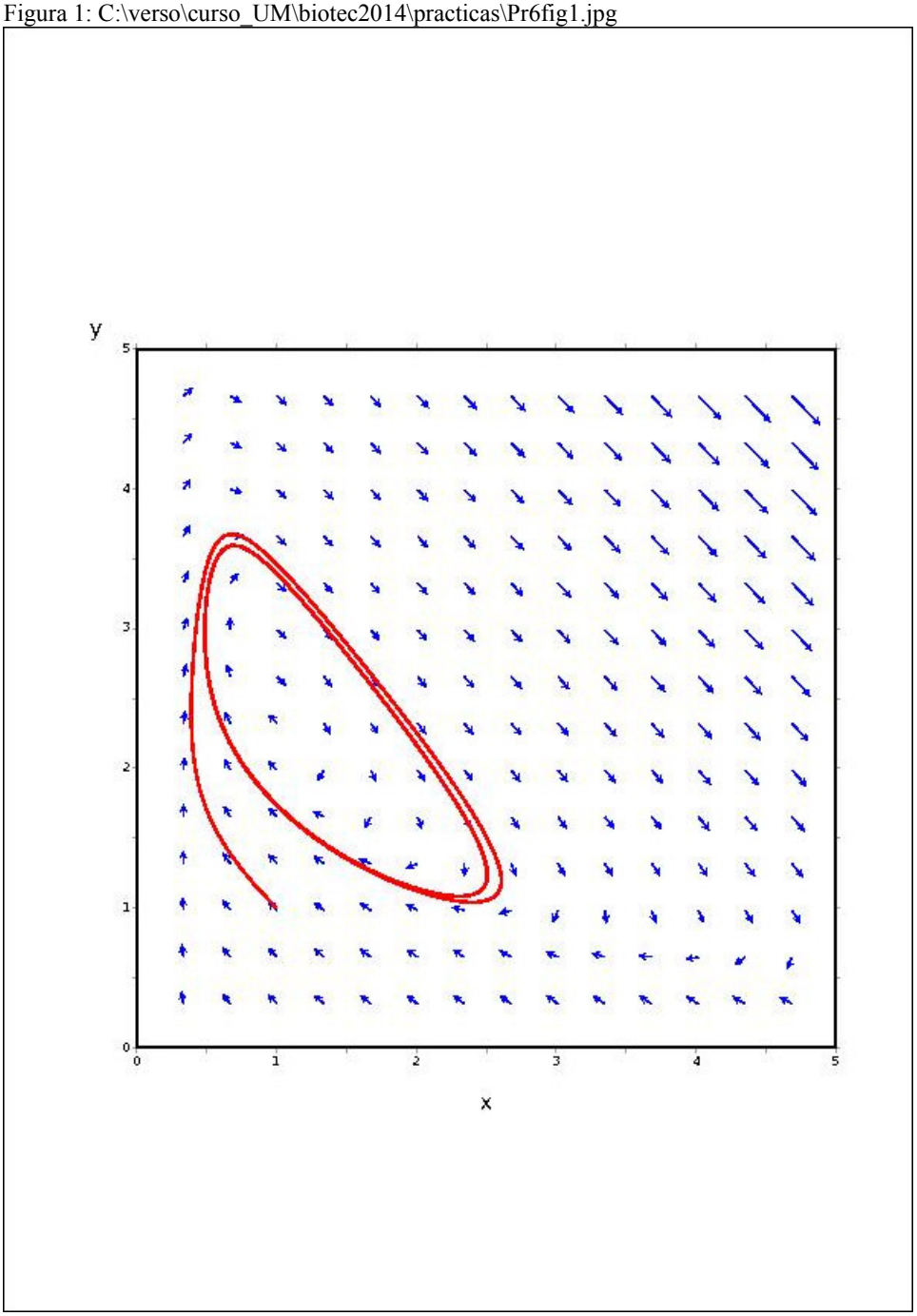


Figura 2: C:\verso\curso_UM\biotec2014\practicas\Pr6fig2.jpg

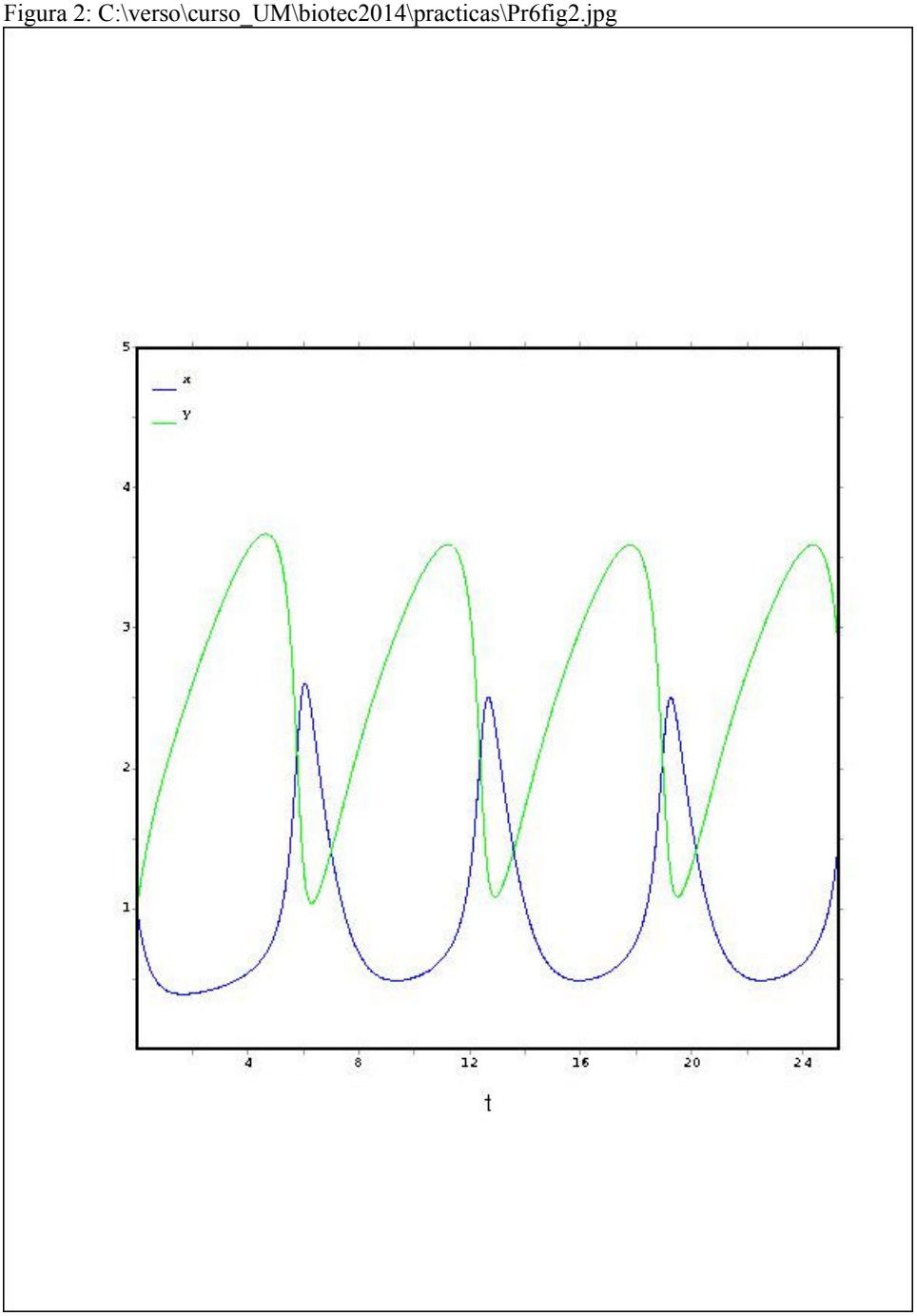


Figura 3: C:\verso\curso_UM\biotec2014\practicas\Pr6fig3.jpg

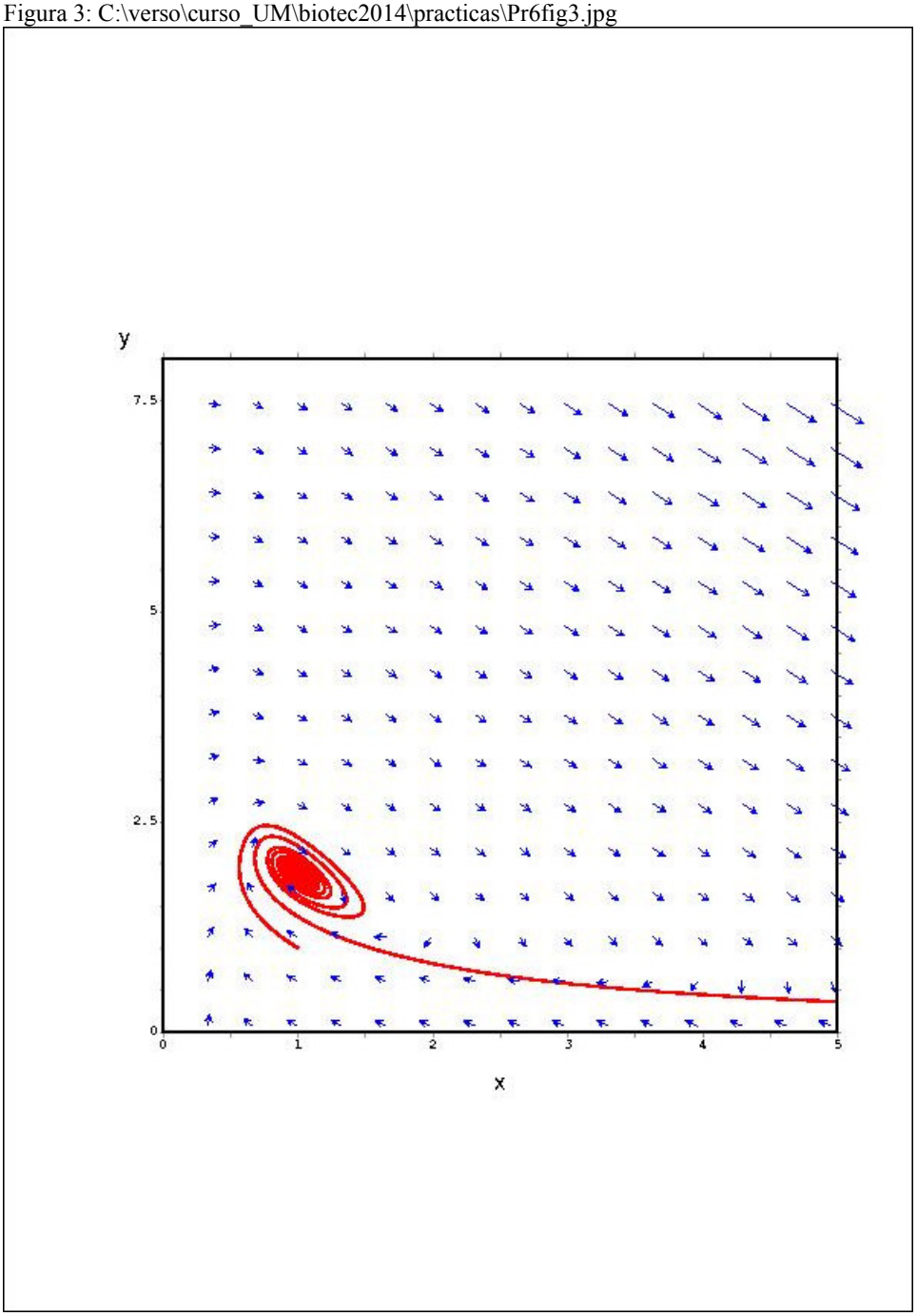
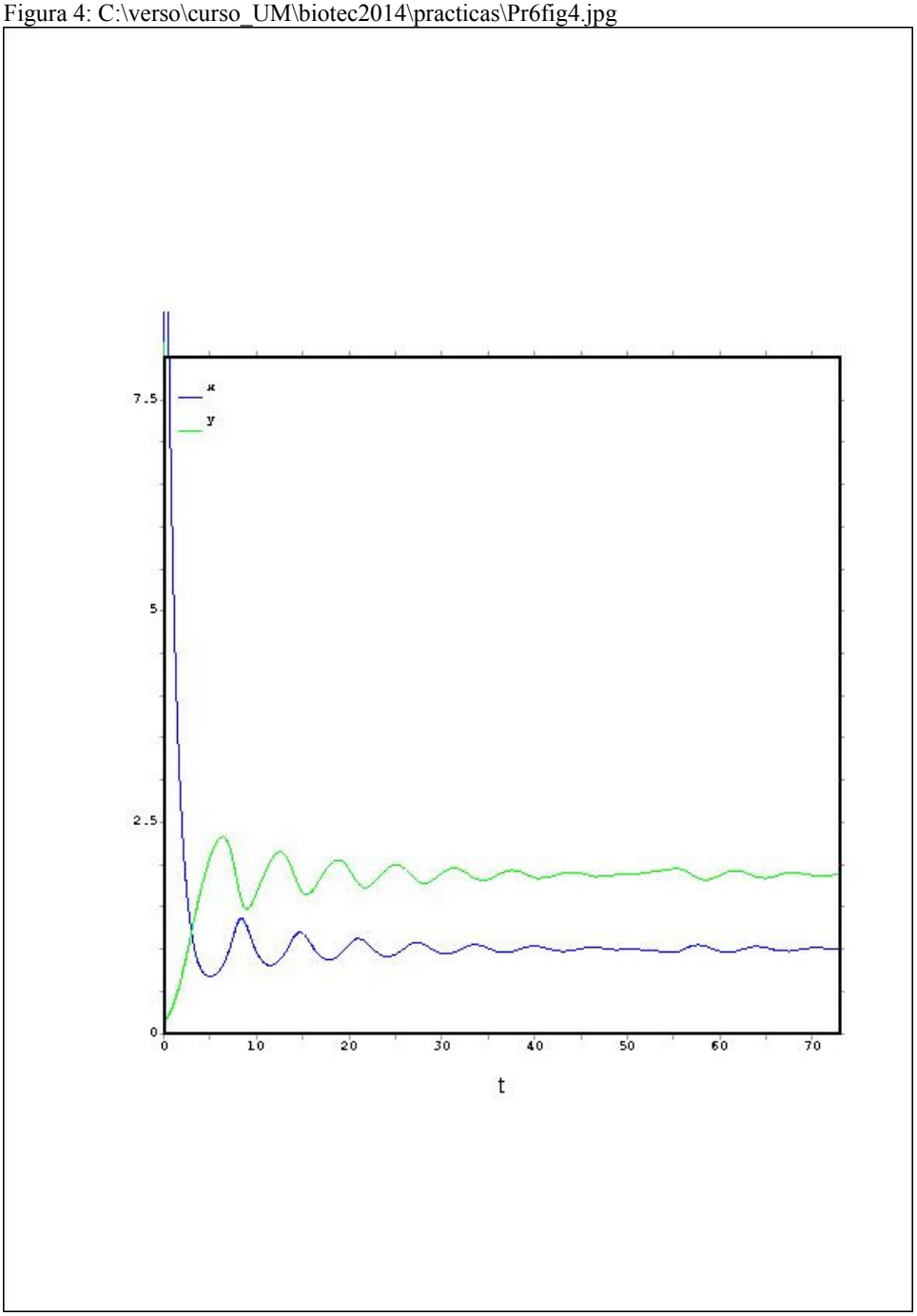


Figura 4: C:\verso\curso_UM\biotec2014\practicas\Pr6fig4.jpg



Created with wxMaxima.