

Capitolo 2

Simulazione di sistemi dinamici

L'analisi dei sistemi non lineari presenta analogie e differenze con quella dei sistemi lineari. Le similitudini derivano dal fatto che una delle tecniche principali di analisi dei sistemi non lineari consiste nella loro approssimazione per mezzo di un sistema lineare e quindi nell'applicazione di metodologie relative a questi ultimi. Le differenze risiedono nel fatto che i sistemi non lineari possono presentare comportamenti completamente nuovi. L'analisi è diversa dal momento che le soluzioni esplicite sono raramente disponibili e quindi devono essere utilizzati metodi particolari per identificare le caratteristiche di comportamento.

2.1 Funzioni e modelli usati nel capitolo

In questo capitolo verranno utilizzati i seguenti files *Matlab*:

`IntegraCircuito.m`, integrazione di un modello differenziale.

`LinNonlinCompare.m`, confronto di un sistema col modello linearizzato.

`LinNonlinCompare long.m`, versione estesa di `LinNonlinCompare.m`.

`PassoBase.m`, confronto dei diversi sistemi di integrazione.

`circuito nl impulso.m`, integrazione di circuito non lineare con ingresso non nullo.

`circuito non lineare.m`, integrazione di circuito non lineare e generazione di grafici per diverse condizioni iniziali.

`tunnel1.m`, implementazione del circuito con diodo tunnel.

`tunnel2.m`, implementazione del circuito con diodo tunnel e ingresso non nullo.

2.2 Analisi di un circuito non lineare.

Si consideri il circuito rappresentato nella Figura 2.1 nel quale l'elemento non lineare T è caratterizzato da una relazione tensione corrente del tipo $i = -G_1 * v + G_2 * v^3$ essendo G_1 e G_2 due costanti a valori positivi.

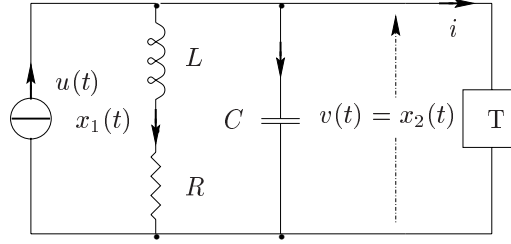


Figura 2.1: Circuito non lineare.

Assumendo come variabile di ingresso la corrente $u(t)$ erogata dal generatore, come uscita la tensione $v(t)$ ai capi dell'elemento non lineare e come variabili di stato la corrente $x_1(t)$ nell'induttore L e la tensione $x_2(t)$ sul condensatore C , il modello del sistema nello spazio degli stati è del tipo

$$\begin{aligned}\dot{x}_1(t) &= -\frac{R}{L} x_1(t) + \frac{1}{L} x_2(t) \\ \dot{x}_2(t) &= -\frac{1}{C} x_1(t) + \frac{1}{C} (G_1 x_2(t) - G_2 x_2^3(t)) + \frac{1}{C} u(t)\end{aligned}\quad (2.1)$$

Con ingresso nullo, $u(t) = 0$, gli stati di equilibrio del sistema si ottengono come soluzioni delle equazioni $x_2 = R x_1$ e $x_1 = G_1 x_2 - G_2 x_2^3$, ovvero

$$\bar{x}' = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad \bar{x}'' = \begin{bmatrix} \sqrt{\frac{G_1 R - 1}{G_2 R^3}} \\ \sqrt{\frac{G_1 R - 1}{G_2 R}} \end{bmatrix} \quad \text{e} \quad \bar{x}''' = \begin{bmatrix} -\sqrt{\frac{G_1 R - 1}{G_2 R^3}} \\ -\sqrt{\frac{G_1 R - 1}{G_2 R}} \end{bmatrix}. \quad (2.2)$$

Si noti nelle ipotesi che $G_1 * R - 1 < 0$ soltanto l'origine dello spazio degli stati è punto di equilibrio del circuito.

Il circuito è stato simulato con i seguenti valori dei parametri

$$G_1 = 0.8, \quad G_2 = 0.05, \quad R = 2, \quad L = 1 \quad \text{e} \quad C = 1$$

in assenza di ingresso e a partire da diverse condizioni iniziali. Con le seguenti funzioni, `tunnel1.m`

```
function xd = tunnel1(t,x,flag,param)

% Funzione che implementa il circuito Di Eq.(2.1) con u(t) = 0.
%
% d x1(t) / dt = - R/L x1(t) + 1/L x2(t)
```

```

% d x2(t) / dt = - 1/C x1(t) + 1/C ( G1 x2(t) - G2 x2(t)^3 ) + 1/C u(t)
%

R = param(1);
L = param(2);
C = param(3) ;
G1 = param(4);
G2 = param(5);

x1d = - (R/L) * x(1) + (1.0/L) * x(2);
x2d = - (1.0/C) * x(1) + (1.0/C)*( G1 * x(2) - G2 * x(2)^3 );

xd = [x1d; x2d];

return

per la realizzazione del sistema di equazioni differenziali e IntegraCircuito.m

% Script-file che integra il sistema differenziale (2.1)
% e grafica i risultati.

options = odeset('RelTol',1e-6); % Opzioni per la funzione di integrazione

R = 2; % Parametri fisici del modello
L = 1;
C = 1;
G1 = 0.8;
G2 = 0.05;

param = [R,L,C,G1,G2]; % Parametri fisici del modello

ci = [2 2]; % Condizioni iniziali per l'integrazione

time = [0 40]; % Tempo di integrazione

[t,x] = ode45('tunnel1',time,ci,options,param); % Funzione che effettua l'integrazione

%% Grafico delle traiettorie dello stato

figure
plot(x(:,1),x(:,2),'-') % Disegna i vettori passati come argomenti
title('Traiettorie dello stato') % Titolo del grafico
xlabel('x1') % Etichetta dell'asse delle ascisse
ylabel('x2') % Etichetta dell'asse delle ordinate

%% Grafico del moto dello stato

figure
plot(t,x(:,1),'-',t,x(:,2),'--')
```

```

title('Andamento nel tempo')
xlabel('Tempo')
ylabel('x1(t) e x2(t)')

```

per l'integrazione del sistema e la visualizzazione dei risultati, si ottengono i seguenti grafici rappresentati nelle Figure 2.2 e 2.3. La Figura 2.3 è stata ottenuta partendo dallo stato iniziale $x_1(0) = 2$ e $x_2(0) = 2$.

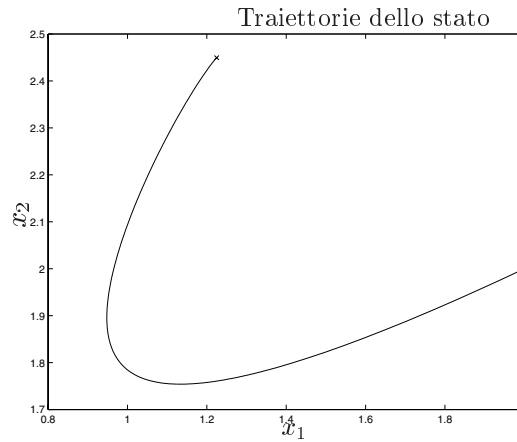


Figura 2.2: Traiettorie dello stato.

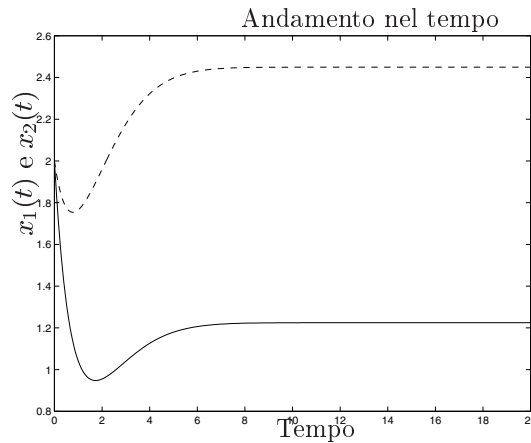


Figura 2.3: Andamento nel tempo delle variabili di stato.

L'istruzione di *Matlab* `ode45` consente di integrare sistemi di equazioni e verrà descritta nel paragrafo 2.3.

Il comando `plot(x,y,'z')` di *Matlab*, con x, y vettori riga o colonna costituiti da elementi reali, produce un grafico con le ascisse costituite dagli elementi del vettore x e con le ordinate costituite dagli elementi del vettore y . L'ulteriore argomento `'z'` impone l'impiego di un determinato stile di linea per la visualizzazione del grafico. Le istruzioni `title('text')`, `xlabel('text')` e

`ylabel('text')` inseriscono la stringa 'text', rispettivamente, come titolo del grafico, come etichetta dell'asse delle ordinate e delle ascisse.

La figura 2.4 rappresenta le traiettorie percorse dalle variabili di stato del sistema, nel caso in cui $G_1 R - 1 > 0$, per diversi valori dello stato iniziale. La retta tratteggiata ha equazione $x_2 = R x_1$ mentre la curva tratteggiata ha equazione $x_1 = G_1 x_2 - G_2 x_2^3$. Appare evidente il comportamento stabile del sistema nell'intorno dei due punti di equilibrio diversi da zero.

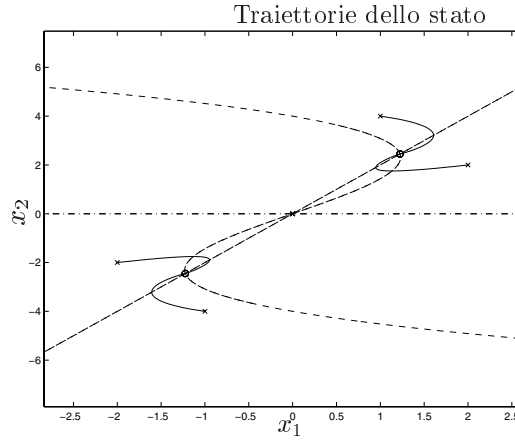


Figura 2.4: Traiettorie dello stato.

Nella situazione per cui $G_1 * R - 1 < 0$, ad esempio ponendo $G_1 = 0.8$ ed $R = 1$, l'origine è l'unico punto di equilibrio per il sistema. La Figura 2.5 mostra l'andamento nel tempo delle traiettorie a partire da diversi stati iniziali, mentre in Figura 2.6 è riportato l'andamento smorzato delle due variabili di stato a partire dalla condizione $x_1(0) = 3$ e $x_2(0) = -3$.

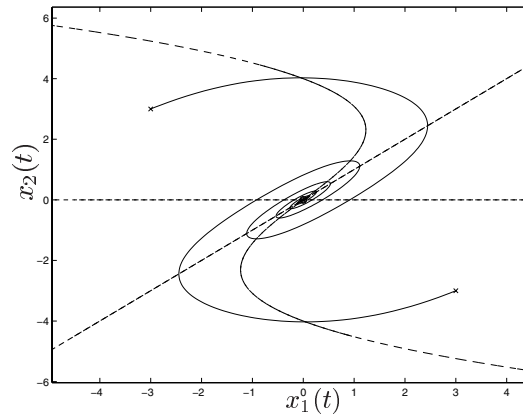


Figura 2.5: Traiettorie dello stato.

Sempre nelle condizioni in cui, $G_1 * R - 1 < 0$, ad esempio ponendo $G_1 = 0.8$ ed $R = 0.5$, l'origine rimane l'unico punto di equilibrio per il sistema. La Figura 2.7

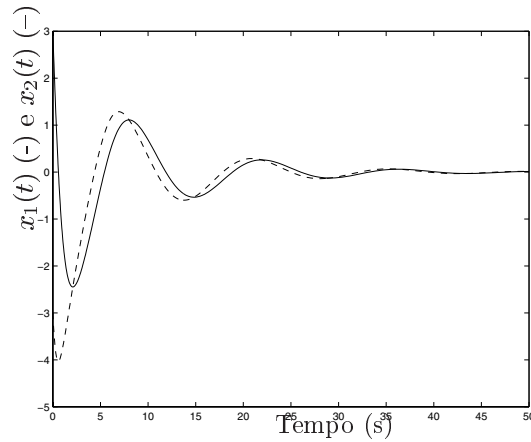


Figura 2.6: Andamento delle variabili di stato.

mostra l'andamento nel tempo delle traiettorie a partire da diversi stati iniziali, mentre in Figura 2.8, è riportato l'andamento oscillatorio delle due variabili di stato a partire dalla condizione $x_1(0) = 1$ e $x_2(0) = 1$.

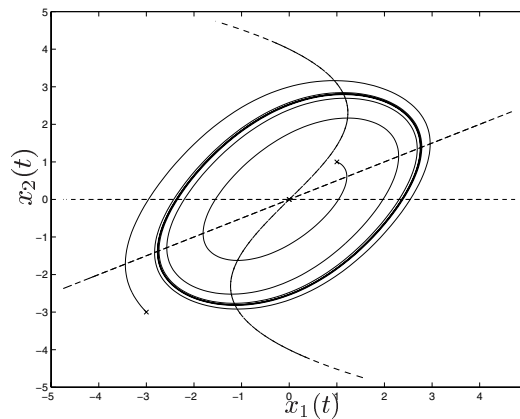


Figura 2.7: Traiettorie dello stato con un punto di equilibrio.

Volendo analizzare la risposta del circuito ad un impulso di corrente di ampiezza pari a $u(t) = 8$ e di durata di 10s. ($u(t) = 0$ per $t > 10$), la funzione `tunnel1.m` viene modificata nel modo seguente (`tunnel2.m`)

```
function xd = tunnel2(t,x,flag,param)
% Funzione che implementa il circuito di Eq. (2.1). E' presente
% anche una funzione dell'ingresso udt funzione del tempo.
%
% d x1(t) / dt = - R/L x1(t) + 1/L x2(t)
% d x2(t) / dt = - 1/C x1(t) + 1/C ( G1 x2(t) - G2 x2(t)^3 ) + 1/C u(t)
%
```

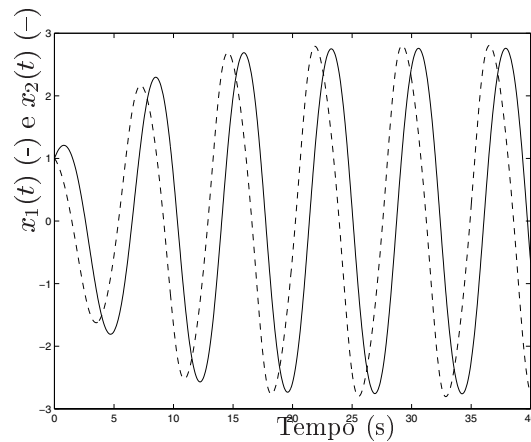


Figura 2.8: Andamento oscillatorio delle variabili di stato.

```

R = param(1); % Parametri del circuito non lineare
L = param(2);
C = param(3);
G1 = param(4);
G2 = param(5);
Tstart = param(6); % Istante di inizio del gradino
Tstop = param(7); % Istante finale del gradino
Value = param(8); % Ampiezza del gradino

if((t>=Tstart)&(t<Tstop)),udt=Value; % Definizione del gradino
    else udt=0.0;
end;

x1d = - (R/L) * x(1) + (1.0/L) * x(2);
x2d = - (1.0/C) * x(1) + (1.0/C)*( G1 * x(2) - G2 * x(2)^3 );

xd = [x1d; x2d + (1.0/C)*udt];

return

```

e il programma di simulazione fornisce a partire dallo stato zero i seguenti risultati riportati nelle Figure 2.9 e 2.10.

Si voglia ora linearizzare il circuito caratterizzato dagli stessi parametri elettrici dell'esempio precedente, con ingresso nullo e nell'intorno dell'origine dello spazio degli stati. Il modello linearizzato assume la struttura

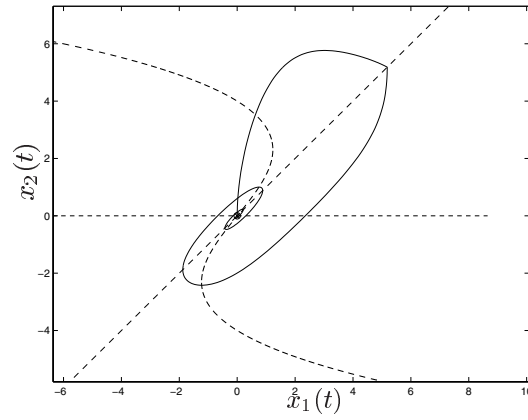


Figura 2.9: Traiettorie dello stato con un punto di equilibrio e impulso di corrente.

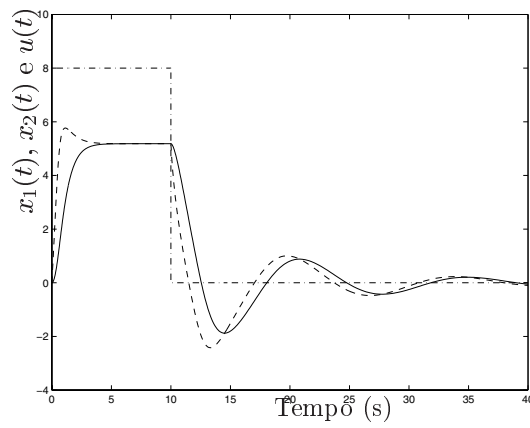


Figura 2.10: Andamento smorzato delle variabili di stato con impulso di corrente.

$$\begin{aligned}\delta \dot{x}(t) &= A\delta x(t) + B\delta u(t) \\ \delta y(t) &= C\delta x(t)\end{aligned}\tag{2.3}$$

avendo posto

$$\begin{aligned}\dot{x}_1(t) &= f_1(x(t), u(t)) \\ \dot{x}_2(t) &= f_2(x(t), u(t)) \\ y(t) &= g(t)\end{aligned}$$

dove $\delta x(t)$, $\delta u(t)$ e $\delta y(t)$ rappresentano, rispettivamente, gli scostamenti dello

stato, dell'ingresso e dell'uscita dai valori di equilibrio e le matrici

$$A = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{bmatrix}, B = \begin{bmatrix} \frac{\partial f_1}{\partial \mu} \\ \frac{\partial f_2}{\partial u} \end{bmatrix} \text{ e } C = \begin{bmatrix} \frac{\partial g}{\partial x_1} & \frac{\partial g}{\partial x_2} \end{bmatrix},$$

vanno calcolate in corrispondenza del punto di equilibrio scelto.

Nel caso in esame risulta

$$A = \begin{bmatrix} -\frac{R}{L} & \frac{1}{L} \\ -\frac{1}{C} & \frac{G_1}{C} \end{bmatrix}, B = \begin{bmatrix} 0 \\ \frac{1}{C} \end{bmatrix} \text{ e } C = \begin{bmatrix} 0 & 1 \end{bmatrix}$$

Il programma seguente `LinNonlinCompare.m` mette a confronto la simulazione del circuito, in assenza di ingresso e a partire da assegnate condizioni iniziali, utilizzando il modello non lineare e quello linearizzato.

```
%%
%% Script-file per il confronto della risposta del sistema lineare
%% e quella del sistema non lineare.
%%

%%
%% Modello non lineare
%%

options = odeset('RelTol',1e-6);

R = 1; % Parametri del circuito non lineare.
L = 1;
C = 1;
G1 = 0.8;
G2 = 0.05;

param = [R,L,C,G1,G2]; % Parametri del circuito non lineare.

ci = [0.5 0.5]; % Condizioni iniziali

ti = 0;
tf = 40;

time = [ti tf]; % Istante iniziale e finale di integrazione

[t,x] = ode45('tunnel1',time,ci,options,param); % Integrazione del sistema

y = x(:,2); % Uscita del sistema

%%
%% Modello lineare nello spazio degli stati
%%
```

```

A = [-R/L    1.0/L;
      -1.0/C  G1/C ];

B = [1.0/C 0]';

C = [0 1];

D = 0;

Sys = ss(A,B,C,D); % Crea il modello nello spazio degli stati

tl = ti:0.01:tf; % Tempo di simulazione
Ul = zeros(size(tl)); % Ingresso del sistema lineare
[y1,tl,x1] = lsim(Sys,Ul,tl,ci);
                % Simula la risposta nel tempo
                % per un sistema lineare tempo-invariante

figure
plot(tl,y1,'-'), hold on
plot(t,y,'--')
title('Risposte')
xlabel('Tempo')
ylabel('y')

figure
plot(x(:,1),x(:,2),'-'), hold on
plot(x1(:,1),x1(:,2),'-'), hold on
title('Traiettorie dello stato')
xlabel('x1')
ylabel('x2')

```

La Figura 2.11 riporta l'andamento temporale dell'uscita, calcolata sia con il modello lineare che con il modello linearizzato a partire dallo stato iniziale $x(0) = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$, vicino all'origine. La linea continua rappresenta la risposta del sistema lineare, mentre la curva tratteggiata, quella del sistema non lineare.

La Figura 2.12 riporta lo stesso grafico a partire da un nuovo stato iniziale $x(0) = \begin{bmatrix} -2 \\ 2 \end{bmatrix}$, lontano dall'origine.

Le Figure 2.13 e 2.14 riportano le traiettorie dello stato calcolate nelle stesse condizioni.

La linea continua rappresenta la traiettoria dello stato del sistema lineare, mentre la curva punteggiata, quella del sistema non lineare.

Si noti che mentre ci si allontana al punto rispetto al quale si è effettuata la linearizzazione, i modelli forniscono risposte significativamente diverse.

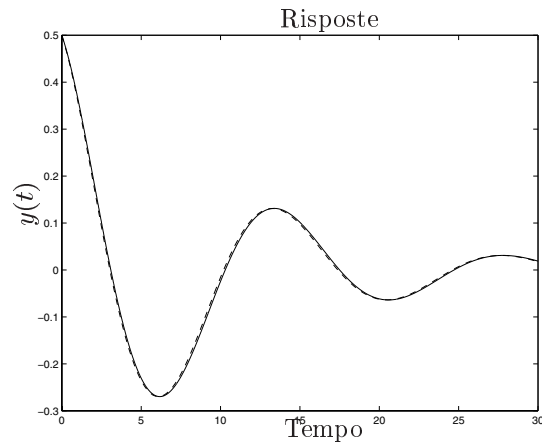


Figura 2.11: Risposta del sistema lineare e del sistema non lineare.

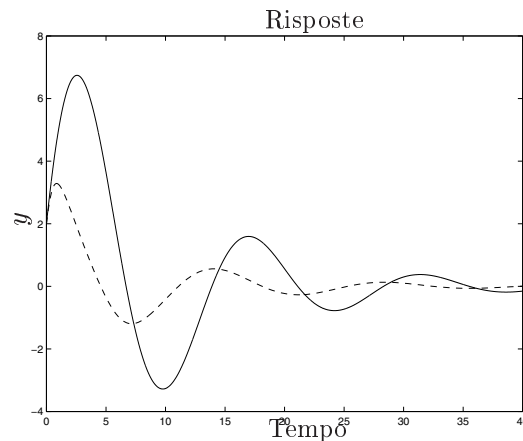


Figura 2.12: Risposta dei sistemi per una condizione iniziale distante dall'origine.

2.3 Metodi numerici per integrazione di equazioni differenziali in Matlab.

La simulazione di sistemi in *Matlab* generalmente richiede l'integrazione di sistemi di equazioni differenziali ordinarie. *Matlab* fornisce un insieme di funzioni per risolvere numericamente queste funzioni differenziali. I risultati in termini di velocità ed accuratezza dipendono dal tipo di modelli e di condizioni. Sono riportati nel seguito alcune caratteristiche dei metodi di integrazione.

1. **ode15s**: risolve equazioni differenziali ordinarie per sistemi stiff autonomi e non (generalmente sistemi non lineari e “smooth”) implementando un metodo di integrazione numerica di ordine variabile fino al 5° con un passo di integrazione quasi fisso. Riesce a risolvere efficientemente anche problemi relativi a sistemi non stiff.

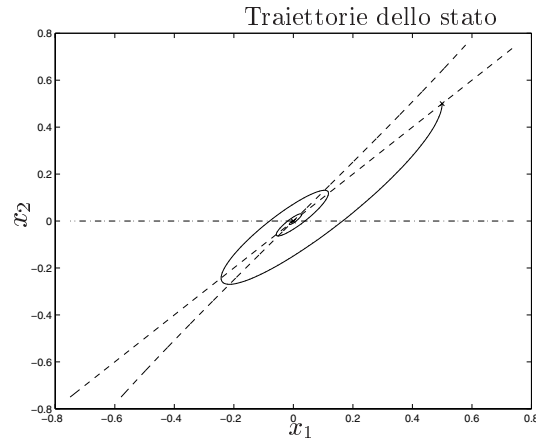


Figura 2.13: Traiettorie dello stato per condizioni iniziali vicino all'origine.

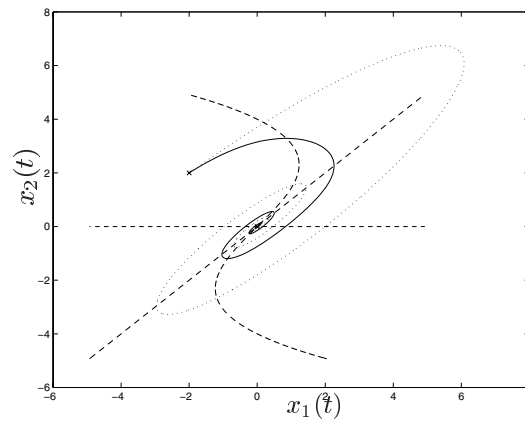


Figura 2.14: Traiettorie dello stato per condizioni iniziali lontane all'origine.

2. `ode23s`: risolve equazioni differenziali per sistemi stiff utilizzando metodi di ordine non elevato.
3. `ode23`: questo programma è un'alternativa a `ode15s` per la soluzione di problemi stiff ed implementa un metodo di Runge-Kutta del 2° ordine fisso. Permette di ottenere soluzioni con tolleranze non troppo spinte ed è indicato per sistemi con autovalori vicini all'asse immaginario, quindi sistemi con memoria a dinamica lenta. Riesce comunque a gestire sistemi con dinamiche veloci, anche non autonomi.
4. `ode45`: come `ode23` implementa il metodo di Runge-Kutta ed è quindi adatta per sistemi fortemente non lineari. Questa funzione non è consigliabile per sistemi che manifestino dinamiche lente e veloci contemporaneamente.
5. `ode113`: è stata introdotta per superare i problemi che presentano le fun-

zioni `ode23` e `ode45`. Permette di ottenere una accuratezza sia moderata che elevata pur mantenendo una complessità accettabile. Utilizzando formule risolutive di ordine elevato fornisce soluzioni con una risoluzione adeguata anche per applicazioni grafiche.

Le funzioni per l'integrazione delle equazioni differenziali ordinarie richiedono tutte la stessa sintassi. Nell'esempio seguente verrà perciò utilizzata la generica funzione `ode45`. Nel paragrafo 2.2, si è utilizzato il comando

```
[t,x] = ode45('tunnel1',[ti tf],x0,options);
```

in cui `'tunnel1'` è l'equazione differenziale da integrare, espressa attraverso un *function-file* di *Matlab*, `[ti tf]` rappresenta l'intervallo di integrazione, a partire dalle condizioni iniziali `x0`. La funzione prevede la possibilità di utilizzare un insieme di opzioni.

```
options = odeset('RelTol',1e-4,'AbsTol',[1e-4 1e-4],'Maxstep',10);
```

Si guardi in proposito l'help in linea per la funzione `'odeset'`. Il vettore `'options'` viene costruito attraverso la suddetta funzione che accetta coppie valore e funzione chiave (e.g. `'RelTol'` e `1e-4`).

Le opzioni più utilizzate sono appunto `'RelTol'` e `'AbsTol'`, le tolleranze relativa ed assoluta associate all'errore per il controllo della convergenza. Per la funzione `ode45`, l'errore ε_i sull' i -esima componente relativa alla stima della soluzione y_i dell'equazione differenziale, soddisfa la disequazione

$$|\varepsilon_i| \leq r|y_i| + a_i \quad (2.4)$$

in cui $r = \text{RelTol}$ e $a_i = \text{AbsTol}(i)$. L'errore relativo scalare ha il valore di default di 10^{-3} , mentre quello assoluto di 10^{-6} . Un ultimo parametro che si può utilizzare è la massima ampiezza del passo di integrazione, per garantire che il metodo di integrazione utilizzato riconosca fenomeni che avvengono in quell'intervallo di tempo prefissato. Viene indicato con `'MaxStep'` e il valore di default coincide con $1/10$ dell'intervallo di integrazione.

2.4 Problematiche relative all'integrazione di sistemi dinamici.

Si riprenda l'esempio precedente relativo all'integrazione del sistema implementato attraverso la funzione `tunnel1.m` e si utilizzi `ode45` con tolleranze e passi di integrazione diversi, come appare nel listato del programma `PassoBase.m`

```
%%
%% Script-file per il confronto delle risposta del sistema non lineare
%% utilizzando diversi metodi di integrazione.
%%

R = 1; % Parametri del circuito non lineare.
L = 1;
C = 1;
```

```

G1 = 0.8;
G2 = 0.05;

param = [R,L,C,G1,G2];

ci = [0.5 0.5];

ti = 0;
tf = 40;

time = [ti tf];

options = odeset('RelTol',1.0,'AbsTol',[1e-1 1e-1],'MaxStep',1000);
[t1,x1] = ode45('tunnel1',time,ci,options,param);
y1 = x1(:,2);

options = odeset('RelTol',1e-1,'AbsTol',[1e-2 1e-2],'MaxStep',100);
[t2,x2] = ode45('tunnel1',time,ci,options,param);
y2 = x2(:,2);

options = odeset('RelTol',1e-3,'AbsTol',[1e-6 1e-6],'MaxStep',10);
[t3,x3] = ode45('tunnel1',time,ci,options,param);
y3 = x3(:,2);

figure
plot(t1,y1,'-'), hold on
plot(t2,y2,'--'), hold on
plot(t3,y3,'-.')
title('Risposte')
xlabel('Tempo')
ylabel('y')

return

```

I risultati di tale simulazione sono riportati nella Figura 2.15. Si noti che i risultati ottenuti sono diversi per i diversi valori assegnati alle tolleranze e ai passi di integrazione. Più piccola è la tolleranza, più passi di integrazione sono richiesti dal metodo utilizzato, che porta a risultati più accurati.

Analogamente, risultati diversi si ottengono integrando le stesse equazioni differenziali con tecniche differenti. Per una descrizione più dettagliata di queste tecniche e per un utilizzo più appropriato si rimanda all'help in linea di *Matlab* e alla consultazione dell'articolo [5].

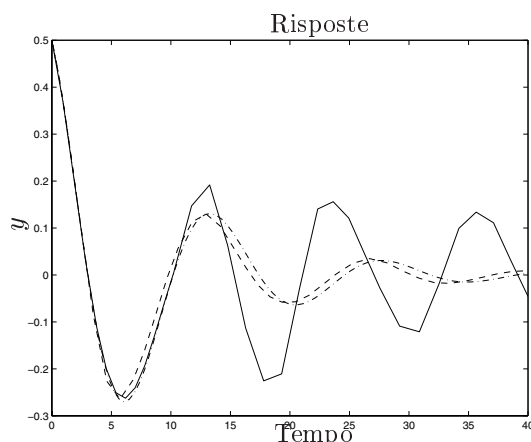


Figura 2.15: Risposta del sistema con diversi passi di integrazione.

2.5 Esercizi proposti in aula didattica.

1. Si consideri il modello matematico di Lotka-Volterra (2.5) che descrive la dinamica di due popolazioni interagenti

$$\begin{aligned} \dot{x}_1(t) &= a_1(1 - x_1(t)/k)x_1(t) - a_2x_1(t)x_2(t) + u(t) \text{ (prede)} \\ \dot{x}_2(t) &= -a_3x_2(t) + a_4x_1(t)x_2(t) \text{ (predatori)} \end{aligned} \quad (2.5)$$

dove $x_1(t)$ e $x_2(t)$ rappresentano rispettivamente il numero di prede e il numero di predatori presenti nell'ecosistema all'istante t ed $u(t)$ l'apporto esterno di cibo per le prede introdotto nell'unità di tempo. Il coefficiente k rappresenta il numero massimo di prede presenti nell'ecosistema in assenza di predatori e in assenza di apporto di cibo esterno ($u(t) = 0$). Il parametro a_3 (> 0) è il tasso di crescita del predatore, in assenza di prede, mentre a_1 (> 0) il tasso di crescita delle prede. Il termine $-a_2x_1(t)x_2(t)$ modella il decremento nella popolazione delle prede per la presenza dei predatori, con $a_2 > 0$, mentre il termine $a_4x_1(t)x_2(t)$ rappresenta l'incremento della popolazione dei predatori dovuto alla presenza delle prede.

Nelle ipotesi di assegnare ai parametri i valori $a_1 = 20$, $a_2 = 1$, $a_3 = 7$, $a_4 = 0.5$ e $k = 30$, si determinino

- (a) l'andamento nel tempo del numero di prede e predatori, supponendo nullo l'ingresso $u(t)$ e nelle ipotesi di partire da un ecosistema contenente 10 prede e 10 predatori. Si calcoli anche la traiettoria percorsa dal sistema nello spazio degli stati.
- (b) gli stati di equilibrio del sistema in assenza di ingresso.
- (c) i valori di regime raggiunti dal numero di prede e predatori nelle ipotesi che $u(t)$ sia un gradino di ampiezza $u(t) = 20$ e a partire dalle stesse condizioni proposte al punto 1). Si determini per tentativi l'ampiezza del gradino che consente di mantenere a regime un numero di predatori pari a 15.

2. (Facoltativo) Si definisce *modello ibrido* un sistema composto da diversi modelli, ciascuno valido in una particolare condizione di funzionamento del sistema stesso. Tale condizione dipende dallo stato del sistema e dai suoi ingressi.

Si consideri quindi il sistema ibrido descritto dal seguente modello matematico:

$$\dot{\mathbf{x}}(t) = \begin{cases} \mathbf{A}_1 \mathbf{x}(t) & \text{se } x_1(t) * x_2(t) < 0 \\ \mathbf{A}_2 \mathbf{x}(t) & \text{se } x_1(t) * x_2(t) \geq 0 \end{cases},$$

in cui

$$\mathbf{x}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

e

$$\mathbf{A}_1 = \begin{bmatrix} -0.1 & 1.0 \\ -10.0 & -0.1 \end{bmatrix}, \quad \mathbf{A}_2 = \begin{bmatrix} -0.1 & 10 \\ -1.0 & -0.1 \end{bmatrix}.$$

- (a) Si discuta la stabilità dei sistemi singoli.
- (b) Si disegnino le traiettorie dello stato per i sistemi singoli e per il sistema completo considerando un tempo di simulazione di 10s e le condizioni iniziali $(0, 1)$, $(1, 0)$ e $(10^{-6}, 10^{-6})$.
- (c) Si disegni l'andamento nel tempo delle variabili di stato dei sistemi singoli e di quello ibrido nelle stesse condizioni.

Bibliografia

- [1] K. Sigmon, *MATLAB Primer*. University of Florida, Florida, Second Edition ed., 1992. (Si scarica dalla rete).
- [2] The MathWorks, Inc., *Matlab, The Language of Technical Computing. Getting Started with MATLAB*, version 5.1 ed., May 1997. (In formato pdf su CD Matlab).
- [3] The MathWorks Inc., *Matlab User's Guide*, 1993.
- [4] G. F. Franklin, J. D. Powell, and M. Workman, *Digital Control of Dynamic Systems*. Addison-Wesley, Third Edition ed., 1998.
- [5] L. F. Shampine and M. W. Reichel, "The Matlab Ode Suite," tech. rep., The MathWorks, Inc, 1997. (Disponibile anche come file in formato pdf).