Modelli Matematici per la Biologia – Esercitazione 4
a.a. 2006-2007

Dott. Simone Zuccher

18 Maggio 2007

Nota. Queste pagine potrebbero contenere degli errori: chi li trova è pregato di segnalarli all'autore (zuccher@sci.univr.it).

1 Interazione tra due popolazioni

1.1 Esercizio

Studiare, al variare dei parametri a, b, c, d (positivi), delle condizioni iniziali e del tempo finale, il comportamento del sistema

$$\begin{cases} x' &= ax - bxy \\ y' &= -cy + dxy \end{cases}$$

detto anche di Lotka-Volterra.

1.1.1 Risoluzione

Il sistema ammette due punti di equilibio, l'origine e $(c/d, a/b)$. Analizzando il sistema linearizzato attorno all'origine, si scopre che gli autovalori sono reali e $\lambda_1 = a, \lambda_2 = -c$. Pertanto, finché $a, c > 0$ l'origine è un punto di sella. La linearizzazione attorno al punto $(c/d, a/b)$ fornisce gli autovalori $\lambda_{1,2} = \pm i\sqrt{ac}$, pertanto finché $a, c > 0$ questo punto è stabile ma non asintoticamente. Infatti, partendo da qualsiasi condizione iniziale, il sistema evolve lungo traiettorie chiuse periodiche.

Utilizzando il file `myLV.m` per GNU Octave di seguito riportato si possono fare diverse prove al variare dei parametri a, b, c, d, della condizione iniziale e del tempo finale di integrazione. **Si ricordi di cambiare gli estremi della finestra di visualizzazione dipendentemente dalla condizione iniziale**.

```
% Name:     myLV.m
% Author:   Simone Zuccher
% Created:  17 May 2007
% Purpose:  solve the Lotka-Volterra system
```

```
%           u'= au - buv
%           v'= - cv + duv
%           given u0 and v0
% Input:    see file
% Output:   1. plot of u(t) versus v(t) together with the vector field
%           2. plot time histories of u(t), v(t)
% Modified:
%
% The equilibrium points are the following, but we consider only u0 and v0
% non-negative
%
% [u = 0, v = 0],
%
%       c        a
% [u = ---, v = ---],
%       d        b
%

% Clear all variables
clear all;

% Window ranges
xmin=0;
xmax=300;
ymin=0;
ymax=300;

% Model constant
global aa;

% Give instructions
disp('');
disp('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~');
disp('This script solves the classic Lotka-Volterra system:');
disp('  u''= au - buv');
disp('  v''= - cv + duv');
disp('given a, b, c, d all positive');
% Set initial conditions
aa=input('Insert constants [a b c d]: ');

% Equilibrium points
eq = [0 0];
eq = [eq; aa(3)/aa(4) aa(1)/aa(2)];
disp('Equilibrium points:');
disp(eq);


% Set initial conditions
```

```
x0=input('Insert initial conditions [x0 y0]: ');


% Set final time for integration
tmax=input('Insert final time: ');

disp('Initial condition:');
disp(x0);

% Time parameters
tmin=0;
dt=.01;
% Create time
t = tmin:dt:tmax;

% dx and dy used only for vectors
dx=abs(xmax-xmin)/30;
dy=abs(ymax-ymin)/30;
% rescales vector size
scale=0.027*max(abs(xmax-xmin),abs(ymax-ymin));

% Definition of the dynamical system
function xdot=dsys(x, t)
  global aa;
  u = x(1);
  v = x(2);
  xdot(1) = aa(1)*u - aa(2)*u*v;
  xdot(2) = -aa(3)*v + aa(4)*u*v;
endfunction

__gnuplot_set__ nokey

setax=[xmin xmax ymin ymax];
axis(setax)

[X, Y] = meshgrid(xmin:dx:xmax, ymin:dy:ymax);
DX = aa(1)*X - aa(2)*X.*Y;
DY = -aa(3)*Y + aa(4)*X.*Y;
L = sqrt(DX.^2 + DY.^2);
mytitle=["Phase portrait. Initial conditions: x0=" num2str(x0(1)) \
         ", y0=" num2str(x0(2))];
__gnuplot_set__ nokey
__gnuplot_set__ xlabel 'x(t)'
__gnuplot_set__ ylabel 'y(t)'
title(mytitle)

% Plot vector field
quiver(X, Y, scale*DX./L, scale*DY./L)
```

```
hold on;

% Plot all equilibrium points
plot( eq(:,1), eq(:,2), '*k')

x =  lsode("dsys", x0, t)';
plot( x(1,1), x(2,1), '*k', x(1,:), x(2,:), '-r')
hold off;

% Wait for keypressed
disp('Please press a key to continue...');
pause();
mytitle=["Time histories. Initial conditions: x0=" num2str(x0(1)) \
         ", y0=" num2str(x0(2))];
__gnuplot_set__ auto
__gnuplot_set__ xlabel 't'
__gnuplot_set__ ylabel 'x(t), y(t)'
title(mytitle)
__gnuplot_set__ key

% Plot time histories
plot( t, x(1,:), '-r;x(t);', t, x(2,:), '-g;y(t);')
```

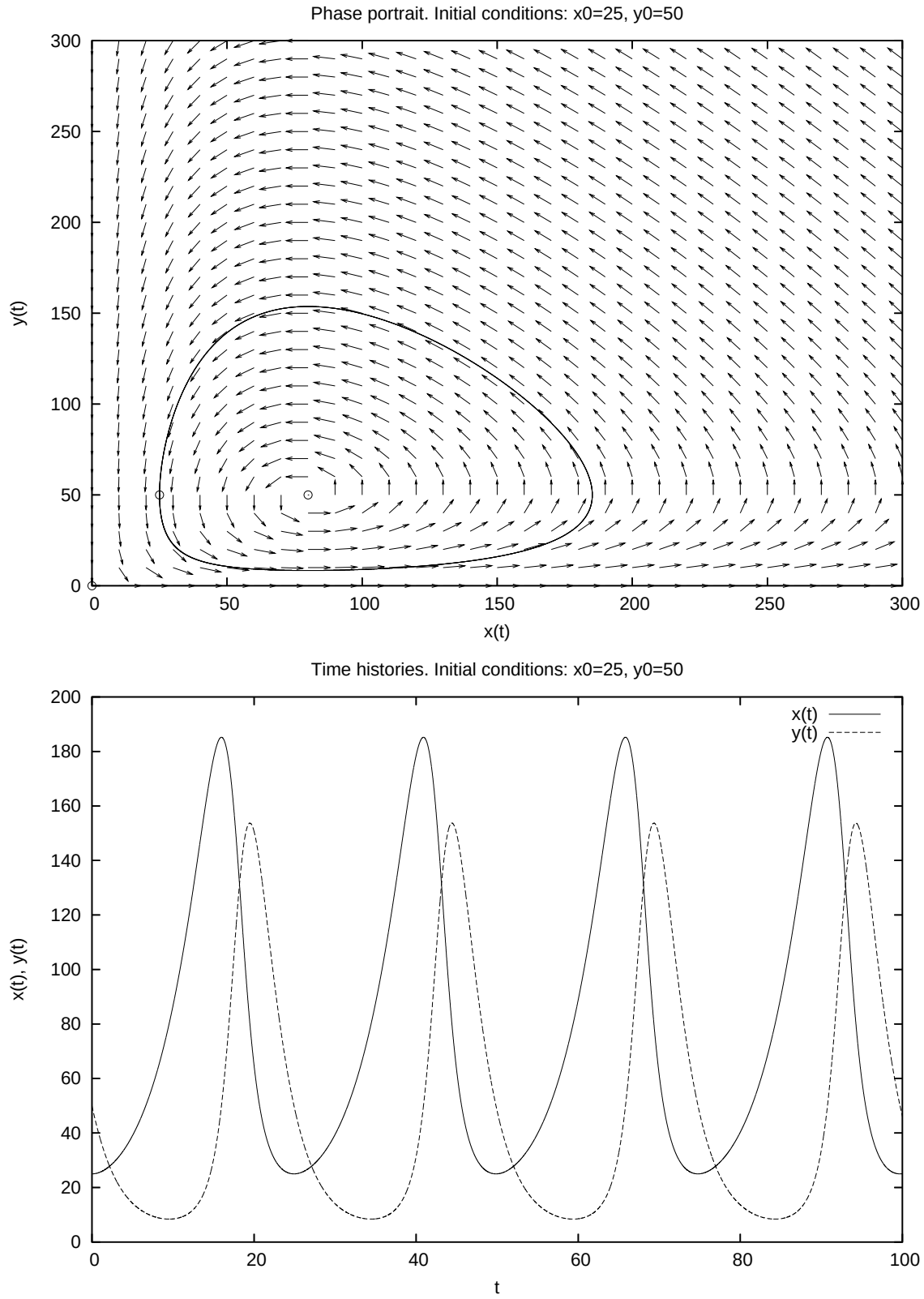

Figura 1: Ritratto di fase e storia temporale del sistema classico Lotka-Volterra per $a = 0.2, b = 0.004, c = 0.4, d = 0.005$, $(x_0, y_0) = (25, 50)$ e $t_f = 100$.

1.2 Esercizio

Studiare, al variare del parametro k (positivo), delle condizioni iniziali e del tempo finale, il comportamento del sistema

$$\begin{cases} x' &= x(1-x) - kxy^2 \\ y' &= y(1-y) - kyx^2. \end{cases} \tag{1}$$

1.2.1 Risoluzione

Il sistema ammette i seguenti punti di equilibrio: $(0,0)$, $(1,0)$, $(0,1)$, $(-\frac{\sqrt{4k+1}+1}{2k}, -\frac{\sqrt{4k+1}+1}{2k})$, $(\frac{\sqrt{4k+1}-1}{2k}, \frac{\sqrt{4k+1}-1}{2k})$, $(\frac{\sqrt{4k-3}+1}{2k}, -\frac{\sqrt{4k-3}-1}{2k})$, $(-\frac{\sqrt{4k-3}-1}{2k}, \frac{\sqrt{4k-3}+1}{2k})$. Ovviamente, trattandosi di popolazioni, consideriamo solo i punti di equilibrio nel primo quadrante, ovvero $(0,0)$, $(1,0)$, $(0,1)$ e $(\frac{\sqrt{4k+1}-1}{2k}, \frac{\sqrt{4k+1}-1}{2k})$. Anziché procedere attraverso lo studio analitico della stabilità dei punti di equilibrio, ci limitiamo allo studio numerico tramite il il file `LVmod.m` deducendo da esso le proprietà del sistema.

Consideriamo il caso $k > 1$, quindi per fissare le idee $k = 2$, e lasciamo allo studente lo studio per $0 < k \leq 1$. Come si può notare dalla figura 2, l'origine non è un punto di equilibrio stabile, mentre lo sono i punti $(1,0)$ e $(0,1)$. Il punto $(\frac{\sqrt{4k+1}-1}{2k}, \frac{\sqrt{4k+1}-1}{2k})$ è di sella. Analizzando il ritratto di fase, pertanto, si osserva che il piano $[0,1] \times [0,1]$ viene suddiviso in 4 regioni, due delle quali sono il bacino di attrazione del punto $(0,1)$ (quelle sotto la bisettrice del primo quadrante) e le altre due sono il bacino di attrazione del punto $(1,0)$ (le due sopra la bisettrice). Il fatto che questi due siano i soli punti di equilibrio del sistema ha una fortissima implicazione dal punto di vista della crescita delle popolazioni interagenti. Infatti, una è sempre destinata ad estinguersi (la minore delle due nella condizione iniziale) mentre l'altra raggiunge il massimo (la maggiore delle due nella condizione iniziale).

Partendo da una condizione iniziale sulla bisettrice, invece, la soluzione viene attratta dal punto di equilibrio $(\frac{\sqrt{4k+1}-1}{2k}, \frac{\sqrt{4k+1}-1}{2k})$, come mostrato nelle figure 3 e 4. Tuttavia, se la simulazione è estesa a tempi più lunghi, per esempio $t_f = 100$, si nota che la soluzione non rimane in $(\frac{\sqrt{4k+1}-1}{2k}, \frac{\sqrt{4k+1}-1}{2k})$ ma viene attratta da $(1,0)$ o $(0,1)$ dipendentemente dagli errori numerici (lo studente diligente lo provi).

Aumentando il valore di k, che rappresenta il coefficiente di competizione tra le due specie, il punto di equilibrio $(\frac{\sqrt{4k+1}-1}{2k}, \frac{\sqrt{4k+1}-1}{2k})$ si sposta verso l'origine (a cui tende per $k \to \infty$). Questo ha come effetto, a parità di condizione iniziale, un'aumentata velocità di convergenza verso i fuochi stabili. Confrontando i risultati per $k = 20$ e riportati in figura 5 con i risultati ottenuti per $k = 2$ (vedi figura 2), si osserva infatti che la specie che tende a zero ci tende molto più velocemente nel caso di k elevato.

Per avere un'idea di quanto velocemente una delle due popolazioni si estingue, basta plottare il prodotto xy.

```
% Name:     LVmod.m
% Author:   Simone Zuccher
% Created:  17 May 2007
% Purpose:  solve the modified Lotka-Volterra system
%           u'=u(1-u)-kuv^2
```

```
%           v'=v(1-v)-kvu^2
%           given u0 and v0, and k>3/4
% Input:    see file
% Output:   1. plot of u(t) versus v(t) together with the vector field
%           2. plot time histories of u(t), v(t)
% Modified:
%
% The equilibrium points are the following, but we consider only u0 and v0
% non-negative
%
% [u = 0, v = 0],
%
% [u = 1, v = 0],
%
% [u = 0, v = 1],
%
%        sqrt(4 k + 1) + 1        sqrt(4 k + 1) + 1
% [u = - -----------------, v = - -----------------],
%              2 k                      2 k
%
%      sqrt(4 k + 1) - 1      sqrt(4 k + 1) - 1
% [u = -----------------, v = -----------------],
%             2 k                    2 k
%
%      sqrt(4 k - 3) + 1        sqrt(4 k - 3) - 1
% [u = -----------------, v = - -----------------],
%             2 k                      2 k
%
%        sqrt(4 k - 3) - 1      sqrt(4 k - 3) + 1
% [u = - -----------------, v = -----------------]
%               2 k                    2 k
%

% Clear all variables
clear all;

% Window ranges
xmin=0;
xmax=1;
ymin=0;
ymax=1;


% Model constant
global k;
% Set model constant
k=input('Insert k: ');
```

```
% Equilibrium points
eq = [0 0];
eq = [eq; 0 1];
eq = [eq; 1 0];
eq= [eq; -(sqrt(4 * k + 1) + 1)/2/k -(sqrt(4 * k + 1) + 1)/2/k];
eq= [eq; (sqrt(4 * k + 1) - 1)/2/k (sqrt(4 * k + 1) - 1)/2/k];
eq= [eq; (sqrt(4 * k - 3) + 1)/2/k -(sqrt(4 * k - 3) - 1)/2/k];
eq= [eq; -(sqrt(4 * k - 3) - 1)/2/k (sqrt(4 * k - 3) + 1)/2/k];
disp('Equilibrium points:');
disp(eq);


% Set initial conditions
x0=input('Insert initial conditions [x0 y0]: ');


% Set final time for integration
tmax=input('Insert final time: ');

disp('Initial condition:');
disp(x0);

% Time parameters
tmin=0;
dt=.01;
% Create time
t = tmin:dt:tmax;

% dx and dy used only for vectors
dx=abs(xmax-xmin)/30;
dy=abs(ymax-ymin)/30;
% rescales vector size
scale=0.027*max(abs(xmax-xmin),abs(ymax-ymin));



% Definition of the dynamical system
function xdot=dsys(x, t)
  global k;
  u = x(1);
  v = x(2);
  xdot(1) = u*(1-u)-k*u*v^2;
  xdot(2) = v*(1-v)-k*v*u^2;
endfunction

__gnuplot_set__ nokey

setax=[xmin xmax ymin ymax];
```

```
axis(setax)

[X, Y] = meshgrid(xmin:dx:xmax, ymin:dy:ymax);

DX = X.*(1-X)-k*X.*(Y.^2) ;
DY = Y.*(1-Y)-k*Y.*(X.^2);
L = sqrt(DX.^2 + DY.^2);
mytitle=["Phase portrait. Initial conditions: x0=" num2str(x0(1)) \
         ", y0=" num2str(x0(2))];
__gnuplot_set__ nokey
__gnuplot_set__ xlabel 'x(t)'
__gnuplot_set__ ylabel 'y(t)'
title(mytitle)

% Plot vector field
quiver(X, Y, scale*DX./L, scale*DY./L)
hold on;

% Plot all equilibrium points
plot( eq(:,1), eq(:,2), '*k')

x =  lsode("dsys", x0, t)';
plot( x(1,1), x(2,1), '*k', x(1,:), x(2,:), '-r')
hold off;

% Wait for keypressed
disp('Please press a key to continue...');
pause();
mytitle=["Time histories. Initial conditions: x0=" num2str(x0(1)) \
         ", y0=" num2str(x0(2))];
__gnuplot_set__ auto
__gnuplot_set__ xlabel 't'
__gnuplot_set__ ylabel 'x(t), y(t)'
title(mytitle)
__gnuplot_set__ key

% Plot time histories
plot( t, x(1,:), '-r;x(t);', t, x(2,:), '-g;y(t);')
```

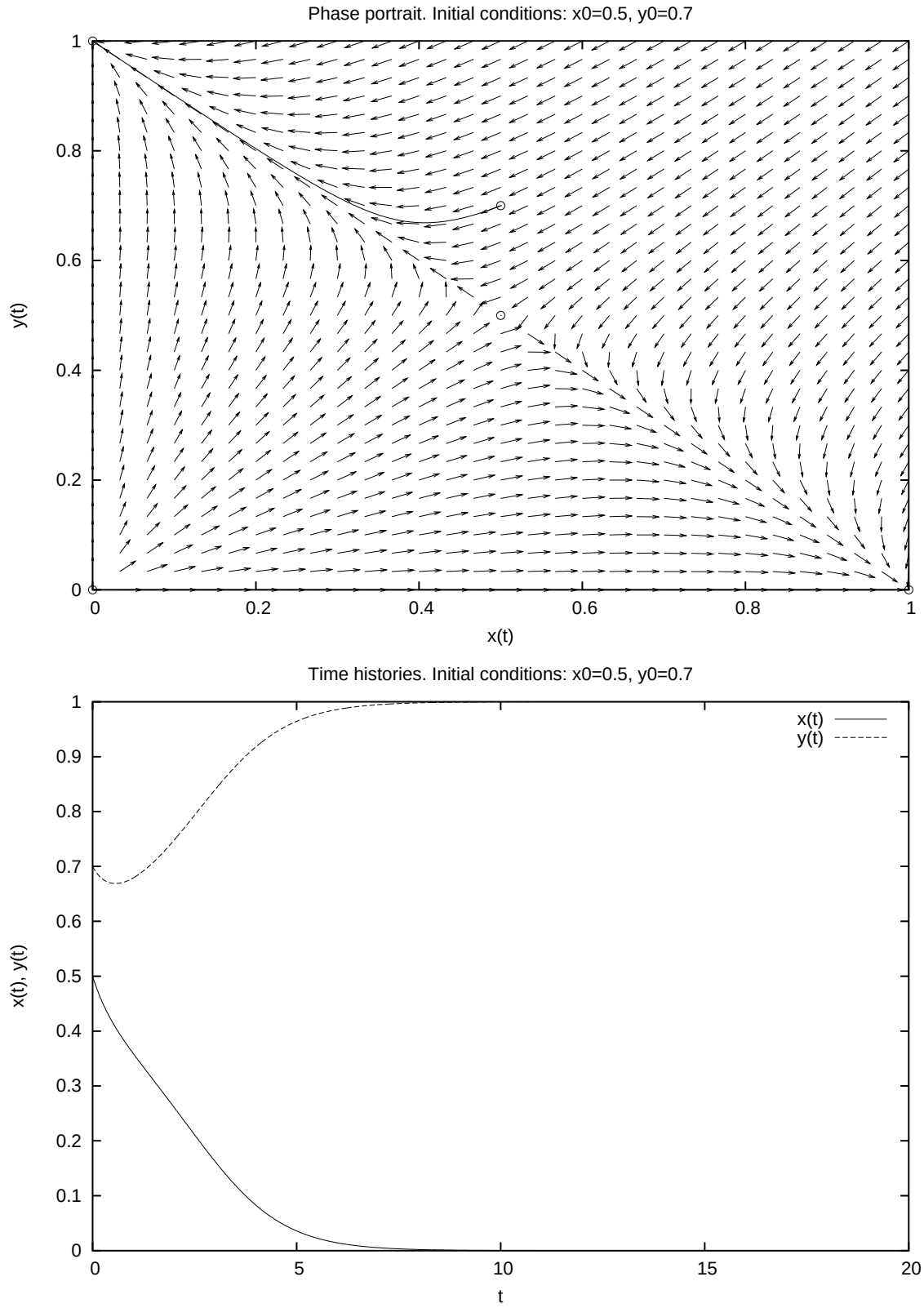

Figura 2: Ritratto di fase e storia temporale del sistema Lotka-Volterra modificato per $k = 2$, $(x_0, y_0) = (0.5, 0.7)$ e $t_f = 20$.

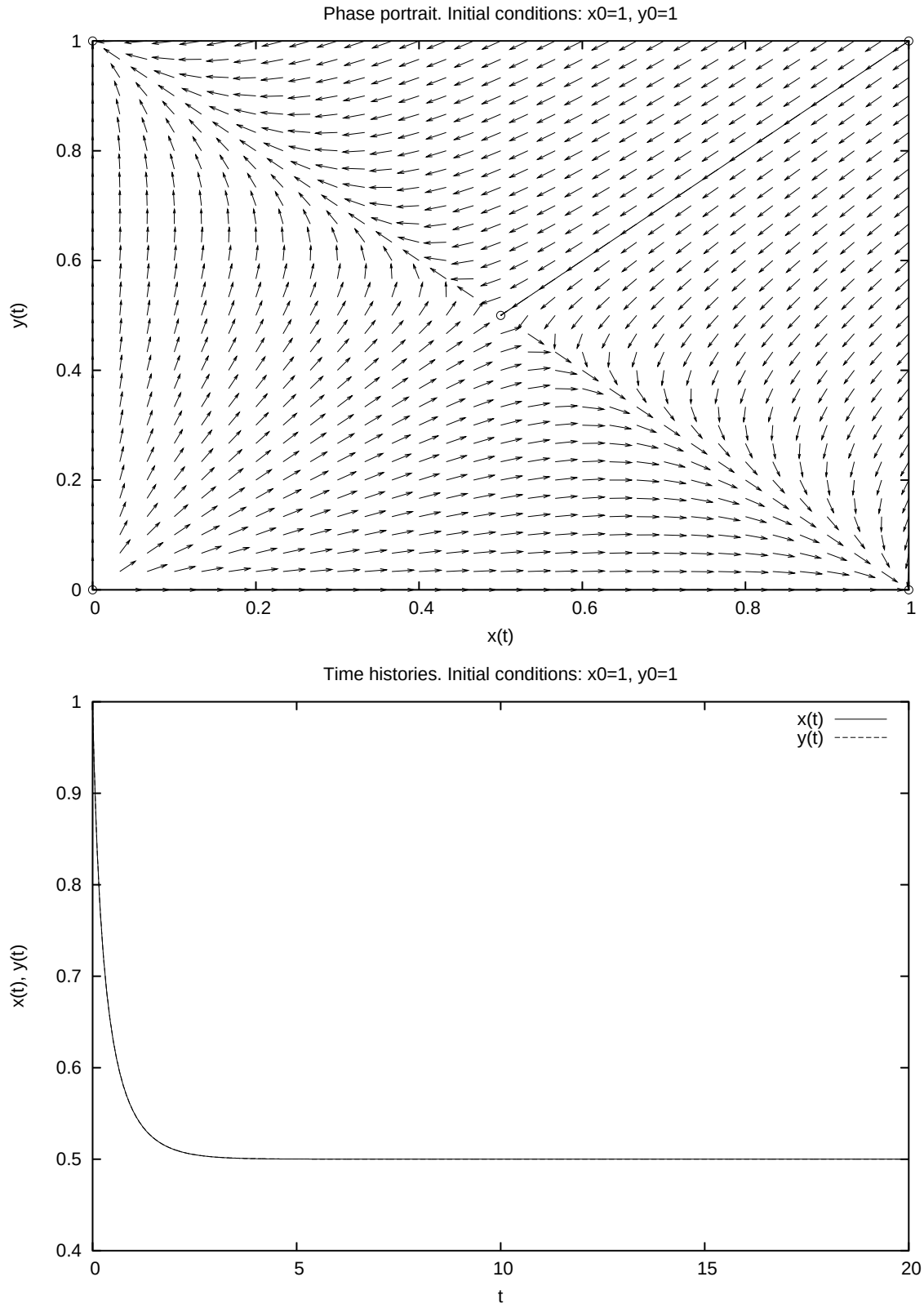

Figura 3: Ritratto di fase e storia temporale del sistema Lotka-Volterra modificato per $k = 2$, $(x_0, y_0) = (1.0, 1.0)$ e $t_f = 20$.

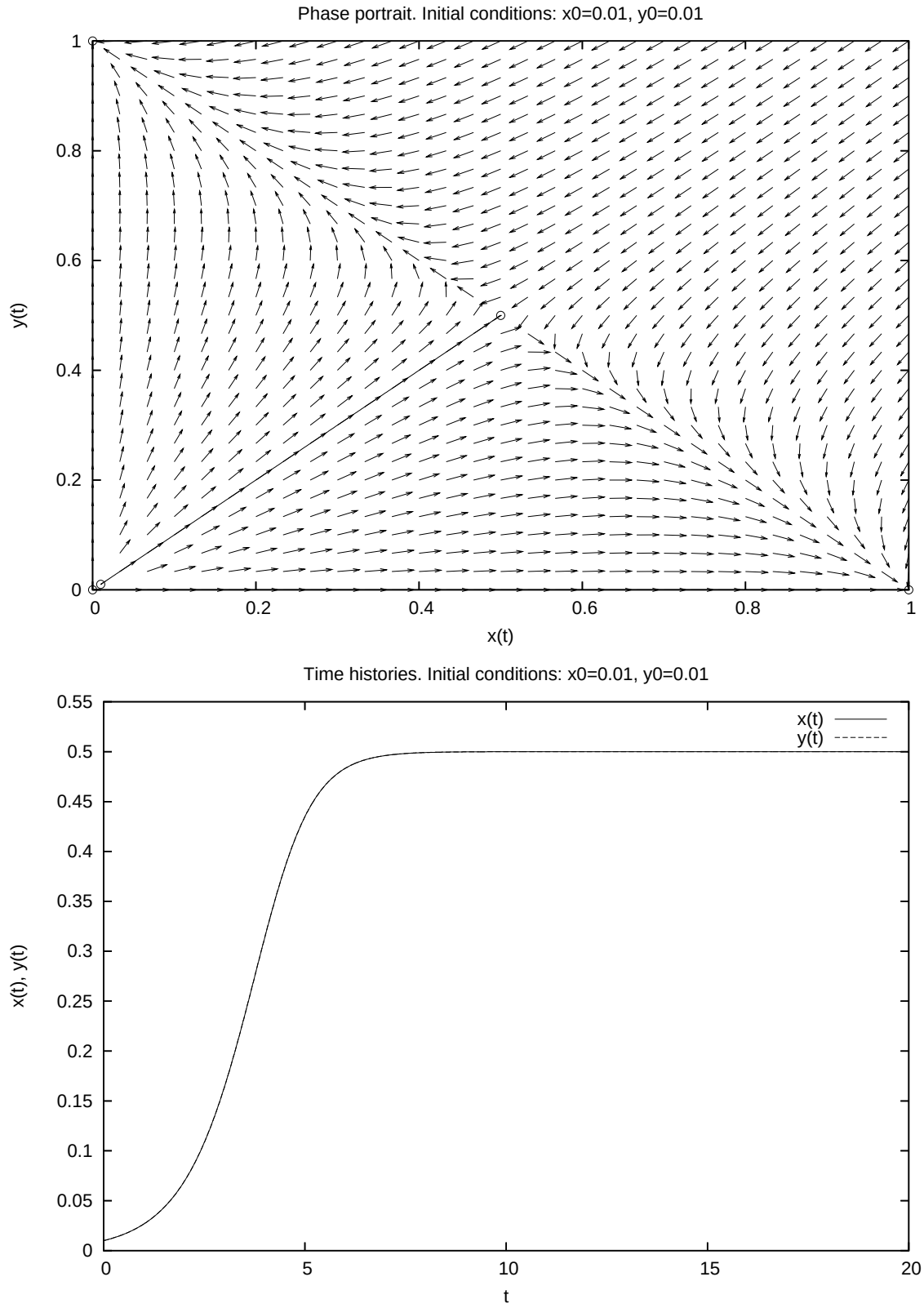

Figura 4: Ritratto di fase e storia temporale del sistema Lotka-Volterra modificato per $k = 2$, $(x_0, y_0) = (0.01, 0.01)$ e $t_f = 20$.

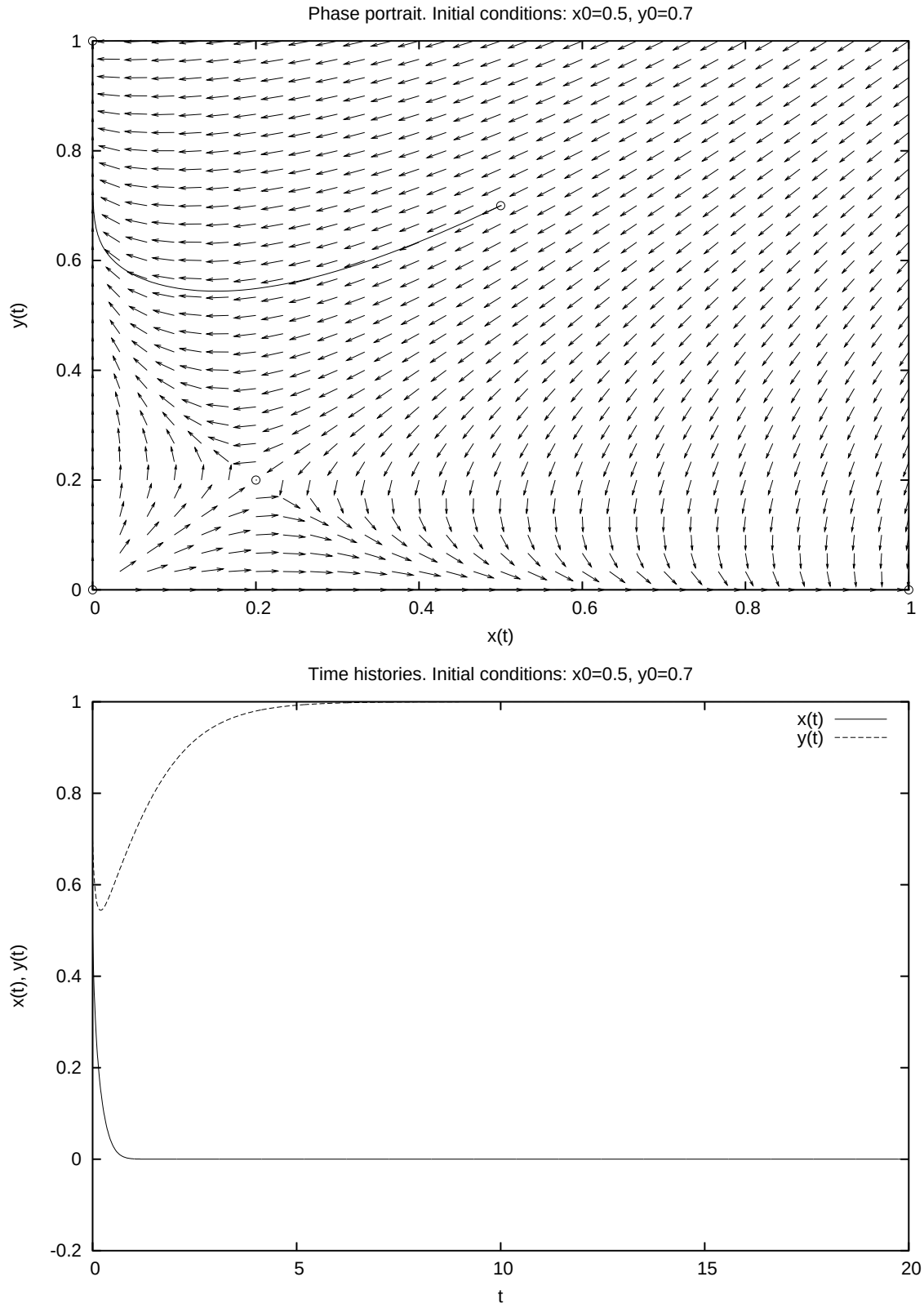

Figura 5: Ritratto di fase e storia temporale del sistema Lotka-Volterra modificato per $k = 20$, $(x_0, y_0) = (0.5, 0.7)$ e $t_f = 20$.

1.3 Esercizio

Studiare, al variare dei parametri $a, b \in \mathbb{R}$ (positivi e/o negativi), delle condizioni iniziali e del tempo finale, il comportamento del sistema

$$\begin{cases} x' &= x(1-x) - axy \\ y' &= y(1-y) - byx. \end{cases} \tag{2}$$

1.3.1 Risoluzione

Il sistema dato può essere riscritto come

$$\begin{cases} x' &= x - x^2 - axy \\ y' &= y - y^2 - byx, \end{cases}$$

e pertanto è un caso particolare del sistema generale

$$\begin{cases} x' &= \alpha_1 x + a_{11}x^2 + a_{12}xy \\ y' &= \alpha_2 y + a_{21}xy + a_{22}y^2 \end{cases} \tag{3}$$

con $\alpha_1 = \alpha_2 = 1$, $a_{11} = a_{22} = -1$ e $a_{12} = -a$, $a_{21} = -b$. Pertanto le due popolazioni hanno una legge di crescita molto simile, con un tasso di crescita positivo e pari a 1 (ossia entrambe possono esistere anche da sole), e un tasso di competizione *intraspecifica* pari a 1 ($-1 < 0$ significa che c'è competizione all'interno della stessa specie). Se a e b sono diversi da zero c'è anche un'interazione tra le due specie, detta *interspecifica*, che può essere di competizione (a, b entrambi positivi) o cooperazione (a, b entrambi negativi). Per lo studio numerico del sistema generale (3) utilizziamo lo scipt `compcoop.m`, di seguito riportato.

I punti di equilibrio sono $(0,0)$, $(1,0)$, $(0,1)$ e $(\frac{a-1}{a\,b-1}, \frac{b-1}{a\,b-1})$.

Esaminiamo dapprima il caso $a = b = 0$, ovvero assenza di competizione/cooperazione. I punti di equilibrio sono i 4 vertici del quadrato di lato 1, di cui solo il punto $(1,1)$ è un pozzo (quindi stabile), mentre gli altri tre punti sono o sorgente (e quindi instabile, l'origine) o di sella ($(1,0)$ e $(0,1)$) e quindi comunque instabili. Partendo da condizioni iniziali positive per entrambe le specie, pertanto, il sistema evolverà verso il punto di equilibrio $(1,1)$. Questi risultati sono riassunti visivamente in figura 6.

Caso $0 < a = b < 1$. Il sistema è competitivo con uguale competizione per entrambe le specie. Il punto $(\frac{a-1}{a\,b-1}, \frac{b-1}{a\,b-1})$ è l'unico stabile ed attrae le soluzioni con condizione iniziale positiva per entrambe le specie. Questo è chiaramente visibile in figura 7.

Caso $a = b = 1$. Il punto $(\frac{a-1}{a\,b-1}, \frac{b-1}{a\,b-1})$ non esiste e gli altri tre sono comunque instabili. La soluzione è attratta dai punti di intersezione tra la retta $y = -x+1$ e la retta $y = \frac{y_0}{x_0}x$, dove (x_0, y_0) è la condizione iniziale. Questa situazione è visibile in figura 8, ottenuta per $a = b = 1$.

Caso $a = b > 1$. Il punto $(\frac{a-1}{a\,b-1}, \frac{b-1}{a\,b-1})$ diventa di sella, e quindi instabile. Al contrario, i punti $(1,0)$ e $(0,1)$ diventano dei fuochi stabili, come visibile in figura 9 per $a = b = 1.5$. Pertanto la competizione è tale da far sopravvivere solo la specie che parte avvantaggiata (si vedano le zone del bacino di attrazione dei fuochi stabili). L'origine rimane una sorgente (instabile). È chiaro che, partendo da condizioni iniziali $0 < x_0 = y_0 < 1$ la soluzione finisce nel punto $(\frac{a-1}{a\,b-1}, \frac{b-1}{a\,b-1})$, che è instabile essendo di sella

Caso $a = b >> 1$. Il punto $(\frac{a-1}{a\,b-1}, \frac{b-1}{a\,b-1})$ tende all'origine e, come visto nell'esempio precedente, una delle due popolazioni si estingue molto velocemente (vedi figura 10). Plottanto il prodotto $\varphi(t) = x(t)y(t)$ si può avere un'idea del tasso di segregazione delle popolazioni.

Caso $-1 < a = b < 0$. Il sistema è cooperativo, nel senso che le due specie anziché contrastarsi vicendevolmente, cooperano alla sopravvivenza l'una dell'altra. In particolare, il punto $(\frac{a-1}{a\,b-1}, \frac{b-1}{a\,b-1})$ è l'unico stabile ed attrae tutte le traiettorie (purché non partano da punti di equilibrio), come si vede in figura 11.

Caso $a = b-1$. Il punto $(\frac{a-1}{a\,b-1}, \frac{b-1}{a\,b-1})$ non esiste e gli altri punti di equilibrio sono instabili. Pertanto, entrambe le soluzioni crescono indefinitamente (il sistema è cooperativo), come chiaramente provato dalla figura 12.

Caso $a = b < -1$. Il punto $(\frac{a-1}{a\,b-1}, \frac{b-1}{a\,b-1})$ si trova nel terzo quadrante e quindi non viene considerato. Gli altri tre sono instabili, per cui entrambe le soluzioni crescono indefinitamente in tempi molto brevi (vedi figura 13).

Nota. Alla luce dei risultati ottenuti in questo esercizio, dovrebbe essere più immediato interpretare quanto trovato nell'esercizio precedente.

```
% Name:     compcoop.m
% Author:   Simone Zuccher
% Created:  17 May 2007
% Purpose:  solve the general system
%           u'= au + buu + cuv
%           v'= dv + evv + fuv
%           given u0 and v0
% Input:    see file
% Output:   1. plot of u(t) versus v(t) together with the vector field
%           2. plot time histories of u(t) and  v(t)
% Modified:
%
% The equilibrium points are the following, but we consider only u0 and v0
% non-negative
%
%
% [u = 0, v = 0],
%
%       a
% [u = - -, v = 0],
%       b
%                d
% [u = 0, v = - -],
%                e
%      a e - c d        a f - b d
% [u = ---------, v = - ---------]
%      c f - b e        c f - b e
```

```
% Clear all variables
clear all;

% Window ranges
xmin=-.0;
xmax=3.;
ymin=-.0;
ymax=3.;

% Model constant
global aa;

% Give instructions
disp('');
disp('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~');
disp('This script solves the general system:');
disp('  u''= au + buu + cuv');
disp('  v''= dv + evv + fuv');
disp('given a, b, c, d, e, f.');
% Set initial conditions
aa=input('Insert constants [a b c d e f]: ');

% Equilibrium points
eq = [0 0];
if(abs(aa(2))>0)
   eq = [eq; -aa(1)/aa(2) 0];
endif
if(abs(aa(5))>0)
   eq = [eq; 0 -aa(4)/aa(5)];
endif
if(abs(aa(3)*aa(6)-aa(2)*aa(5))>0)
   eq = [eq; (aa(1)*aa(5)-aa(3)*aa(4))/(aa(3)*aa(6)-aa(2)*aa(5)) \
             -(aa(1)*aa(6)-aa(2)*aa(4))/(aa(3)*aa(6)-aa(2)*aa(5)) ];
endif
disp('Equilibrium points:');
disp(eq);


% Set initial conditions
x0=input('Insert initial conditions [x0 y0]: ');


% Set final time for integration
tmax=input('Insert final time: ');

disp('Initial condition:');
disp(x0);
```

```
% Time parameters
tmin=0;
dt=.01;
% Create time
t = tmin:dt:tmax;

% dx and dy used only for vectors
dx=abs(xmax-xmin)/30;
dy=abs(ymax-ymin)/30;
% rescales vector size
scale=0.027*max(abs(xmax-xmin),abs(ymax-ymin));

% Definition of the dynamical system
function xdot=dsys(x, t)
  global aa;
  u = x(1);
  v = x(2);
  xdot(1) = aa(1)*u + aa(2)*u*u + aa(3)*u*v;
  xdot(2) = aa(4)*v + aa(5)*v*v + aa(6)*u*v;
endfunction

__gnuplot_set__ nokey

setax=[xmin xmax ymin ymax];
axis(setax)

[X, Y] = meshgrid(xmin:dx:xmax, ymin:dy:ymax);
DX = aa(1)*X + aa(2)*X.*X + aa(3)*X.*Y;
DY = aa(4)*Y + aa(5)*Y.*Y + aa(6)*X.*Y;
L = sqrt(DX.^2 + DY.^2);
mytitle=["Phase portrait. Initial conditions: x0=" num2str(x0(1)) \
         ", y0=" num2str(x0(2))];
__gnuplot_set__ nokey
__gnuplot_set__ xlabel 'x(t)'
__gnuplot_set__ ylabel 'y(t)'
title(mytitle)

% Plot vector field
quiver(X, Y, scale*DX./L, scale*DY./L)
hold on;

% Plot all equilibrium points
plot( eq(:,1), eq(:,2), '*k')

x =  lsode("dsys", x0, t)';
plot( x(1,1), x(2,1), '*k', x(1,:), x(2,:), '-r')
hold off;
```

```
% Wait for keypressed
disp('Please press a key to continue...');
pause();
mytitle=["Time histories. Initial conditions: x0=" num2str(x0(1)) \
         ", y0=" num2str(x0(2))];
__gnuplot_set__ auto
__gnuplot_set__ xlabel 't'
__gnuplot_set__ ylabel 'x(t), y(t)'
title(mytitle)
__gnuplot_set__ key

% Plot time histories
plot( t, x(1,:), '-r;x(t);', t, x(2,:), '-g;y(t);')
```

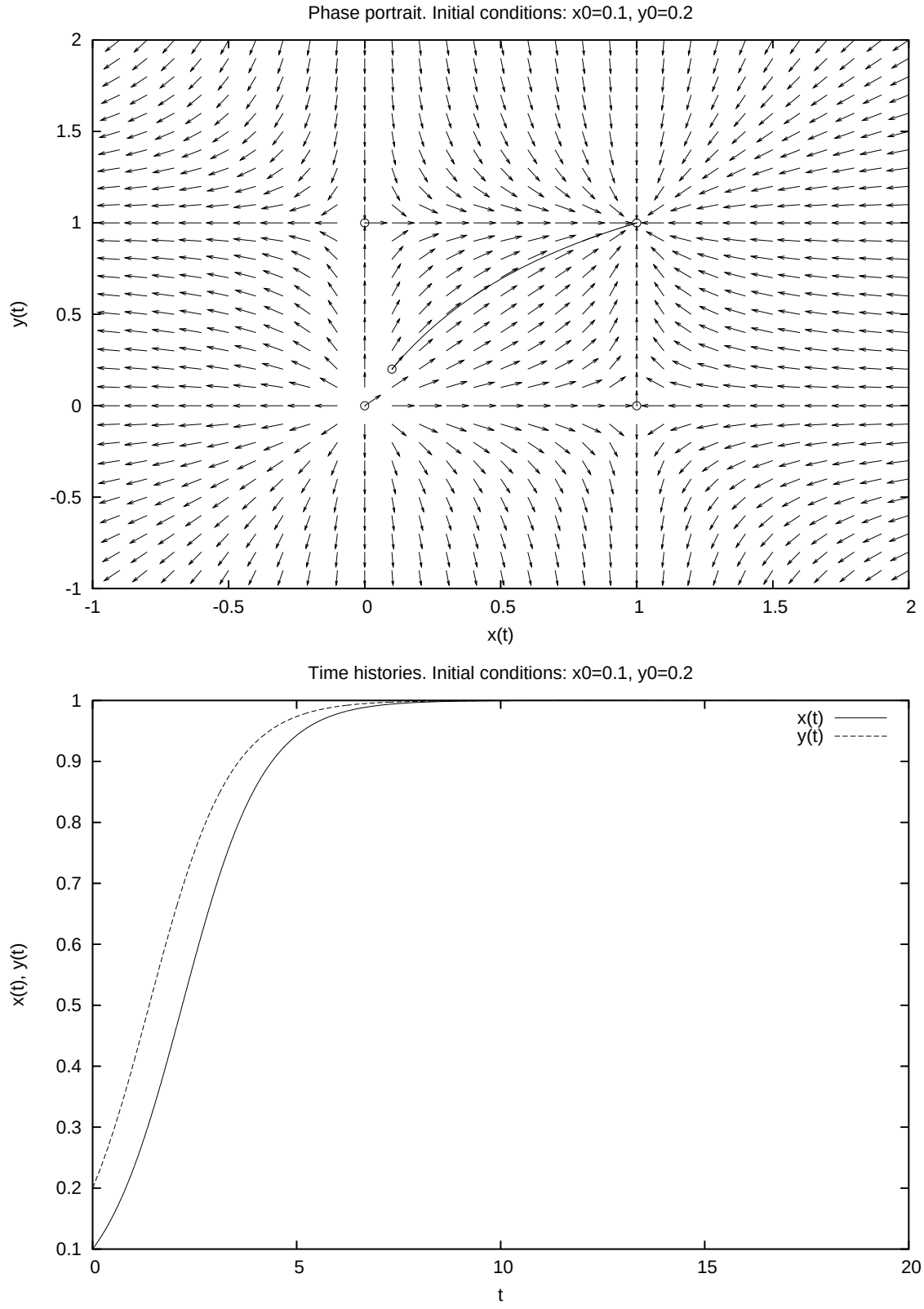

Figura 6: Ritratto di fase e storia temporale per $a = b = 0$, $(x_0, y_0) = (0.1, 0.2)$ e $t_f = 20$.

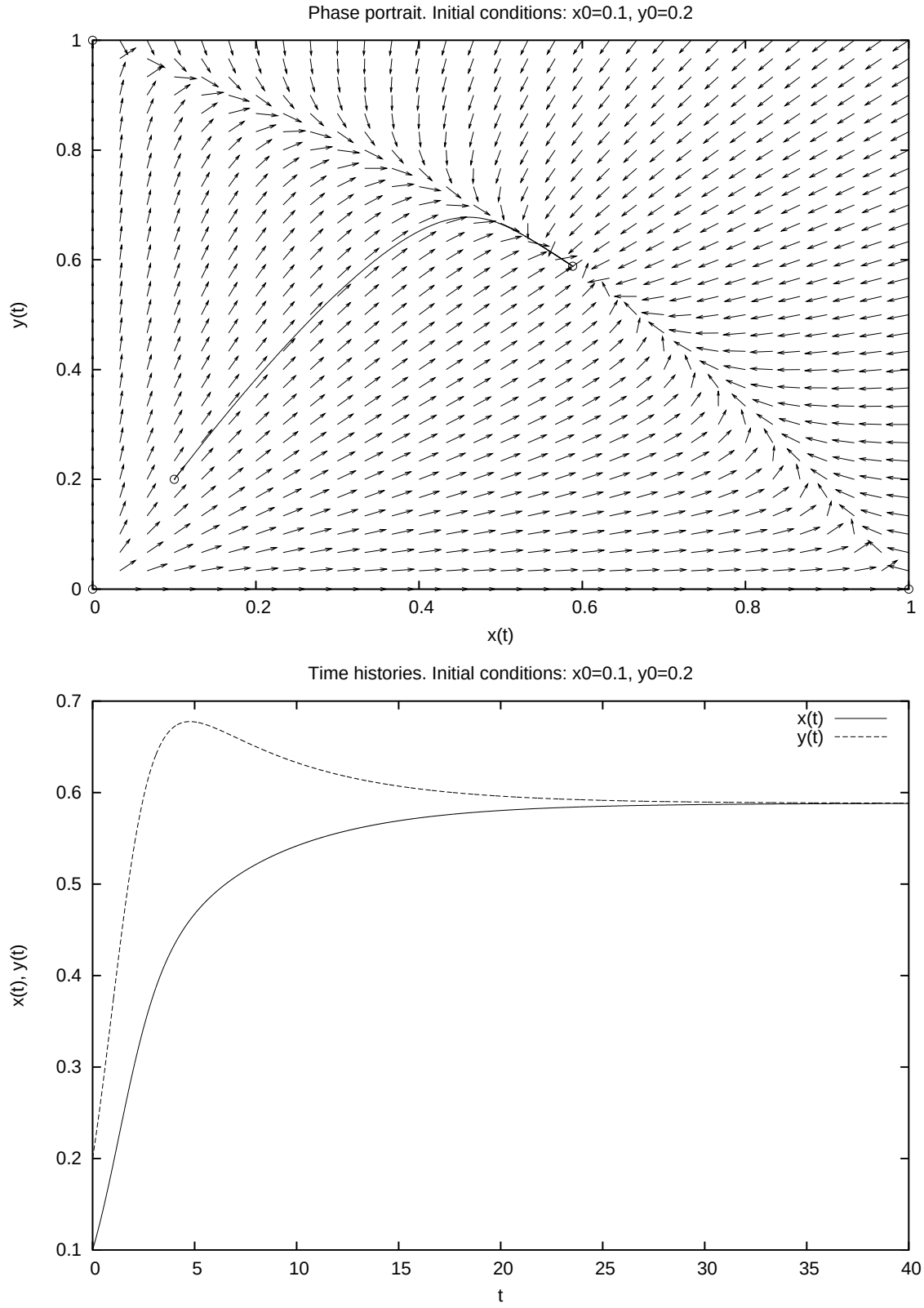

Figura 7: Ritratto di fase e storia temporale per $a = b = 0.7$, $(x_0, y_0) = (0.1, 0.2)$ e $t_f = 40$.

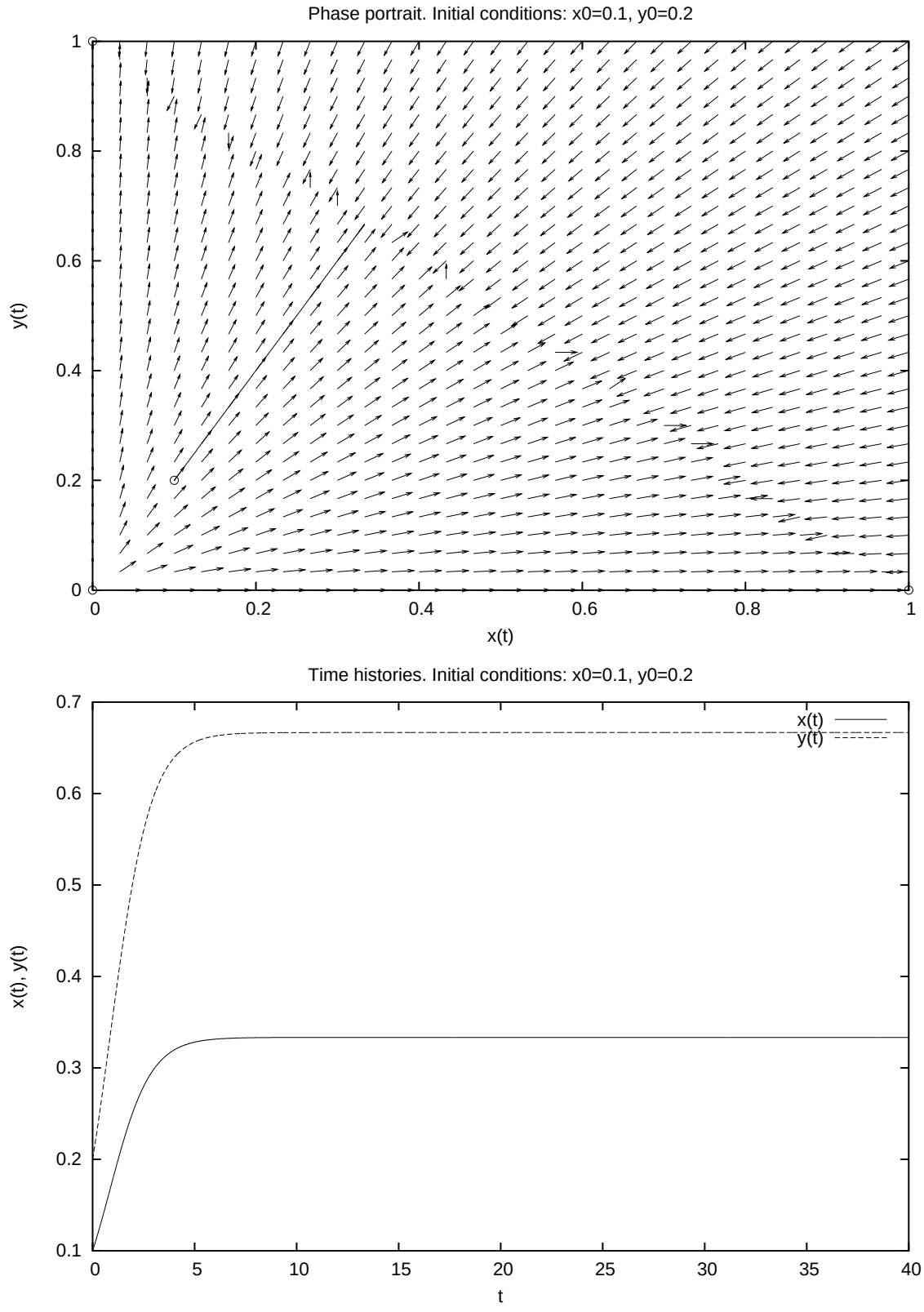

Figura 8: Ritratto di fase e storia temporale per $a = b = 1$, $(x_0, y_0) = (0.1, 0.2)$ e $t_f = 40$.

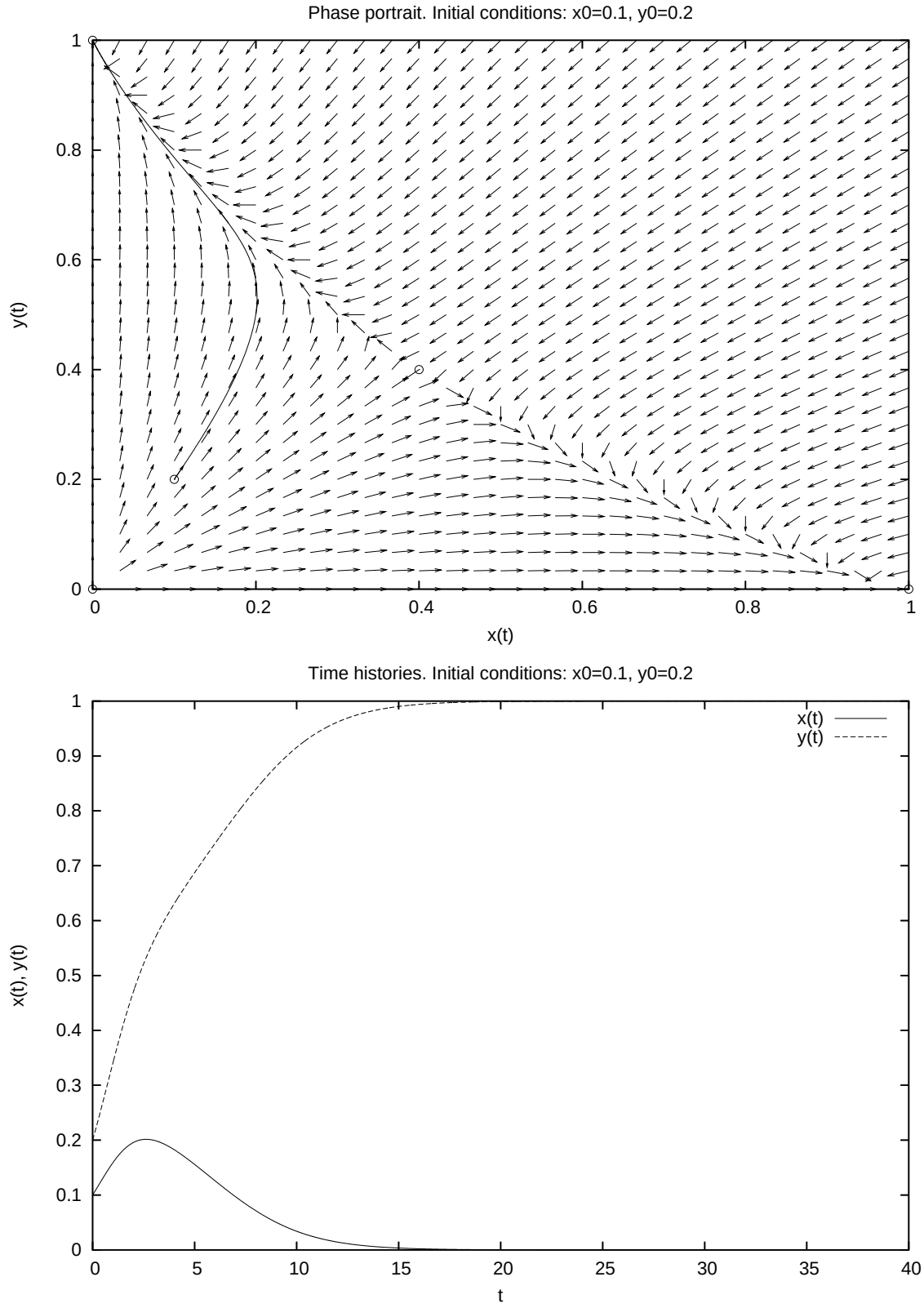

Figura 9: Ritratto di fase e storia temporale per $a = b = 1.5$, $(x_0, y_0) = (0.1, 0.2)$ e $t_f = 40$.

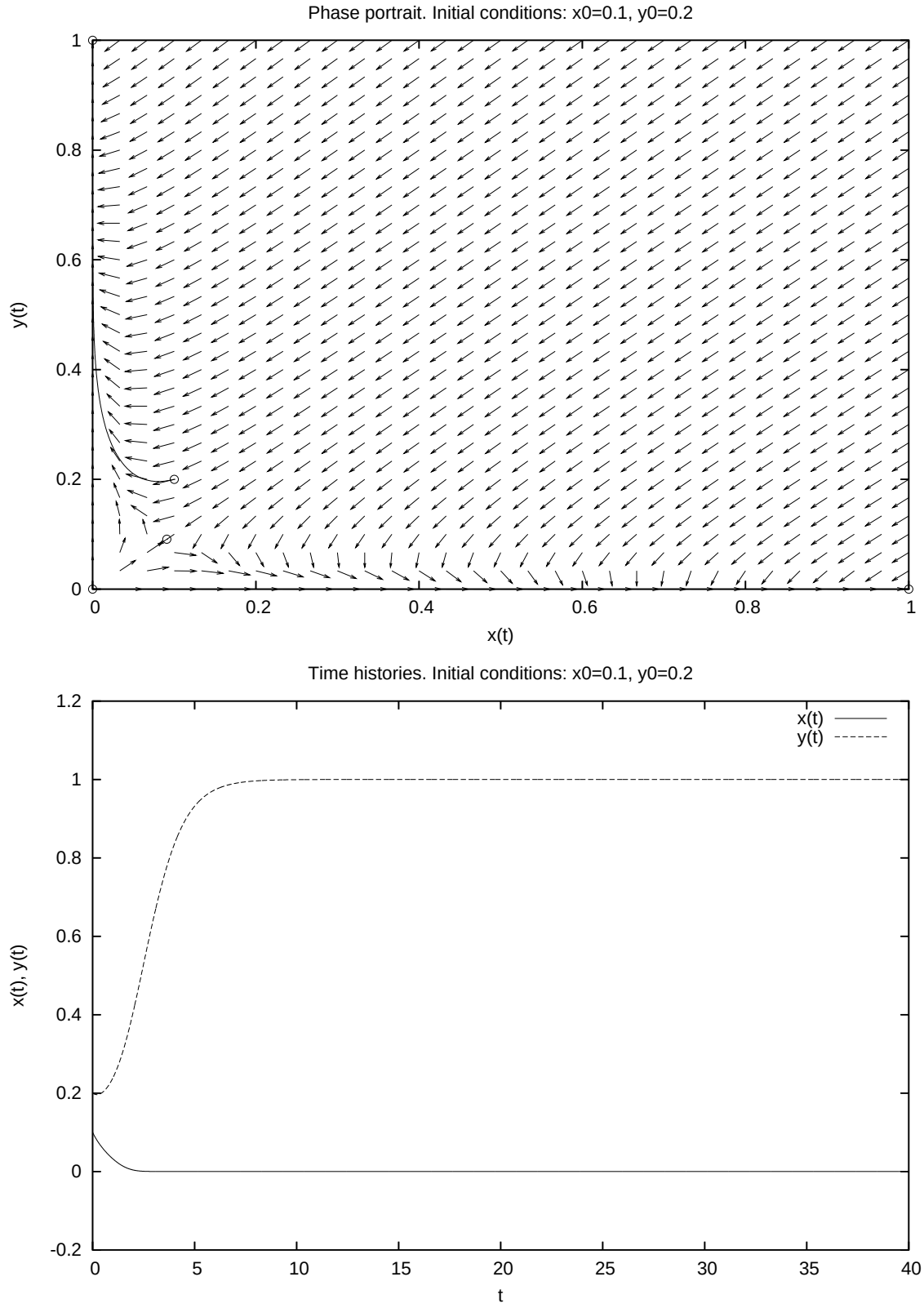

Figura 10: Ritratto di fase e storia temporale per $a = b = 10$, $(x_0, y_0) = (0.1, 0.2)$ e $t_f = 40$.

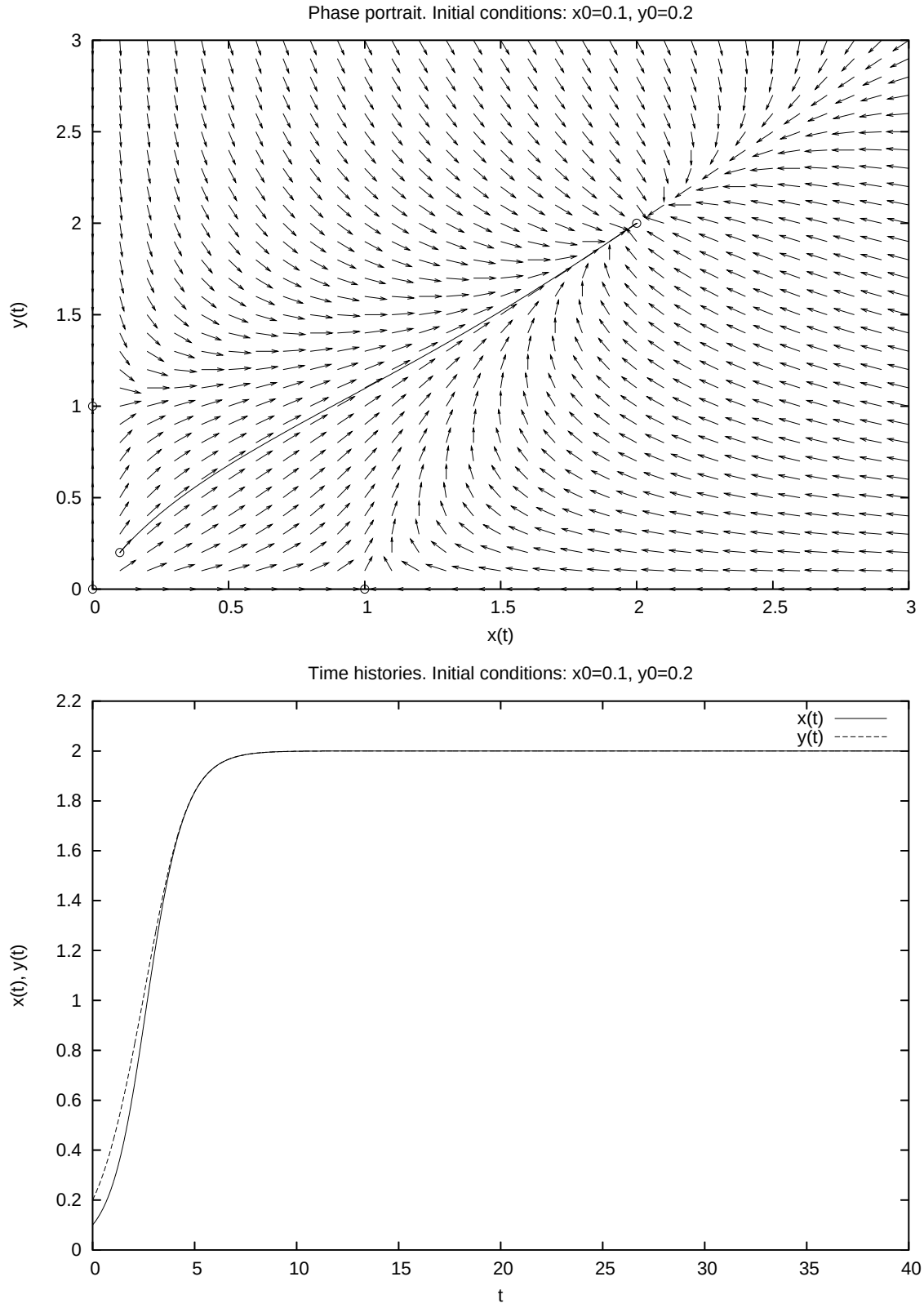

Figura 11: Ritratto di fase e storia temporale per $a = b = -0.5$, $(x_0, y_0) = (0.1, 0.2)$ e $t_f = 40$.

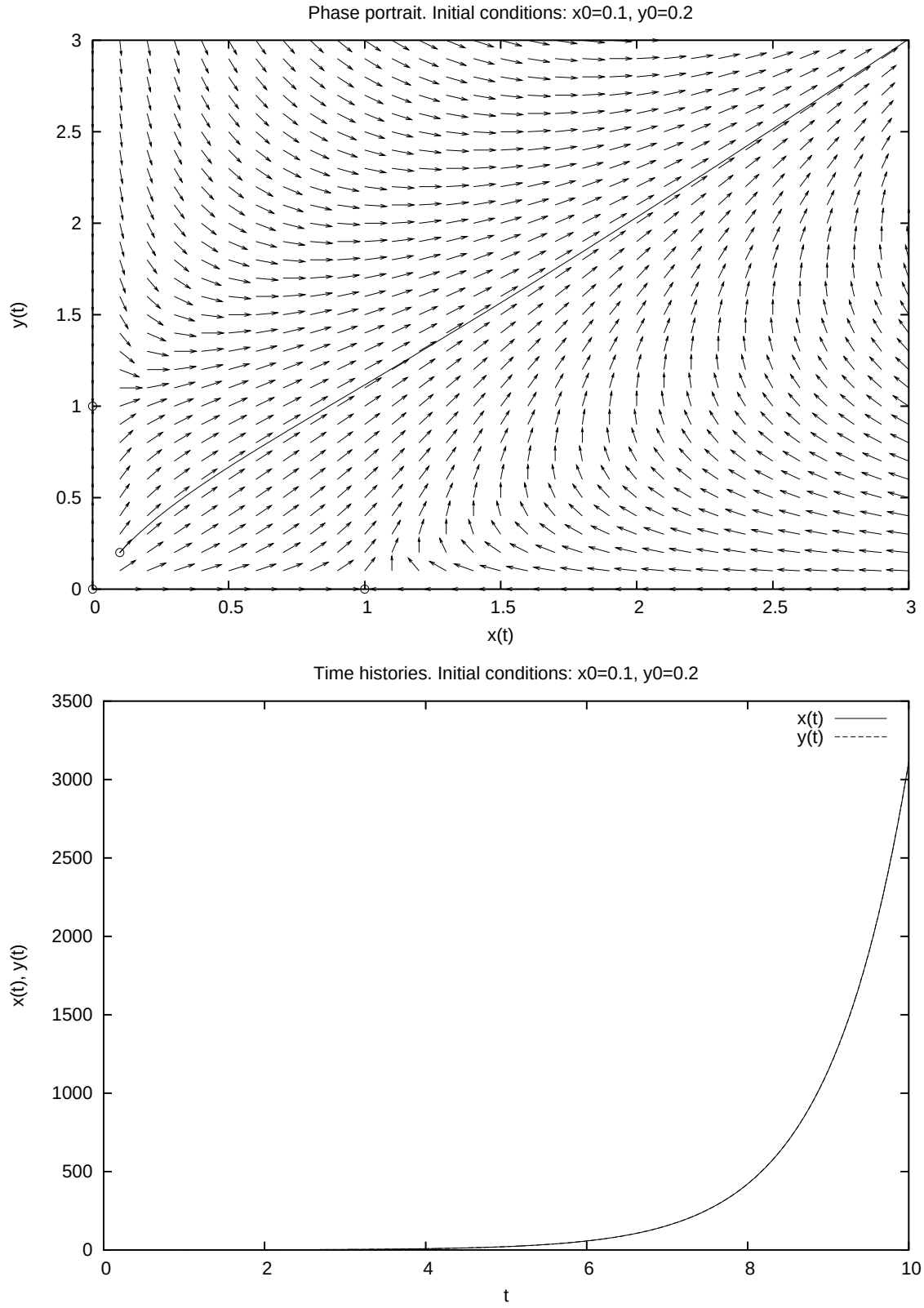

Figura 12: Ritratto di fase e storia temporale per $a = b = -1.0$, $(x_0, y_0) = (0.1, 0.2)$ e $t_f = 10$.

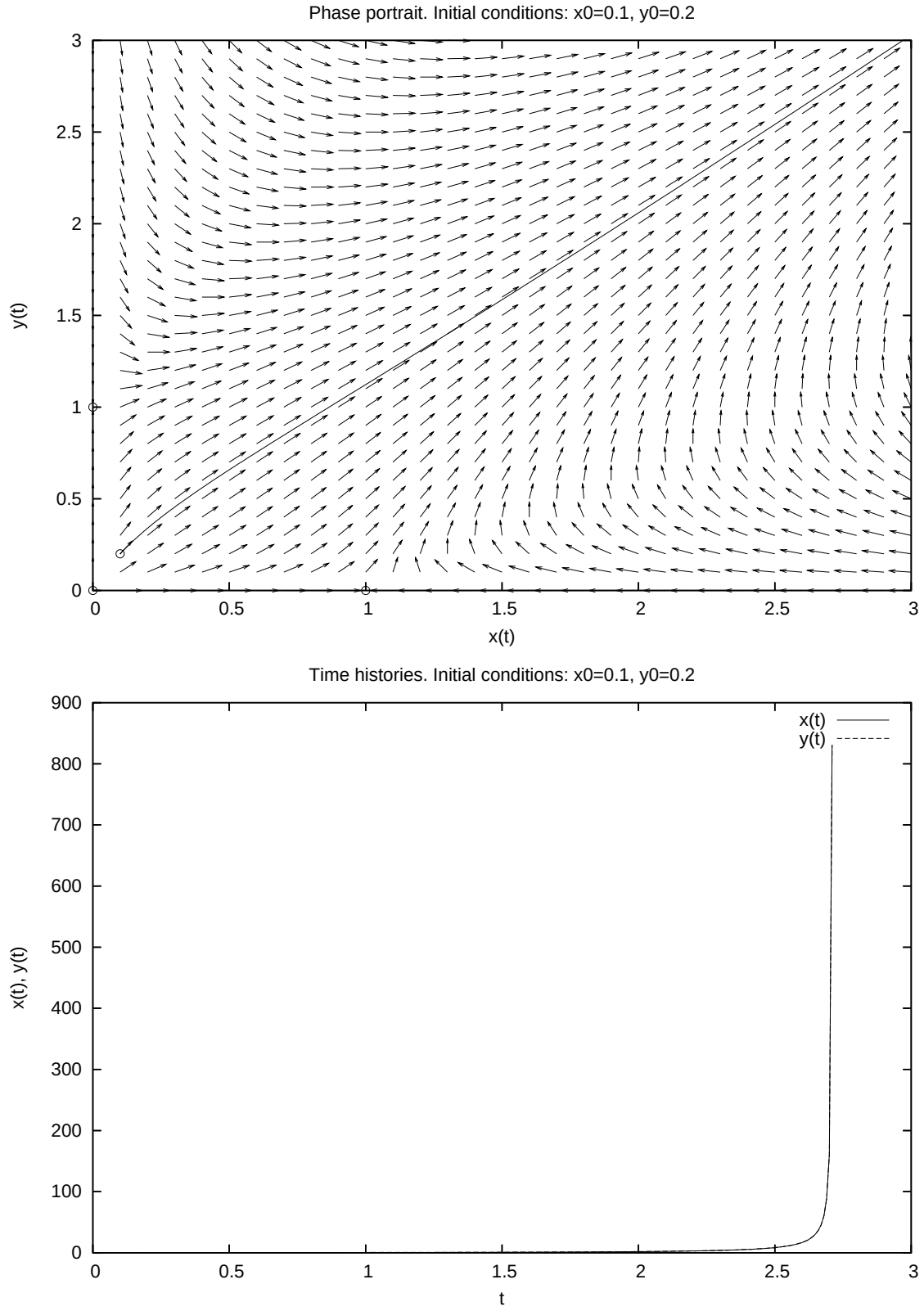

Figura 13: Ritratto di fase e storia temporale per $a = b = -1.5$, $(x_0, y_0) = (0.1, 0.2)$ e $t_f = 2.71$.

1.4 Esercizio

Studiare, al variare delle condizioni iniziali e del tempo finale, il comportamento del sistema (1) per $0 < k \leq 1$.

1.5 Esercizio

Studiare, al variare dei parametri $a, b \in \mathbb{R}, a \neq b$ (positivi e/o negativi), delle condizioni iniziali e del tempo finale, il comportamento del sistema (2).