Modelli Matematici per la Biologia – Esercitazione 3
a.a. 2006-2007

Dott. Simone Zuccher

04 Maggio 2007

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

1 Ritratti di fase e traiettorie di alcuni sistemi dinamici simulati al calcolatore

1.1 Esercizio

Studiare, al variare delle condizioni iniziali e del tempo finale, il comportamento del sistema

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

1.1.1 Risoluzione

L'unico punto di equilibrio per il sistema dato è l'origine. Esso risulta essere un fuoco stabile fintanto che la distanza della condizione iniziale dall'origine è minore di 1 (orbite spiraleggianti verso l'origine). I punti della circonferenza di raggio unitario, invece, formano un ciclo instabile che comunque porta la soluzione nell'origine per tempi lunghi. Infine, partendo da punti con distanza dall'origine maggiore di 1 il sistema diverge in tempo finito.

Tutte queste caratteristiche si possono vedere facilmente moltiplicando la prima equazione per x, la seconda per y, sommando le due equazioni ottenute ed indicando con u la somma dei quadrati di x e y, (i.e. il quadrato della distanza dall'origine). Si ottiene così l'equazione differenziale ordinaria $u' = u(u-1)$, la cui soluzione esplode in tempo finito se si parte da dati iniziali tali per cui $u > 1$ (ossia con distanza dall'origine maggiore di 1 e quindi fuori dalla circonferenza unitaria). La soluzione, quindi, non esiste per tempi grandi e, numericamente, si può facilmente determinare il tempo di *blow up*.

Di seguito sono riportati alcuni esempi che evidenziano la natura dell'origine come fuoco stabile (figura 1), il ciclo instabile $x^2+y^2=1$ (figura 2), e la divergenza del sistema

in tempo finito (figure 3 e 4). Si noti, confrontando le figure 3 e 4, che il tempo di blow-up diminuisce al crescere della distanza dall'origine per la condizione iniziale.

Utilizzando il file `sys1.m` per GNU Octave di seguito riportato si possono fare diverse prove al variare della condizione iniziale e del tempo finale di integrazione. **Si ricordi di cambiare gli estremi della finestra di visualizzazione dipendentemente dalla condizione iniziale**.

```
% Name:     sys1.m
% Author:   Simone Zuccher
% Created:  03 May 2007
% Purpose:  Solve the ODE system
%           x'=y+x(x^2+y^2-1)
%           y'=-x+y(x^2+y^2-1)
%           given the initial condition x0 and y0
% Input:    Initial condition [x0 y0], final time tfinal
% Output:   1. plot of x(t) versus y(t) together with the vector field
%           2. plot time histories of x(t) and y(t)
% Modified:
%
% The equilibrium points are the following:
%
% [x = 0, y = 0],
%

% Clear all variables
clear all;

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

% Equilibrium points
eq = [0 0];
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=limitc(x, t)
  u = x(1);
  v = x(2);
  xdot(1) = v+u*(u^2+v^2-1);
  xdot(2) = -u+v*(u^2+v^2-1);
endfunction

__gnuplot_set__ nokey

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

[X, Y] = meshgrid(xmin:dx:xmax, ymin:dy:ymax);
DX = Y+X.*(X.^2 + Y.^2 - 1);
DY = -X+Y.*(X.^2 + Y.^2 - 1);
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("limitc", 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);')
```

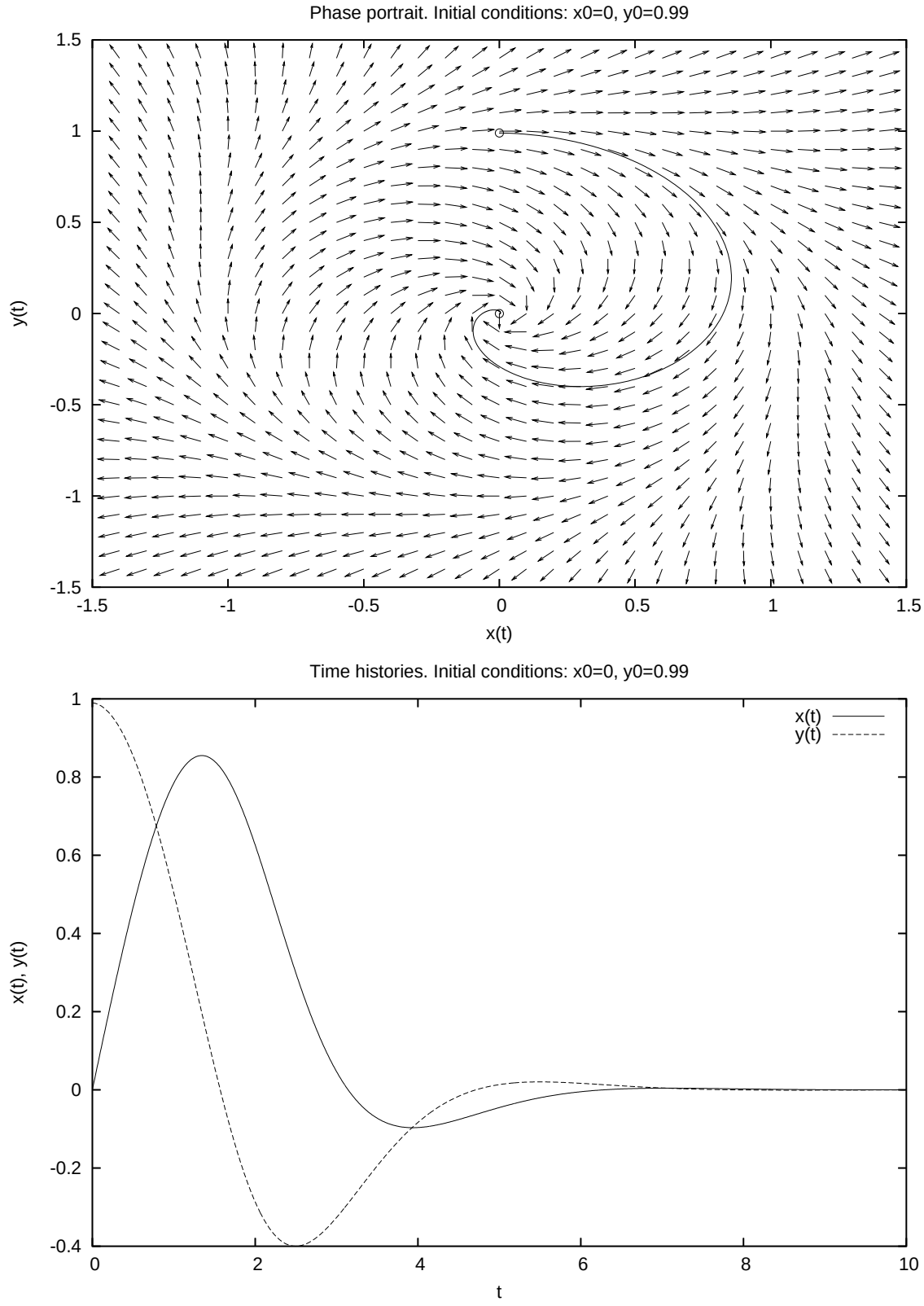

Figura 1: Ritratto di fase e storia temporale per $(x_0, y_0) = (0, 0.99)$ e $t_f = 10$.

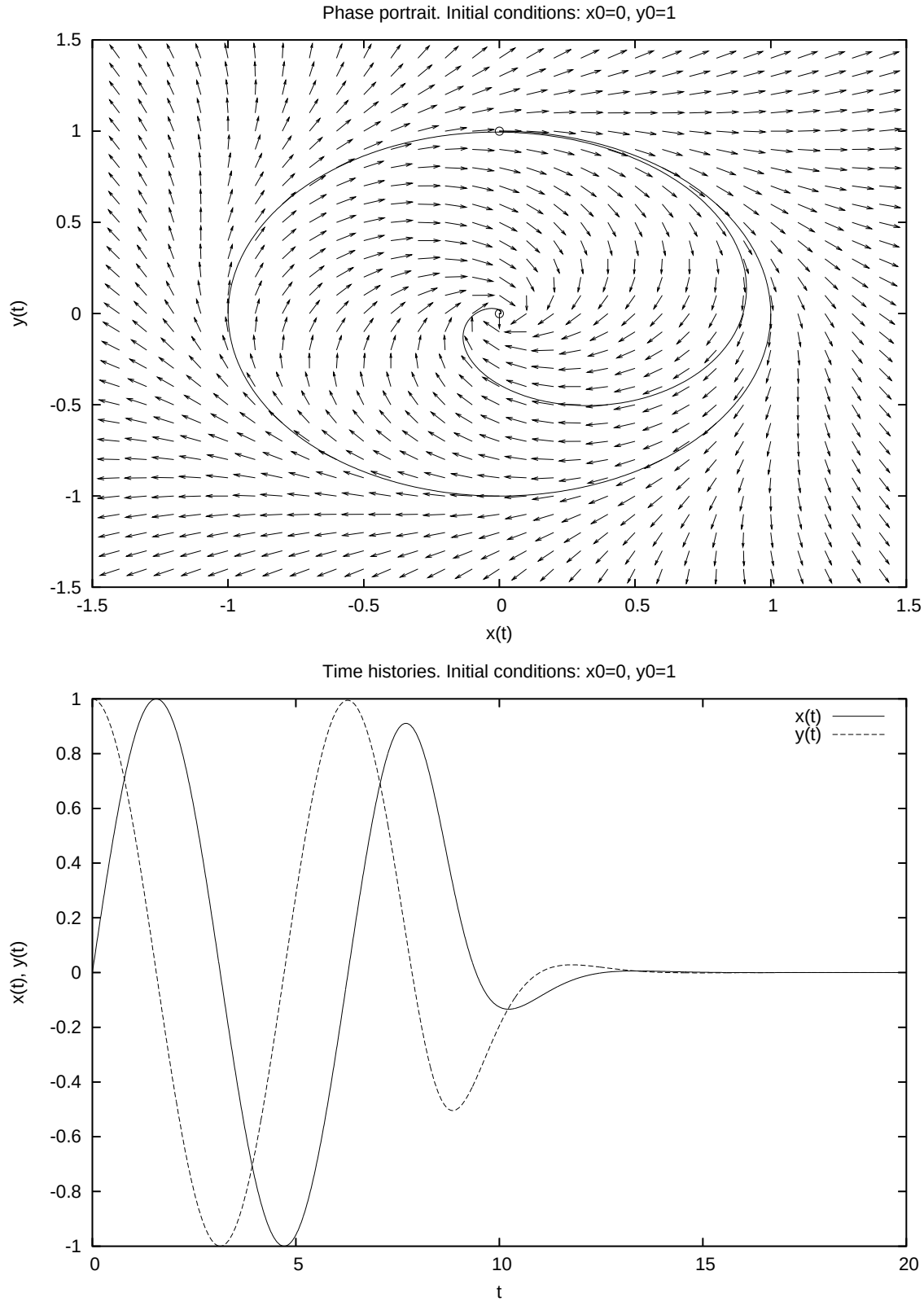

Figura 2: Ritratto di fase e storia temporale per $(x_0, y_0) = (0, 1)$ e $t_f = 20$.

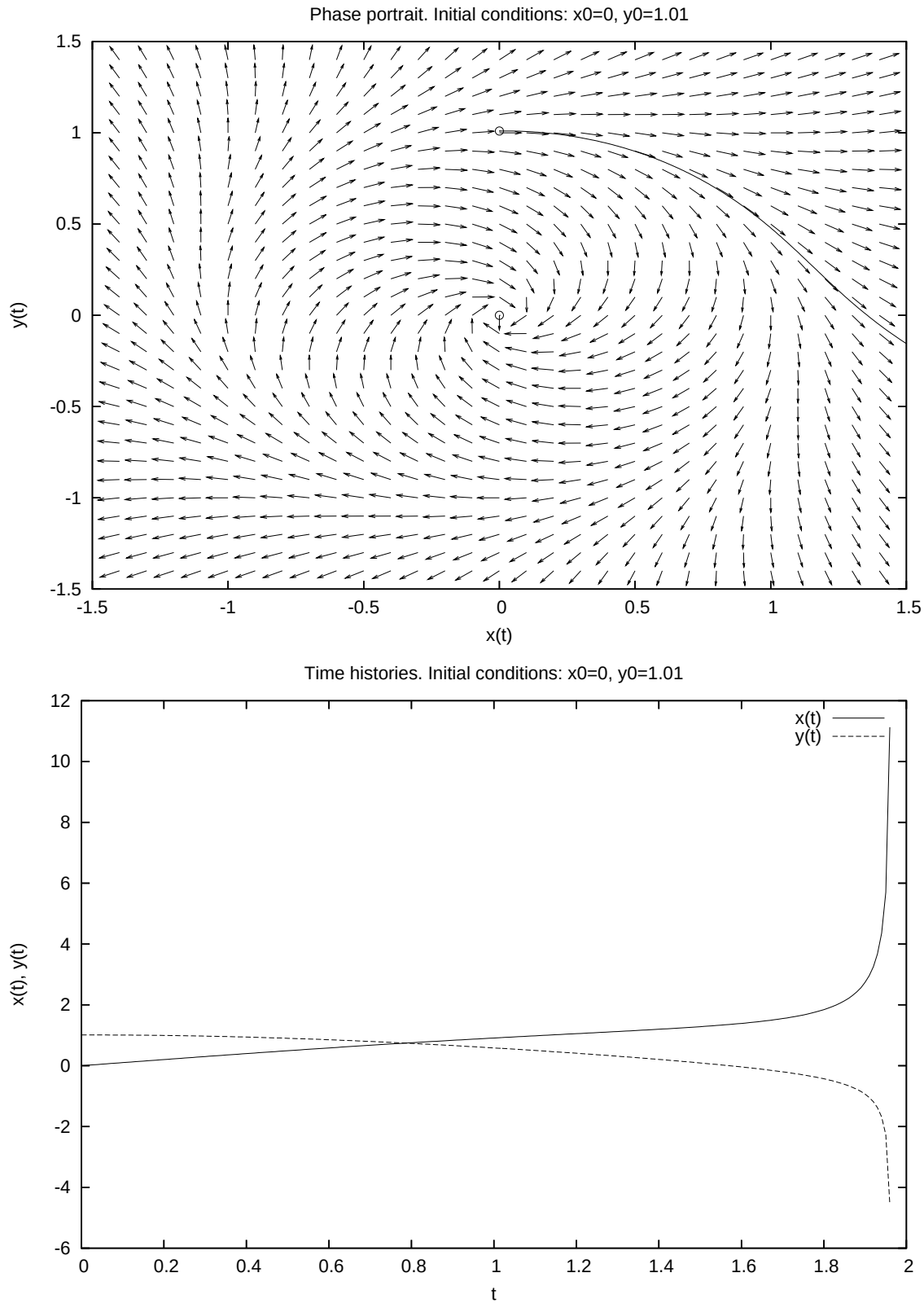

Figura 3: Ritratto di fase e storia temporale per $(x_0, y_0) = (0, 1.01)$ e $t_f = 1.96$.

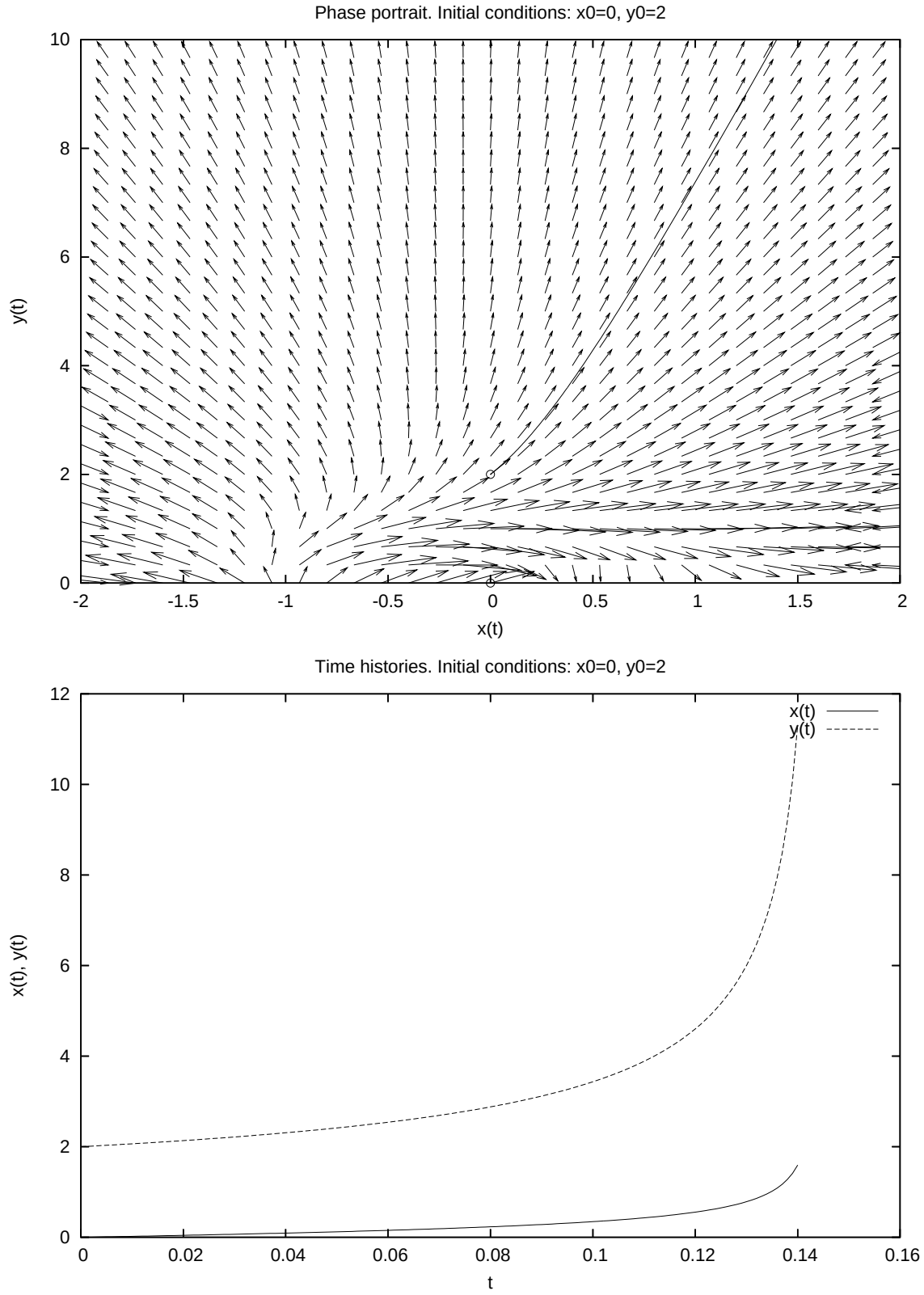

Figura 4: Ritratto di fase e storia temporale per $(x_0, y_0) = (0, 2)$ e $t_f = 0.14$.

1.2 Esercizio

Studiare, al variare delle condizioni iniziali e del tempo finale, il comportamento del sistema

$$\begin{cases} x' &= y + x(\cos(x^2+y^2)-1)(\sin(x^2+y^2)-1) \\ y' &= -x + y(\cos(x^2+y^2)-1)(\sin(x^2+y^2)-1) \end{cases}$$

1.2.1 Risoluzione

L'unico punto di equilibrio per il sistema dato è l'origine. Per capirne la natura, basta studiare l'equazione differenziale ordinaria della distanza dall'origine al quadrato ($u = x^2+y^2$) ricavata moltiplicando la prima equazione per x, la seconda per y, e sommandole. Così facendo si ottiene $u' = 2u(\cos u - 1)(\sin u - 1)$, da cui $u' \geq 0$ sempre, tranne quando $u = 2k\pi$ oppure $u = \pi/2 + 2k\pi, k \in \mathbb{Z}^+$. Questo significa che la distanza dall'origine non diminuisce mai e le orbite vengono catturate da circonferenze del tipo

$$x^2 + y^2 = 2k\pi \quad \text{e} \quad x^2 + y^2 = \pi/2 + 2k\pi, \qquad k \in \mathbb{Z}^+$$

che sono, quindi, dei cicli semi-stabili.

Utilizzando il file `sys2.m` di seguito riportato si possono visualizzare i diversi comportamenti al variare della condizione iniziale e del tempo finale di integrazione. In particolare, si può analizzare cosa succede partendo da condizioni iniziali (x_0, y_0) tali per cui la loro distanza dall'origine $r = \sqrt{x_0^2 + y_0^2}$ varia in certi range. Per fissare le idee, si può prendere $x_0 = 0$ e variare y_0.

- $0 < r < 0.25$. Si osserva un allontanamento dall'origine piuttosto lento, come riportato in figura 5.
- $0 < r < \sqrt{\pi/2} \approx 1.253$. Si osserva un allontanamento dall'origine piuttosto veloce e lo stabilizzarsi sull'orbita distante dall'origine $\sqrt{\pi/2}$, come riportato in figura 6.
- $\sqrt{\pi/2} \approx 1.253 < r < \sqrt{2\pi} \approx 2.5$. Si osserva un allontanamento dall'origine piuttosto veloce e lo stabilizzarsi sull'orbita distante dall'origine $\sqrt{2\pi}$, come riportato in figura 7.

Si ricordi di cambiare gli estremi della finestra di visualizzazione dipendentemente dalla condizione iniziale.

```
% Name:     sys2.m
% Author:   Simone Zuccher
% Created:  03 May 2007
% Purpose:  Solve the ODE system
%           x'=y+x(cos(x^2+y^2)-1)(sin(x^2+y^2)-1)
%           y'=-x+y(cos(x^2+y^2)-1)(sin(x^2+y^2)-1)
%           given the initial condition x0 and y0
% Input:    Initial condition [x0 y0], final time tfinal
% Output:   1. plot of x(t) versus y(t) together with the vector field
%           2. plot time histories of x(t) and y(t)
```

```
% Modified:
%
% The equilibrium points are the following:
%
% [x = 0, y = 0],
%

% Clear all variables
clear all;

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

% Equilibrium points
eq = [0 0];
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)
  u = x(1);
  v = x(2);
  xdot(1) = v+u*(cos(u^2+v^2)-1)*(sin(u^2+v^2)-1);
```

```
  xdot(2) = -u+v*(cos(u^2+v^2)-1)*(sin(u^2+v^2)-1);
endfunction

__gnuplot_set__ nokey
axis([xmin xmax ymin ymax]);

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

DX = Y+X.*(cos(X.^2 + Y.^2) - 1).*(sin(X.^2 + Y.^2) - 1);
DY = -X+Y.*(cos(X.^2 + Y.^2) - 1).*(sin(X.^2 + Y.^2) - 1);
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);')
```

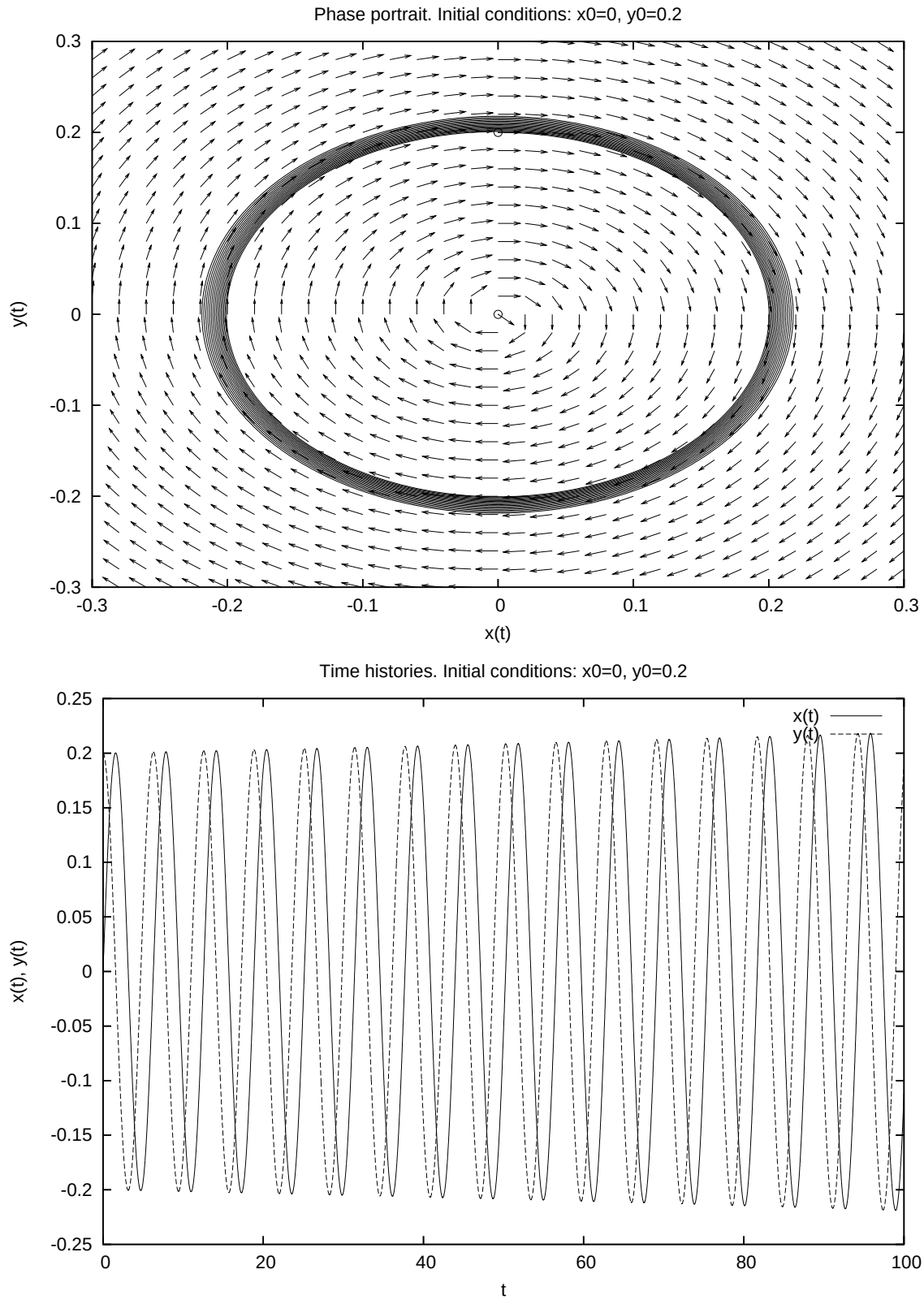

Figura 5: Ritratto di fase e storia temporale per $(x_0, y_0) = (0, 0.2)$ e $t_f = 100$.

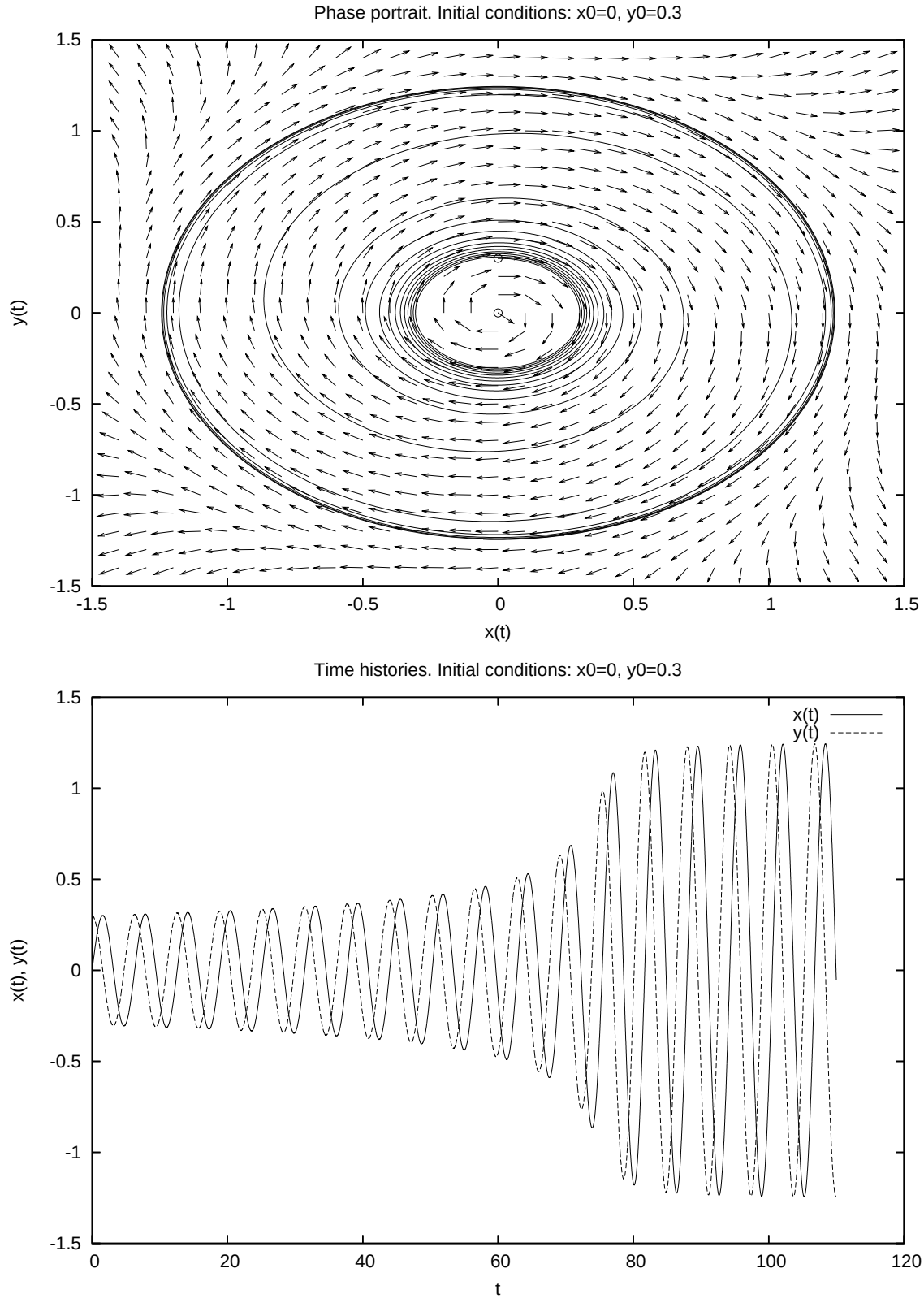

Figura 6: Ritratto di fase e storia temporale per $(x_0, y_0) = (0, 0.3)$ e $t_f = 110$.

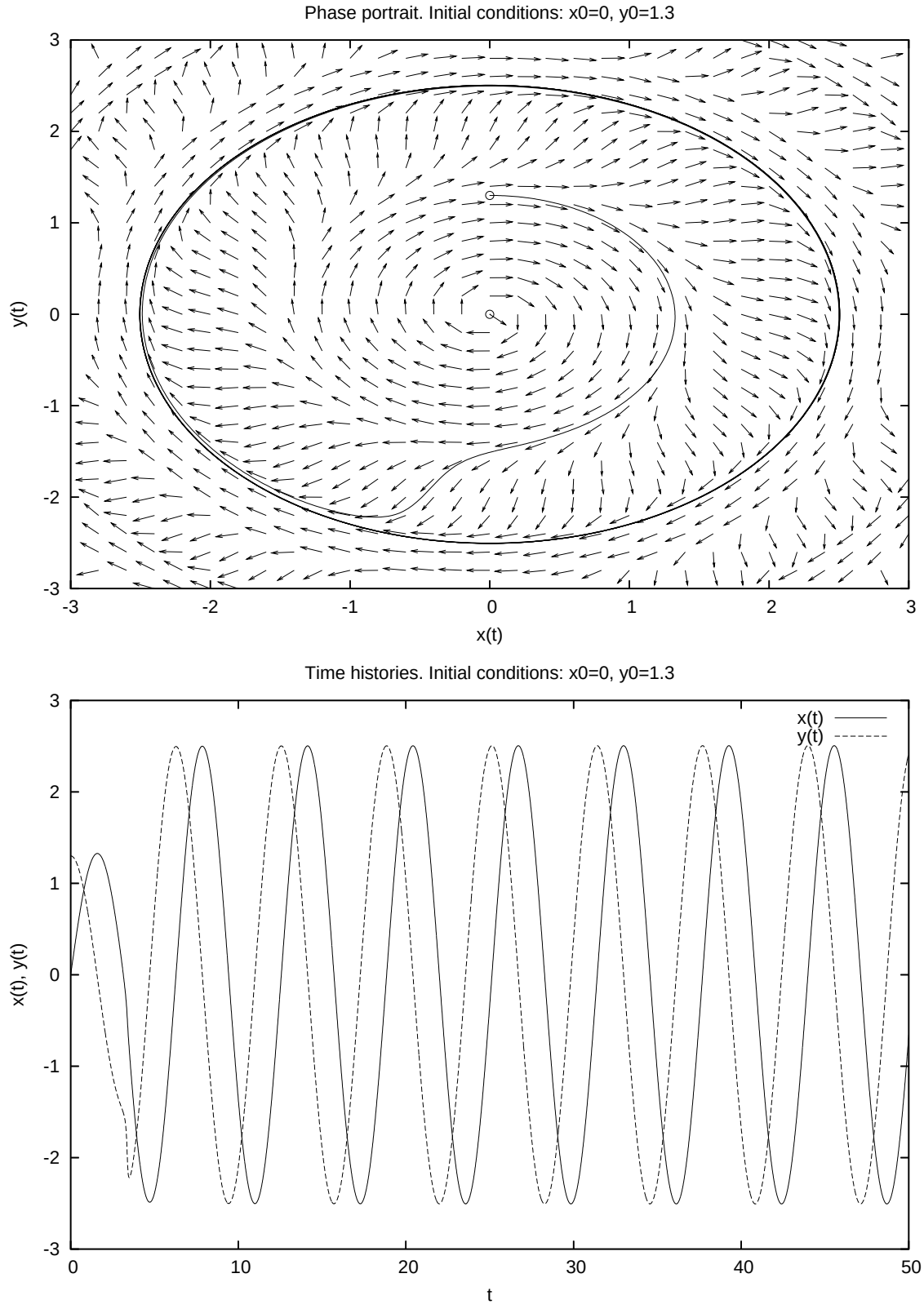

Figura 7: Ritratto di fase e storia temporale per $(x_0, y_0) = (0, 1.3)$ e $t_f = 50$.

1.3 Esercizio

Studiare, al variare delle condizioni iniziali e del tempo finale, il comportamento del sistema

$$\begin{cases} x' &= -y + x(x^2+y^2)\cos(\pi/(x^2+y^2)) \\ y' &= x + y(x^2+y^2)\cos(\pi/(x^2+y^2)) \end{cases}$$

1.3.1 Risoluzione

L'unico punto di equilibrio per il sistema dato è l'origine. Per capirne la natura, basta studiare, di nuovo, l'equazione differenziale ordinaria della distanza al quadrato dall'origine ($u = x^2 + y^2$) ricavata moltiplicando la prima equazione per x, la seconda per y, e sommandole. Così facendo si ottiene $u' = 2u^2\cos(\pi/u)$. Quindi, la distanza dall'origine può aumentare o diminuire dipendentemente dal segno di $\cos(\pi/u)$. In particolare, le orbite per le quali è costante la distanza dall'origine si ottengono da $\cos(\pi/u) = 0 \Rightarrow x^2 + y^2 = 2/(2k+1)$, ovvero sono circonferenze concentriche aventi centro nell'origine e raggio $\sqrt{2/(2k+1)}$. La più ampia di queste circonferenze ha raggio $r = \sqrt{2}$, mentre al limite ($k \to +\infty$) il raggio tende a zero.

Per ottenere gli intervalli in cui la triettorie diminuiscono la loro distanza dal centro, basta risolvere la disequazione $\cos(\pi/u) < 0$, che dopo alcuni passaggi porta a $\sqrt{2/(4k+3)} < r < \sqrt{2/(4k+1)}$. Ovviamente, per valori di r positivi e complementari ai precedenti le traiettorie si allontanano dall'origine, i.e. per $\sqrt{(2/(4 * k + 5))} < r < \sqrt{2/(4k+3)}$. Riassumendo:

- per condizioni iniziali aventi distanza dall'origine $r > \sqrt{2}$ le traiettorie divergono molto velocemente;
- per condizioni iniziali tali per cui $\sqrt{2/(4k+3)} < r < \sqrt{2/(4k+1)}$, ovvero comprese in certe corone circolari, la distanza dall'origine decresce ma vengono catturate dalla circonferenza più interna della corona avente $r = \sqrt{2/(4k+3)}$;
- per condizioni iniziali tali per cui $\sqrt{2/(4k+5)} < r < \sqrt{2/(4k+3)}$, ovvero comprese nelle corone circolari complementari alle precedenti, la distanza dall'origine cresce ma vengono catturate dalla circonferenza più esterna della corona avente $r = \sqrt{2/(4k+3)}$.

Utilizzando il file `sys3.m`, di seguito riportato, si facciano diverse prove al variare della condizione iniziale e del tempo finale di integrazione. In particolare, si discuta cosa succede partendo da condizioni iniziali (x_0, y_0) tali per cui la loro distanza dall'origine $r = \sqrt{x_0^2 + y_0^2}$ varia in certi range. Per fissare le idee, si può prendere $x_0 = 0$ e variare y_0. **Si ricordi di cambiare gli estremi della finestra di visualizzazione dipendentemente dalla condizione iniziale**.

- $r > \sqrt{4}$. Si osserva un rapido allontanamento dall'origine come riportato in figura 8.
- $\sqrt{2/(4k+3)} < r < \sqrt{2/(4k+1)}$. Se si prende $k = 0$ questo significa $\sqrt{2/3} < r < \sqrt{2}$, e si ottiene un rapido collasso sull'orbita più interna, i.e. la circonferenza avente raggio $r = \sqrt{2/(4k+3)} \approx 0.81650$, come mostrato in figura 9.

- $\sqrt{2/(4k+5)} < r < \sqrt{2/(4k+3)}$. Se si prende $k = 0$ questo significa $sqrt2/5 < r < \sqrt{2/3}$, e si ottiene un rapido collasso sull'orbita più esterna, i.e. la circonferenza avente raggio $r = \sqrt{2/(4k+3)} \approx 0.81650$, come mostrato in figura 10.

- Limite per $r \to 0$. In questo caso si parte da condizioni iniziali vicine all'origine, per cui il comportamento è determinato da quanto succede per $k \to +\infty$. Si osserva che, al tendere di k a $+\infty$, le quantità $\sqrt{2/(4k+5)}$, $\sqrt{2/(4k+3)}$, $\sqrt{2/(4k+1)}$ tendono tutte a 0 e si "compattano" molto velocemente. Questo significa che, indipendentemente da dove si parte, si finisce su una circonferenza prossima alla condizione iniziale. In questo senso, l'origine è stabile ma non asintoticamente poiché le triettorie possono allontanarsi o avvicinarsi ad essa (dipendentemente dalla discussione precedente) ma rimangono comunque intrappolate in corone di raggio arbitrariamente piccolo. Per esempio, in figura 11 è riportato il caso della condizione iniziale $(x_0, y_0) = (0, 0.0703)$, che rimane intrappolata nelle corone di raggio $r_1 = 0.0702728368926307$ e $r_2 = 0.0704469953676347$, che corrispondono a $k = 100$ e quindi viene attratta verso la circonferenza più esterna, come si può facilmente notare scrivendo a schermo la distanza dall'origine per l'ultimo punto calcolato con il comando `sqrt(x(1,size(x)(2))^2+x(2,size(x)(2))^2)`.

```
% Name:     sys3.m
% Author:   Simone Zuccher
% Created:  03 May 2007
% Purpose:  Solve the ODE system
%           x'=-y+x(x^2+y^2)cos(\pi/(x^2+y^2))
%           y'=x+y(x^2+y^2)cos(\pi/(x^2+y^2))
%           given the initial condition x0 and y0
% Input:    Initial condition [x0 y0], final time tfinal
% Output:   1. plot of x(t) versus y(t) together with the vector field
%           2. plot time histories of x(t) and y(t)
% Modified:
%
% The equilibrium points are the following:
%
% [x = 0, y = 0],
%

% Clear all variables
clear all;

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

% Equilibrium points
eq = [0 0];
```

```
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)
  u = x(1);
  v = x(2);
  xdot(1) = -v+u*(u^2+v^2)*cos(pi/(u^2+v^2));
  xdot(2) = u+v*(u^2+v^2)*cos(pi/(u^2+v^2));
endfunction

__gnuplot_set__ nokey
axis([xmin xmax ymin ymax]);

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

DX = -Y+X.*(X.^2 + Y.^2).*cos(pi./(X.^2 + Y.^2));
DY = X+Y.*(X.^2 + Y.^2).*cos(pi./(X.^2 + Y.^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);')
```

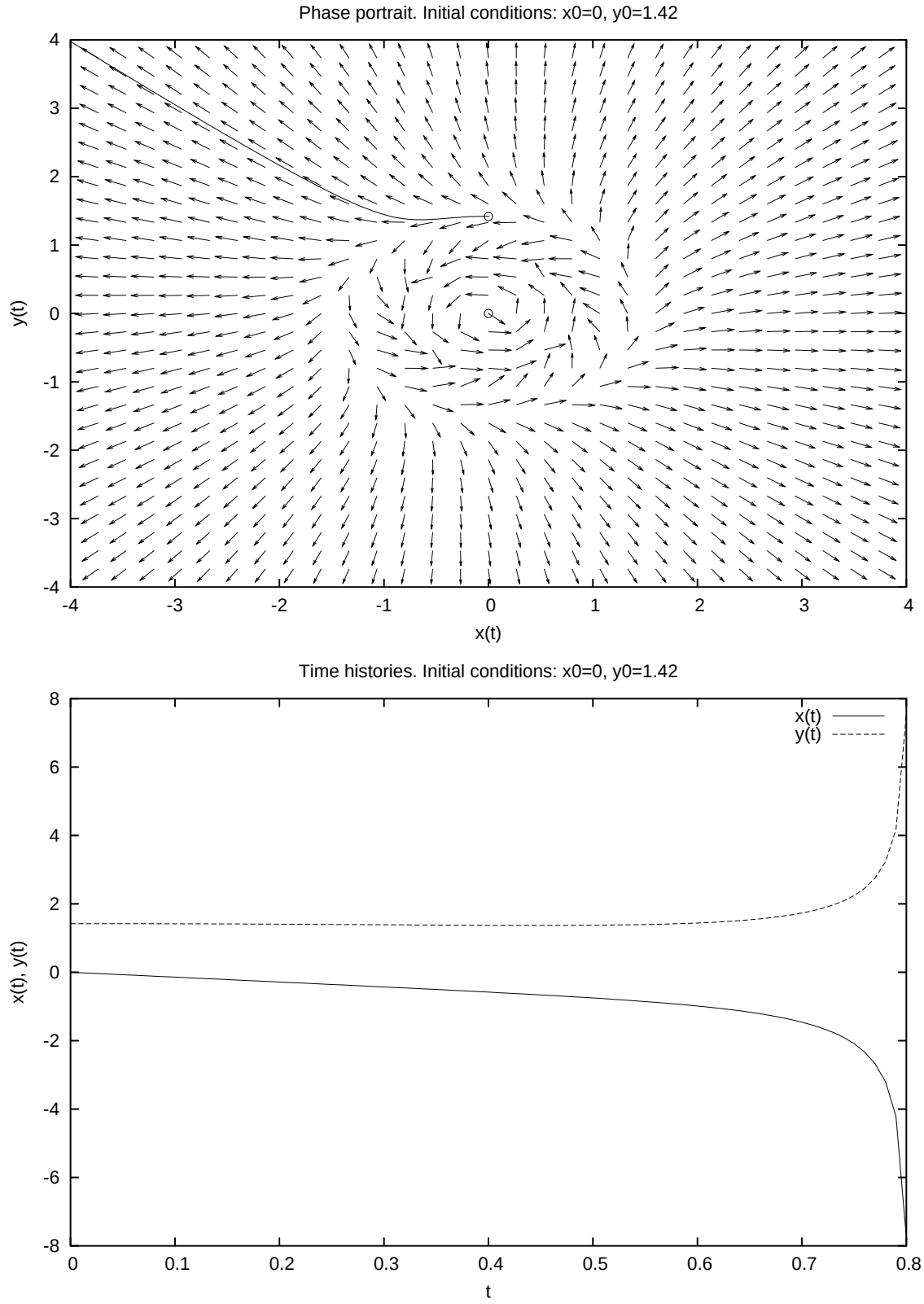

Figura 8: Ritratto di fase e storia temporale per $(x_0, y_0) = (0, 1.42)$ e $t_f = 0.8$.

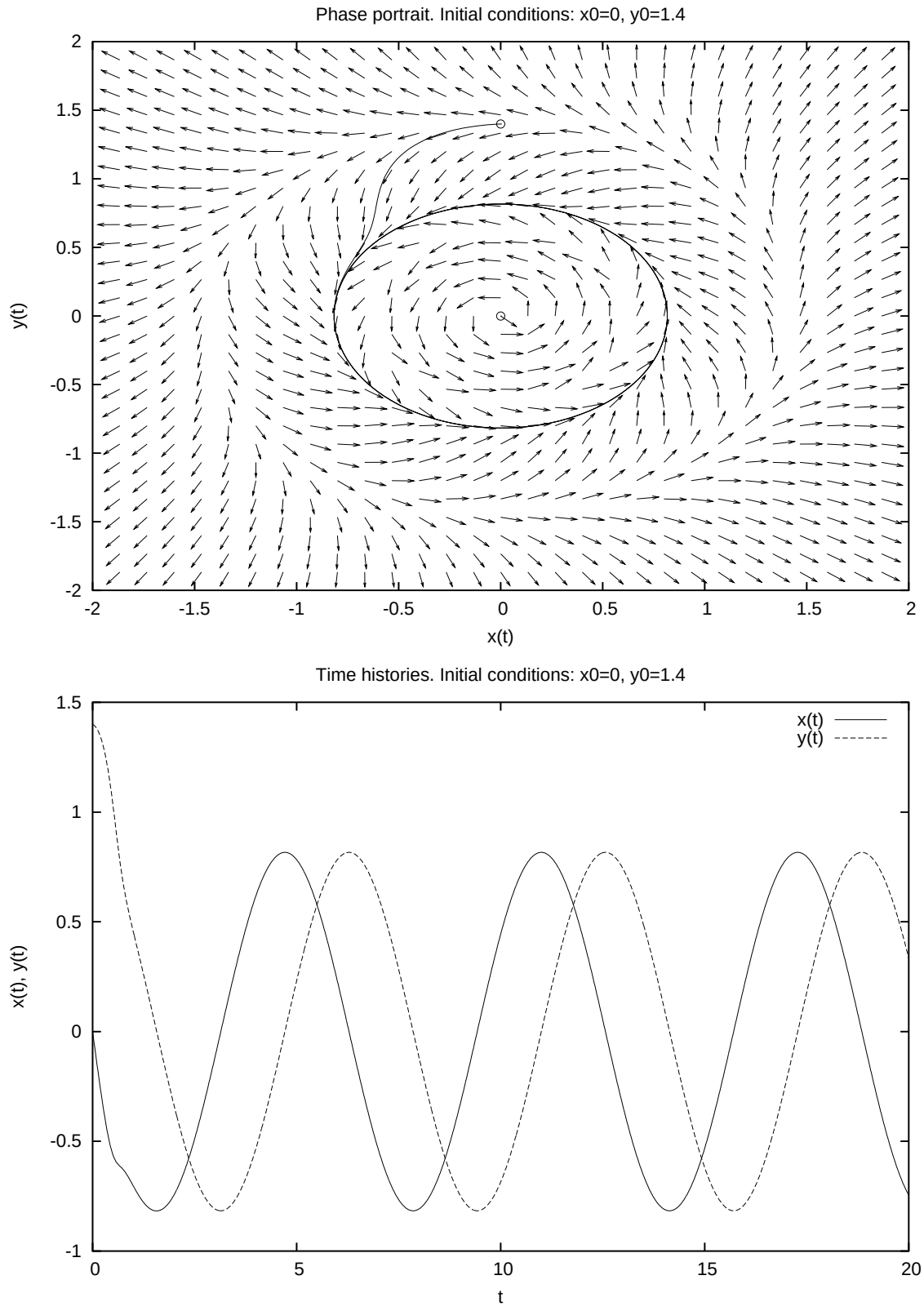

Figura 9: Ritratto di fase e storia temporale per $(x_0, y_0) = (0, 1.4)$ e $t_f = 20$.

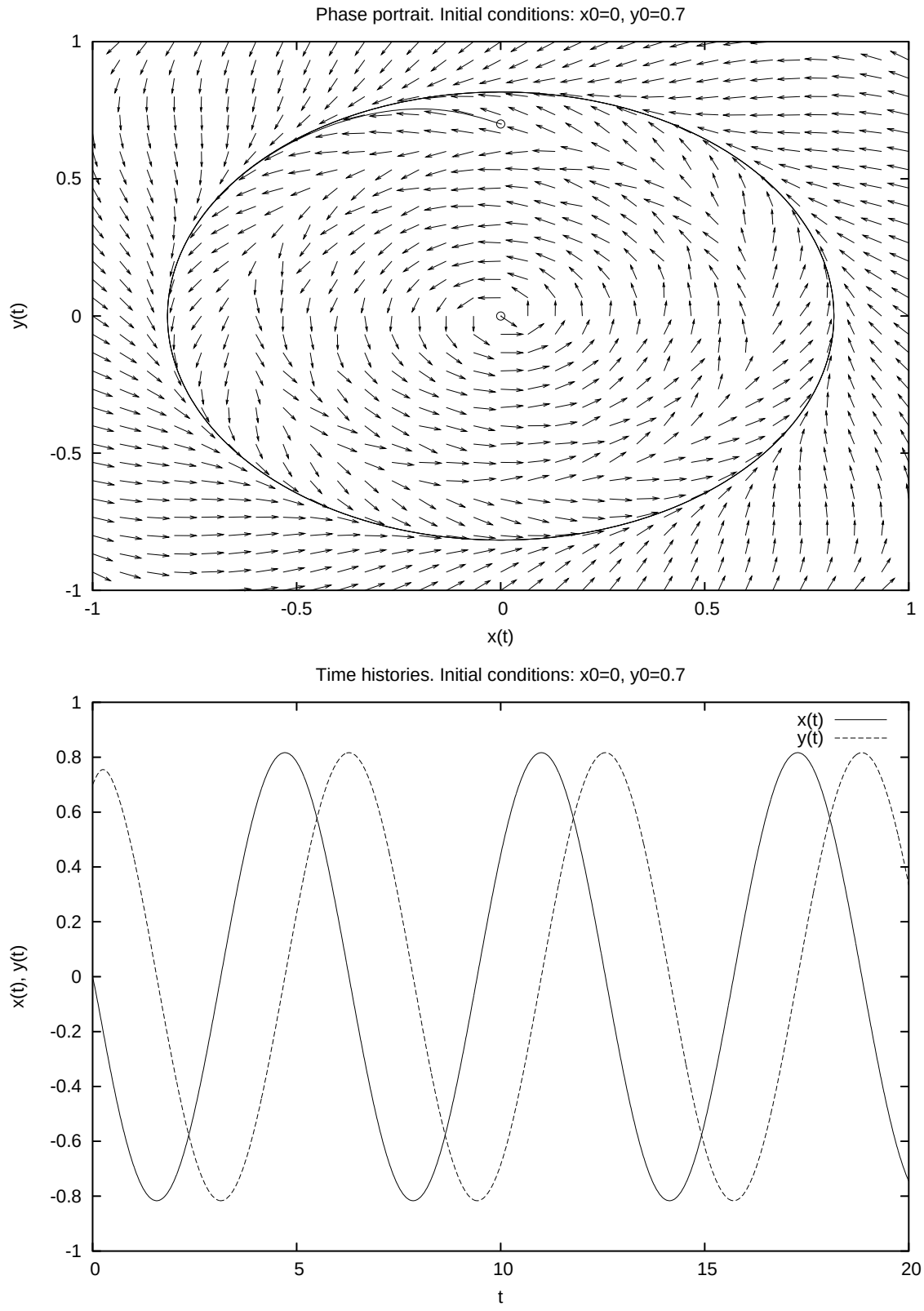

Figura 10: Ritratto di fase e storia temporale per $(x_0, y_0) = (0, 0.7)$ e $t_f = 20$.

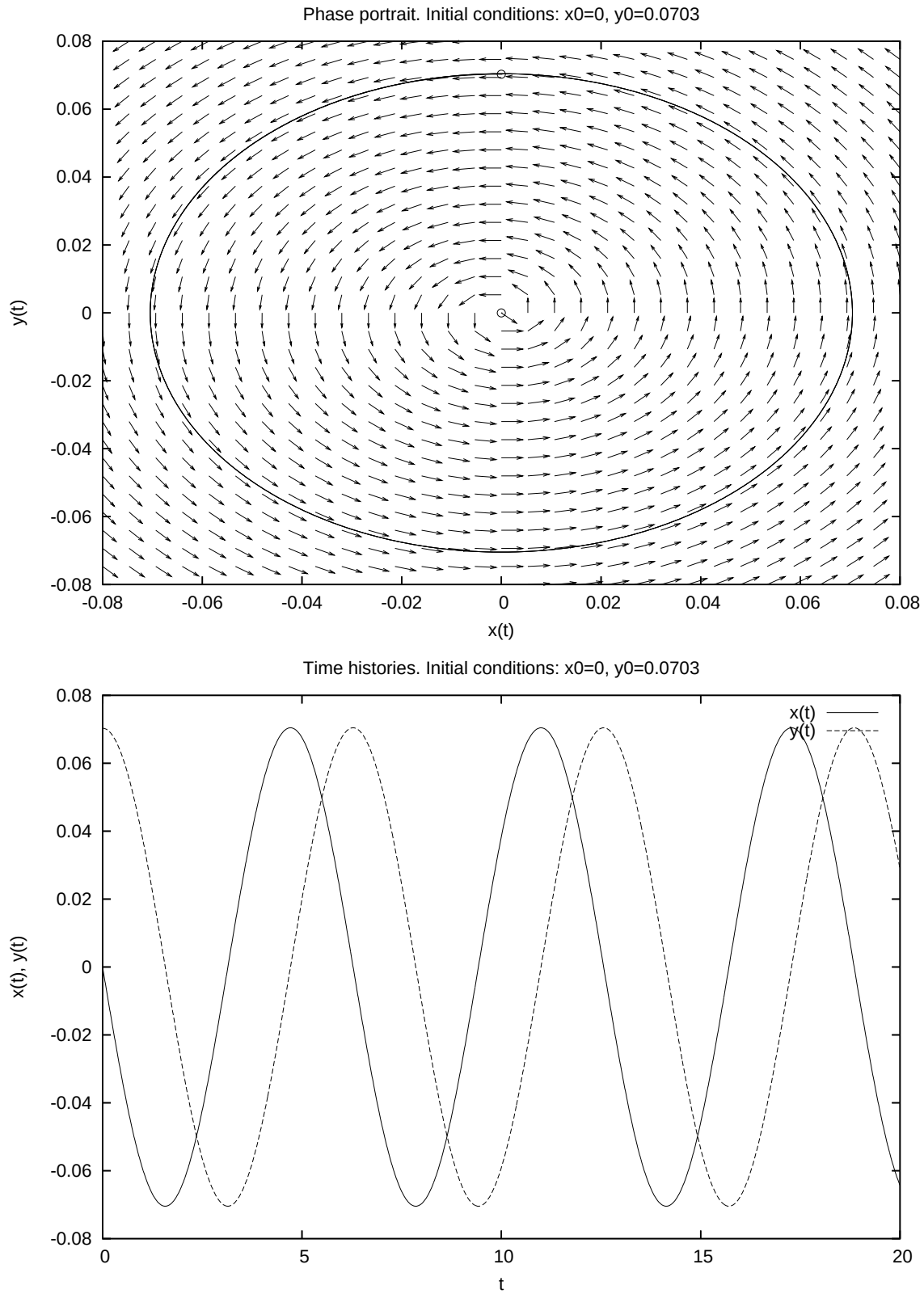

Figura 11: Ritratto di fase e storia temporale per $(x_0, y_0) = (0, 0.0703)$ e $t_f = 20$.

1.4 Esercizio

Studiare, al variare delle condizioni iniziali, del parametro $a > 0$ e del tempo finale, il comportamento dell'equazione (di van der Pool)

$$y'' + a(y^2 - 1)y' + y = 0.$$

1.4.1 Risoluzione

Si noti innanzitutto che l'equazione data può essere riscritta sotto forma di sistema del prim'ordine

$$\begin{cases} x' &= -y - a(y^2 - 1)x \\ y' &= x \end{cases}$$

L'unico punto di equilibrio per il sistema dato è l'origine, che risulta essere un fuoco instabile in quanto le orbite se ne allontanano seguendo una spirale. Si può dimostrare (attraverso l'analisi nel *piano di Lienard*) che esiste un unico ciclo stabile nel senso che ogni traiettoria non passante per l'origine si avvolge su di esso per $t \to +\infty$. Omettiamo qui la dimostrazione, ma procediamo con l'analisi di tipo numerico.

La figura 12 mostra la natura dell'origine, i.e. il suo essere un fuoco instabile ($a = 1$, $(x_0, y_0) = (10^{-4}, 10^{-4})$ e $t_f = 100$). Come si può facilmente intuire guardando il ritratto di fase, partendo da ogni punto interno al ciclo stabile, la traiettoria ne viene attratta. La figura 13 ($a = 1$, $(x_0, y_0) = (-2, 2.1)$ e $t_f = 100$) mostra invece come anche partendo da una condizione iniziale esterna al ciclo stabile, comunque per $t \to +\infty$ la traiettoria ne viene attratta. Infine, in figura 14 viene riportato un caso con $a = 0.1$. Come si può notare, cambiare il parametro a significa essenzialmente cambiare la forma del ciclo stabile ma non la sua natura.

Utilizzanto il file `sys4.m` si osservi il comportamento del sistema al variare dei parametri a disposizione.

```
% Name:     sys4.m
% Author:   Simone Zuccher
% Created:  03 May 2007
% Purpose:  Solve the ODE system (Van der Pool equation)
%           x'=-y-a*x*(y^2-1)
%           y'=x
%           given the initial condition x0 and y0
% Input:    Initial condition [x0 y0], final time tfinal
% Output:   1. plot of x(t) versus y(t) together with the vector field
%           2. plot time histories of x(t) and y(t)
% Modified:
%
% The equilibrium points are the following:
%
% [x = 0, y = 0],
%

% Clear all variables
```

```
clear all;
global a;

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

% Equilibrium points
eq = [0 0];
disp('Equilibrium points:');
disp(eq);

% Set parameter a
a=input('Insert a: ');

% 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 a;
  u = x(1);
  v = x(2);
  xdot(1) = -v-a*u*(v^2-1);
  xdot(2) = u;
endfunction

__gnuplot_set__ nokey
```

```
axis([xmin xmax ymin ymax]);

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

DX = -Y-a*X.*(Y.^2-1);
DY = X;
L = sqrt(DX.^2 + DY.^2);
mytitle=["Phase portrait. Initial conditions: x0=" num2str(x0(1))\
         ", y0=" num2str(x0(2)) ". a=" num2str(a) "."];
__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)) ". a=" num2str(a) "."];
__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);')
```

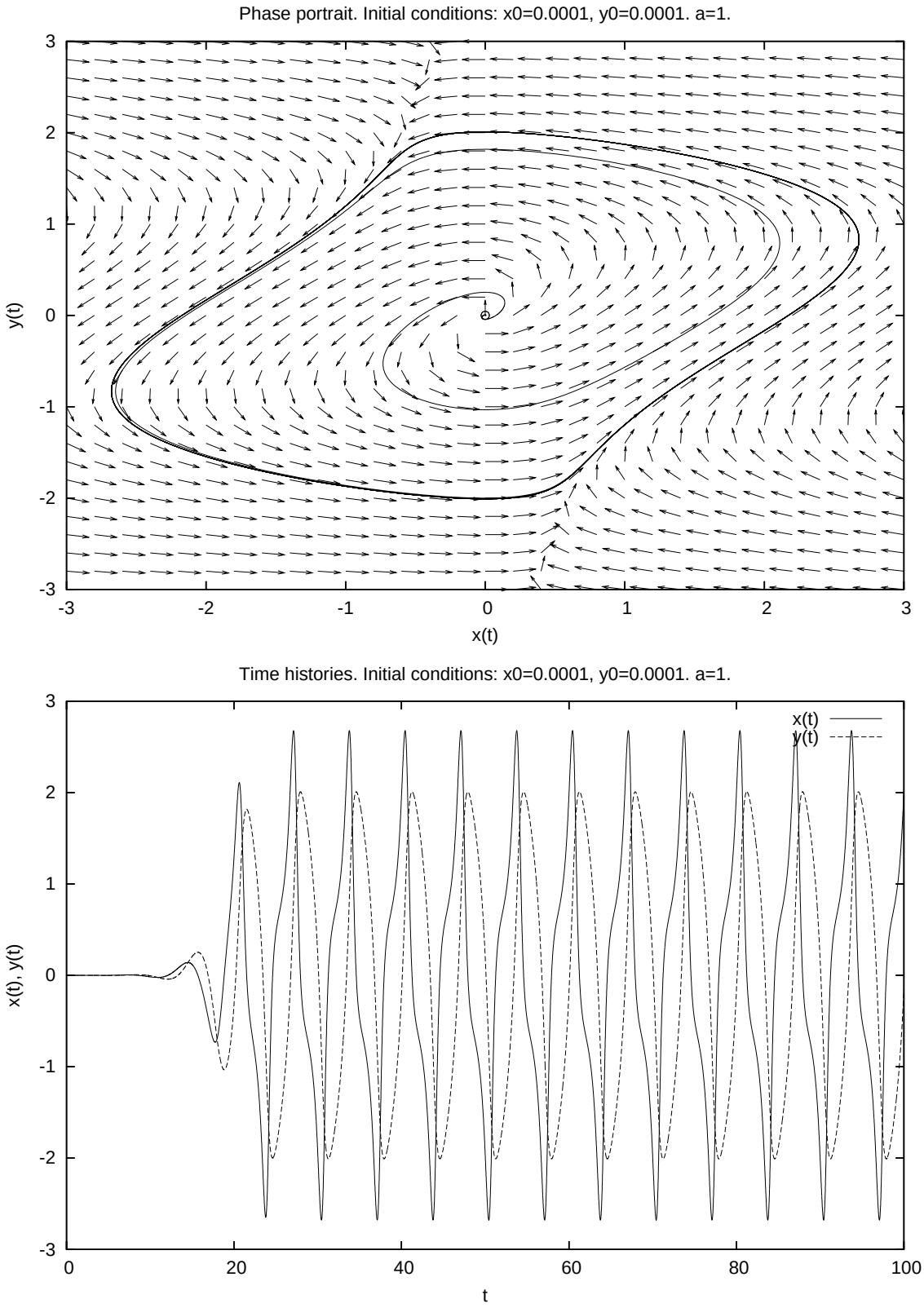

Figura 12: Ritratto di fase e storia temporale per $a = 1$, $(x_0, y_0) = (10^{-4}, 10^{-4})$ e $t_f = 100$.

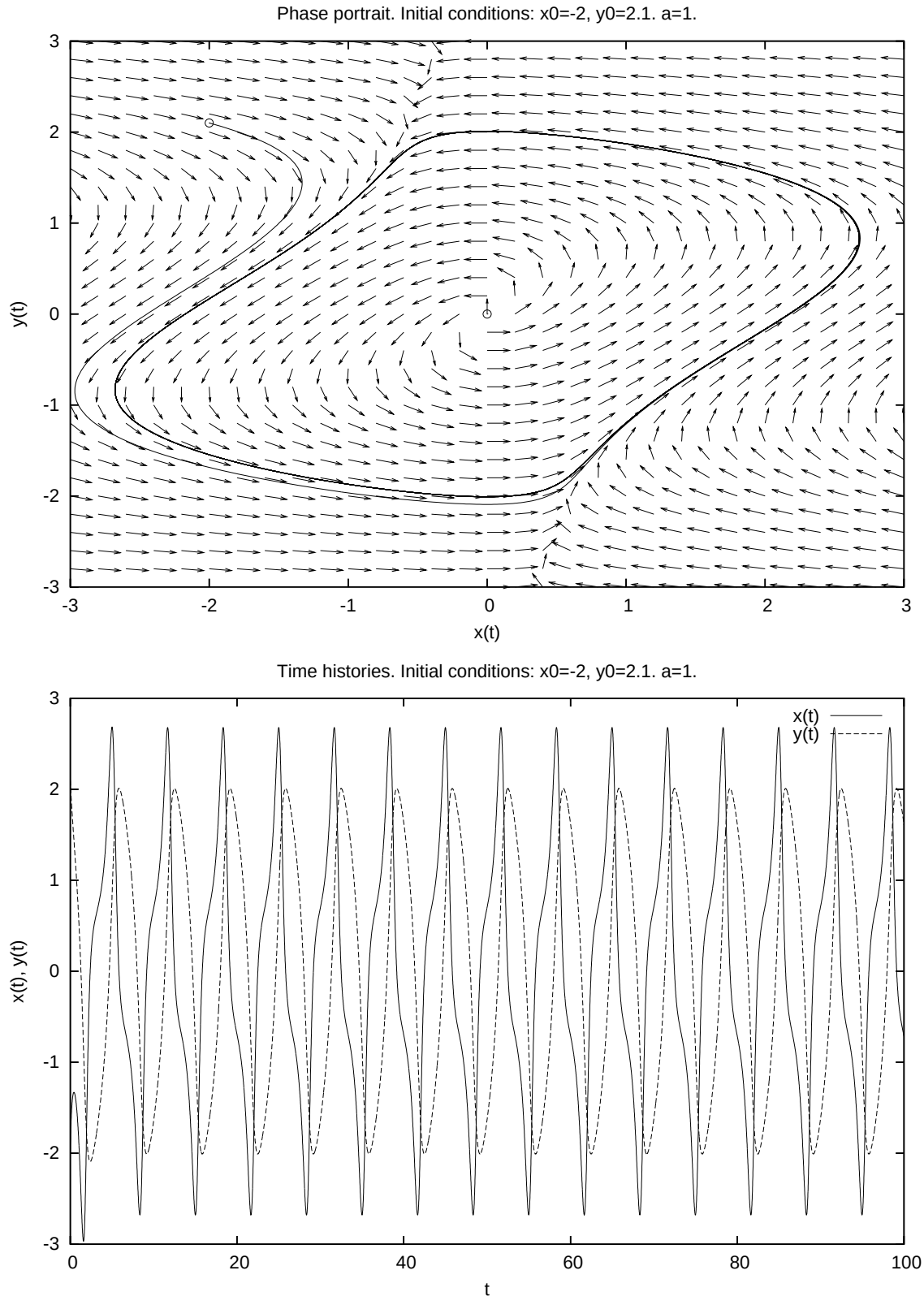

Figura 13: Ritratto di fase e storia temporale per $a = 1$, $(x_0, y_0) = (-2, 2.1)$ e $t_f = 100$.

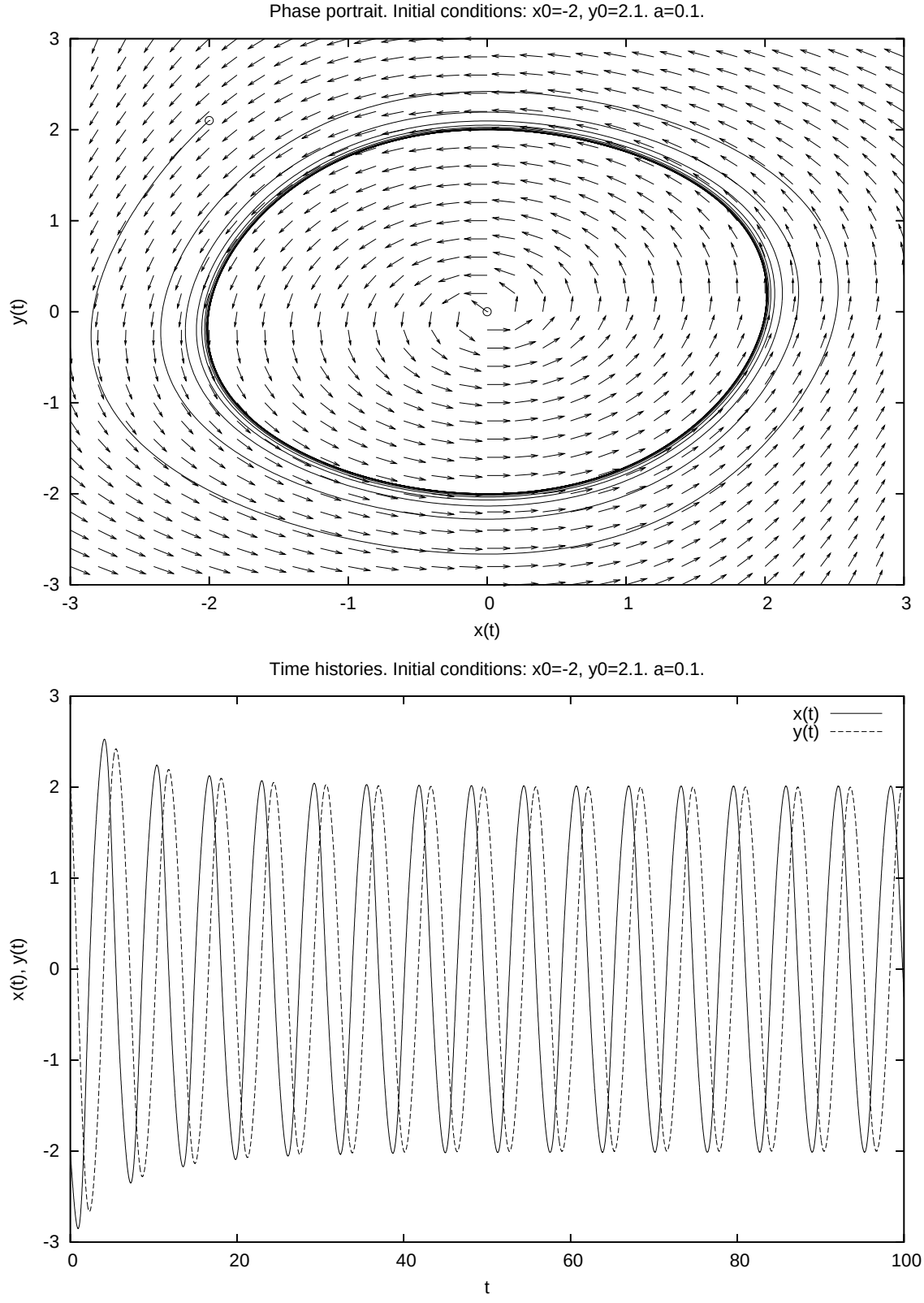

Figura 14: Ritratto di fase e storia temporale per $a = 0.1$, $(x_0, y_0) = (-2, 2.1)$ e $t_f = 100$.

1.5 Esercizio

Studiare, al variare delle condizioni iniziali e del tempo finale, il comportamento del sistema

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

1.5.1 Risoluzione

L'unico punto di equilibrio per il sistema dato è l'origine. Per capirne la natura, basta studiare, di nuovo, l'equazione differenziale ordinaria della distanza al quadrato dall'origine ($u = x^2 + y^2$) ricavata moltiplicando la prima equazione per x, la seconda per y, e sommandole. Così facendo si ottiene $u' = 2y^2(1 - u)$. Quindi, l'unica orbita per la quale $u' = 0$ è la circonferenza unitaria con centro nell'origine. La distanza dall'origine aumenta per condizioni iniziali interne a tale circonferenza e diminuisce per condizioni iniziali esterne. Pertanto, la circonferenza unitaria è un ciclo stabile e l'origine un fuoco instabile.

In file `sys5.m` può essere utilizzato per lo studio del sistema dato. Le figure 15 e 16 mostrano le caratteristiche sopra descritte.

```
% Name:     sys5.m
% Author:   Simone Zuccher
% Created:  03 May 2007
% Purpose:  Solve the ODE system (Van der Pool equation)
%           x'=y
%           y'=(1 - x^2 - y^2)y - x
%           given the initial condition x0 and y0
% Input:    Initial condition [x0 y0], final time tfinal
% Output:   1. plot of x(t) versus y(t) together with the vector field
%           2. plot time histories of x(t) and y(t)
% Modified:
%
% The equilibrium points are the following:
%
% [x = 0, y = 0],
%

% Clear all variables
clear all;

% Window ranges
xmin=-2;
xmax=2;
ymin=-2;
ymax=2;

% Equilibrium points
eq = [0 0];
```

```
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)
  u = x(1);
  v = x(2);
  xdot(1) = v;
  xdot(2) = (1-u^2-v^2)*v - u;
endfunction

__gnuplot_set__ nokey
axis([xmin xmax ymin ymax]);

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

DX = Y;
DY = (1-X.^2 - Y.^2).*Y - X;
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);')
```

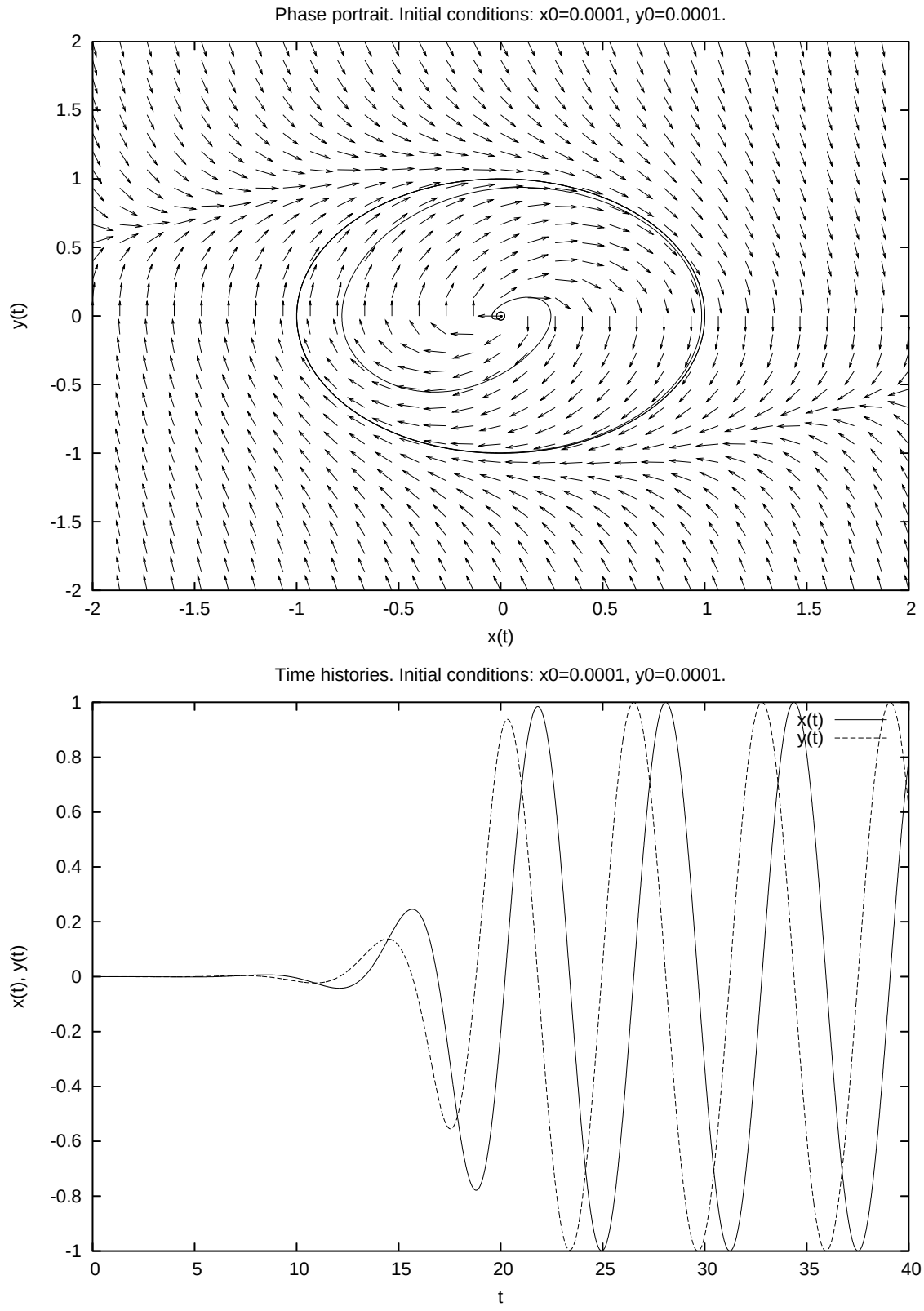

Figura 15: Ritratto di fase e storia temporale per $(x_0, y_0) = (10^{-4}, 10^{-4})$ e $t_f = 40$.

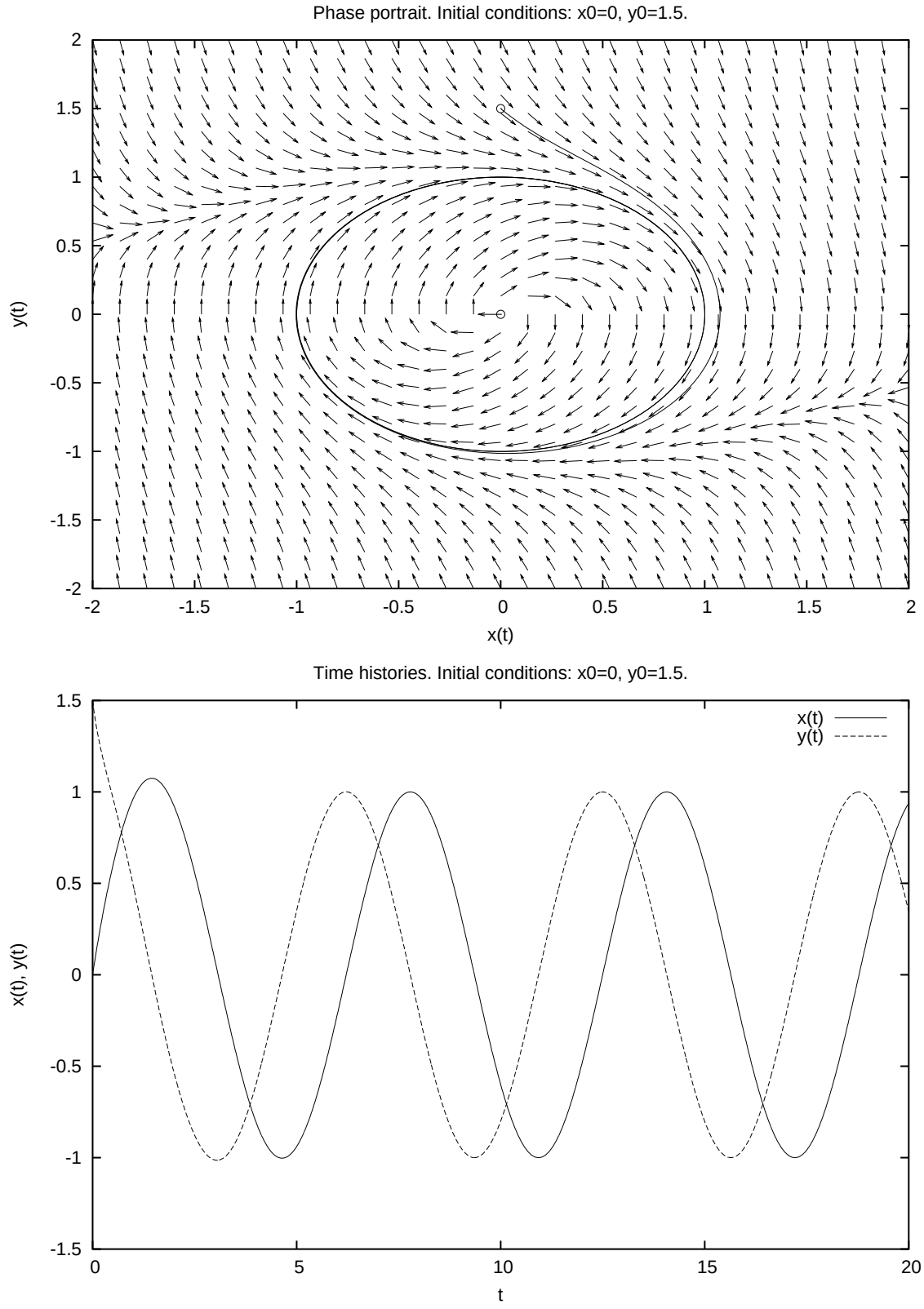

Figura 16: Ritratto di fase e storia temporale per $(x_0, y_0) = (0, 1.5)$ e $t_f = 20$.

1.6 Esercizio

Utilizzando le competenze acquisite in questa esercitazione e modificando adeguatamente gli script per Octave, studiare, al variare delle condizioni iniziali e del tempo finale, il comportamento dei sistemi

$$\begin{cases} x' &= x - y - x^3 \\ y' &= x - x^2 y \end{cases}$$

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