Corso di
Gasdinamica II

Tommaso Astarita
astarita@unina.it
www.docenti.unina.it

Metodo di Eulero

Si supponga di avere una equazione differenziale del primo ordine:

$$\begin{cases} y' = f(x,y) \\ x = x_o \quad y = y_o \end{cases}$$

Definendo $x_i = x_o + ih$

E di conseguenza: $y_i = y(x_i)$

Sviluppando in serie la funzione y si ha:

$$y(x_o + h) = y_o + hy'_o + \frac{h^2 y''_o}{2} + o(h^3)$$

Ovvero in generale:

$$y(x_i + h) = y_{i+1} = y_i + hy'_i + \frac{h^2 y''_i}{2} + o(h^3)$$

Arrestando l'espansione al primo ordine si trova la formula di Eulero:

$$y_{i+1} = y_i + hy'_i + o(h^2)$$

Che produce un errore di ordine h^2 per ogni passo d'integrazione.

Metodo di Eulero

$$y_{i+1} = y_i + hy_i' + o(h^2)$$

Supponendo che si voglia integrare l'equazione differenziale fra x_o=a e x=b si dovranno effettuare $n = (b-a)/h$ passi d'integrazione totali. Quindi l'errore alla fine dell'integrazione sarà dell'ordine di $h(b-a)$.

Metodo di Eulero Cauchy

Sviluppando in serie la funzione y si ha:

$$y_{i+1} = y_i + hy_i' + \frac{h^2 y_i''}{2} + \frac{h^3 y_i'''}{6} + o(h^4)$$

$$y_{i-1} = y_i - hy_i' + \frac{h^2 y_i''}{2} - \frac{h^3 y_i'''}{6} + o(h^4)$$

Sottraendo la seconda dalla prima si trova una formula simmetrica:

$$y_{i+1} = y_{i-1} + 2hy_i' + \frac{h^3 y_i'''}{3}$$

Che produce un errore di ordine h^3 per ogni passo d'integrazione.

Metodo di Runge Kutta

Si utilizza una formula del tipo:

$$y_{i+1} = y_i + \int_{x_i}^{x_{i+1}} f(x,y)dx$$

Dove si suppone che l'integrale possa essere valutato come:

$$y_{i+1} = y_i + \sum_{s=1}^{m} w_s k_s$$

Con:

$$k_1 = hf(x_i, y_i) \qquad k_s = hf\left(x_i + \alpha_s h, y_i + \sum_{j=1}^{s-1} \lambda_{s,j} k_j\right)$$

Le incognite sono:

- m pesi w_s
- (m-1) coefficienti α_s
- (m-1)m/2 coefficienti λ_{sj}

Il numero di incognite è tale che già per m=2 i passaggi risultano complicati.

Metodo di Runge Kutta

Per m=2 si ha:

$$y_{i+1} = y_i + w_1 k_1 + w_2 k_2$$

Con la simbologia abbreviata $f = f(x_i, y_i)$ si ha:

$$k_1 = hf$$

$$k_2 = hf(x_i + \alpha h, y_i + \lambda k_1)$$

Sviluppando in serie si ha:

$$k_2 = hf(x_i + \alpha h, y_i + \lambda k_1) = hf + \alpha h^2 f_x + \lambda k_1 h f_y = hf + \alpha h^2 f_x + \lambda h^2 f f_y$$

Chiaramente: $y' = f$

Mentre: $$y'' = \frac{\partial f}{\partial x} = f_x + f f_y$$

Quindi: $$y_{i+1} = y_i + hy_i' + \frac{h^2 y_i''}{2} + o(h^3) = y_i + hf + f_x \frac{h^2}{2} + ff_y \frac{h^2}{2} + o(h^3)$$

Metodo di Runge Kutta

$$y_{i+1} = y_i + hf + f_x \frac{h^2}{2} + ff_y \frac{h^2}{2} + o\left(h^3\right)$$

$$y_{i+1} = y_i + w_1 k_1 + w_2 k_2$$

$$k_1 = hf$$

$$k_2 = hf + \alpha h^2 f_x + \lambda h^2 ff_y$$

Uguagliando le due formule si ha:

$$y_i + w_1 k_1 + w_2 k_2 = y_i + hf + f_x \frac{h^2}{2} + ff_y \frac{h^2}{2}$$

$$y_i + w_1 hf + w_2 hf + w_2 \alpha h^2 f_x + w_2 \lambda h^2 ff_y = y_i + hf + f_x \frac{h^2}{2} + ff_y \frac{h^2}{2}$$

Confrontando i vari termini si ottengono le seguenti 3 equazioni nelle 4 incognite:

$$w_1 + w_2 = 1 \quad w_2 \alpha = \frac{1}{2} \quad w_2 \lambda = \frac{1}{2}$$

Si ha la possibilità di assegnare una condizione aggiuntiva, ad esempio:

$$w_1 = w_2$$

Metodo di Runge Kutta

$$w_1 + w_2 = 1 \quad w_2 \alpha = \frac{1}{2} \quad w_2 \lambda = \frac{1}{2} \qquad w_1 = w_2$$

Risolvendo il sistema si trova **una** formula di Runge Kutta del secondo ordine:

$$w_1 = w_2 = \frac{1}{2} \quad \alpha = 1 \quad \lambda = 1$$

$$y_{i+1} = y_i + w_1 k_1 + w_2 k_2 = y_i + \frac{1}{2}\left(k_1 + k_2\right)$$

$$k_1 = hf\left(x_i, y_i\right)$$

$$k_2 = hf\left(x_i + \alpha h, y_i + \lambda k_1\right) = hf\left(x_i + h, y_i + hf\right)$$

La formula derivata è normalmente chiamata formula di Heun.

Metodo di Runge Kutta

Il metodo di Runge Kutta più utilizzato è una formula del quarto ordine:

$$y_{i+1} = y_i + \frac{1}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right)$$

$$k_1 = hf\left(x_i, y_i\right)$$

$$k_2 = hf\left(x_i + \frac{h}{2}, y_i + \frac{k_1}{2}\right)$$

$$k_3 = hf\left(x_i + \frac{h}{2}, y_i + \frac{k_2}{2}\right)$$

$$k_4 = hf\left(x_i + h, y_i + k_3\right)$$

Soluzione di sistemi di equazioni del primo ordine

L'estensione delle formule precedenti per la soluzione di un sistema di equazioni differenziali del primo ordine è immediata. Ad esempio per un sistema di due equazioni utilizzando la formula di Heun si ha:

$$\begin{cases} y_1' = f_1\left(x, y_1, y_2\right) \\ y_2' = f_2\left(x, y_1, y_2\right) \end{cases}$$

$$y_1^{n+1} = y_1^n + \frac{1}{2}\left(k_1^1 + k_1^2\right)$$

$$k_1^1 = hf_1\left(x_i, y_1^n, y_2^n\right)$$

$$k_1^2 = hf_1\left(x_i + h, y_1^n + k_1^1, y_2^n + k_2^1\right)$$

$$y_2^{n+1} = y_2^n + \frac{1}{2}\left(k_2^1 + k_2^2\right)$$

$$k_2^1 = hf_2\left(x_i, y_1^n, y_2^n\right)$$

$$k_2^2 = hf_2\left(x_i + h, y_1^n + k_1^1, y_2^n + k_2^1\right)$$

Equazioni di ordine superiore al primo

La soluzione di equazioni differenziali di ordine superiore al primo passa attraverso la trasformazione dell'equazione in un sistema di equazioni del primo ordine. Ad esempio per un'equazione del secondo ordine del tipo:

$$y'' = f(x, y, y')$$

Con le posizioni:

$$y_1 = y \quad y_2 = y'$$

Si ottiene il sistema:

$$\begin{cases} y_1' = y_2 \\ y_2' = f(x, y_1, y_2) \end{cases}$$

Problemi al contorno

Si esamina il caso di un'equazione differenziale ordinaria lineare del secondo ordine:

$$\begin{cases} y'' + m(x)y' + n(x)y = f(x) \\ \quad y(a) = a_o \quad y(b) = b_o \end{cases}$$

La soluzione generale di questa equazione è:

$$y = c_1 y_1 + c_2 y_2 + y_p$$

Dove y_1 e y_2 sono due soluzioni dell'equazione omogenea associata e y_p è una soluzione particolare. Integrando numericamente l'equazione completa con le seguenti condizioni iniziali si ottiene la soluzione particolare.

$$y_p(a) = a_o \quad y_p'(a) = 0$$

Mentre integrando, sempre numericamente, l'equazione omogenea associata con le seguenti condizioni al contorno si ottiene y_1.

$$y_1(a) = 0 \quad y_1'(a) = 1$$

Problemi al contorno

Senza ledere la generalità si impone c_2=0 e si ha:

$$y = c_1 y_1 + y_p$$

Per x=a:

$$y(a) = c_1 y_1(a) + y_p(a) = y_p(a) = a_o$$

Che è chiaramente soddisfatta per qualunque valore di c_1. Mentre per x=b:

$$y(b) = c_1 y_1(b) + y_p(b) = b_o$$

In questa equazione è presente la sola incognita c_1. Risolvendo si ha:

$$c_1 = \frac{b_o - y_p(b)}{y_1(b)}$$

La soluzione diventa:

$$y = \frac{b_o - y_p(b)}{y_1(b)} y_1 + y_p$$

Metodo di Eulero

Una possibile codifica del metodo di Eulero in Matlab è:

```
% \provaRK\eulero.m
function [t,y]=eulero(t0, h ,y0 ,df)
k1=h*df(t0,y0);
y=y0+k1;
t=t0+h;
return
```

$$y_{i+1} = y_i + h y_i' + o(h^2)$$

Soluzione delle equazioni

Per provare se il codice scritto funziona si utilizzerà un'equazione differenziale di cui è nota la soluzione:

$y(x)=cos(x)$ $\quad y'(x)=-sin(x)$ $\quad y''(x)=-cos(x)$

$y''(x)=yy'+cos(x)(sin(x)-1)$

Associata alle condizioni iniziali:

$y(0)=1$ $\quad y'(0)=0$

Con le posizioni:

$y_1=y$ $\quad y_2=y'$

Si ottiene il sistema d'equazioni differenziali:

$$\begin{cases} y_1'=y_2 \\ y_2'=y_1y_2+cos(x)(sin(x)-1) \end{cases}$$

Associato alle condizioni:

$$\begin{cases} y_1(0)=1 \\ y_2(0)=0 \end{cases}$$

Soluzione delle equazioni

$$\begin{cases} y_1'=y_2 \\ y_2'=y_1y_2+cos(x)(sin(x)-1) \end{cases}$$

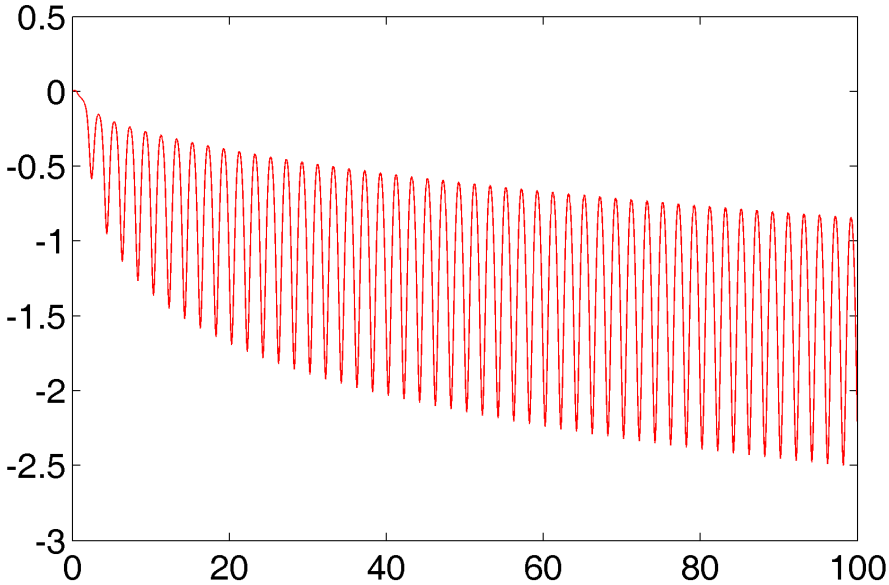
```
% \provaRK\DerCos.m
function DerCos = DerCos(x,y)
DerCos=[y(2)
    +y(1)*y(2)+cos(x)*(sin(x)-1)];

return
```

Metodo di Eulero

```
%\provaRK\provaEulero.m
L=[0 pi*100];%dominio d'integrazione
M0=[1; 0]; %Condizioni iniziali
h=0.025; %Passo integrazione
N=floor((L(2)-L(1))/h)+1;
xe=zeros(N,1);
Me=zeros(N,length(M0));
xe(1)=L(1);
Me(1,:)=M0;
tic
for i=1:N-1
   [xe(i+1) , Me(i+1,:)]=eulero(xe(i),h,Me(i,:)',@DerCos);
end
toc
plot(xe/pi,Me(:,1)-cos(xe), 'r')
```


Metodo di Eulero



Errore assoluto (Tempo di calcolo=0.3s)

Metodo di Eulero

In c lo stesso codice diventa:

```
#include "stdio.h"
#include "math.h"

#define PIGR 3.14159265358979323846
#define  N 2


int DerCos(double x, double *y0, double*y);
double Eulero(double t0, double h, double *y0, double *y, int n,int
df(double , double*, double*));
```


Metodo di Eulero

In c lo stesso codice diventa:

```
int main()
{
  double y0[N], y[N];
  double a=0,b=PIGR*1,x,h;
  int i;
  h=0.025;
  y0[0]=1;
  y0[1]=0;
  for (x=a;x<=b;)
  {
    x=Eulero(x, h, y0, y, N,DerCos);
    for (i=0;i<N;i++)
      y0[i]=y[i];
    printf("%g %g\n",x/PIGR, y[0]);

  }
  return 0;
}
```

Metodo di Eulero

In c lo stesso codice diventa:

```
int DerCos(double x, double *y0, double *y)
{
  y[0]=y0[1];
  y[1]=y0[0]*y0[1]+cos(x)*(sin(x)-1.);
  return 0;
}

double Eulero(double t0, double h, double *y0, double *y, int n,int
df(double , double*, double*))
{
  int i;
  df(t0,y0,y);
  for(i=0;i<N;i++)
  {
    y[i]=y0[i]+h*y[i];
  }
  return t0+h;
}
```


Metodo di Heun

Una possibile codifica del metodo di Heun in Matlab è:

```
% \provaRK\heun.m
function [t,y]=heun(t0, h ,y0 ,df)
k1=h*df(t0,y0);
k2=h*df(t0+h,y0+k1);
y=y0+(k1+k2)/2;
t=t0+h;
return
```

$$y_{i+1} = y_i + \frac{1}{2}(k_1 + k_2)$$

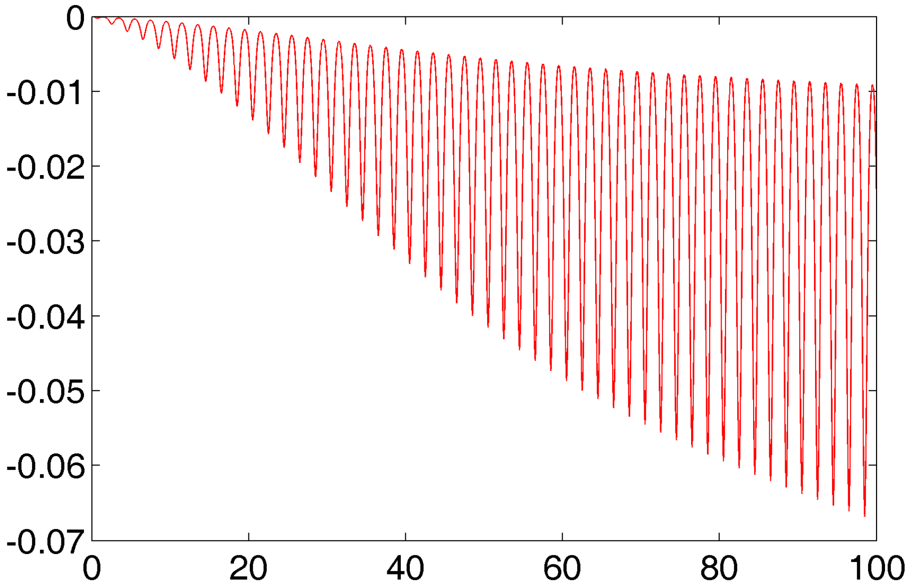
$$k_1 = hf(x_i, y_i)$$

$$k_2 = hf(x_i + h, y_i + hf)$$

Metodo di Heun

```
%\provaRK\provaHeun.m
L=[0 pi*100];%dominio d'integrazione
M0=[1; 0]; %Condizioni iniziali
h=0.025; %Passo integrazione
N=floor((L(2)-L(1))/h)+1;
xe=zeros(N,1);
Me=zeros(N,length(M0));
xe(1)=L(1);
Me(1,:)=M0;
tic
for i=1:N-1
    [xe(i+1) , Me(i+1,:)]=heun(xe(i),h,Me(i,:)',@DerCos);
end
toc
plot(xe/pi,Me(:,1)-cos(xe), 'r')
```


Metodo di Heun



Errore assoluto (Tempo di calcolo=0.37s)

Metodo RK4

Una possibile codifica del metodo RK4:

```
% \provaRK\rk4.m
function [t,y]=rk4(t0, h ,y0 ,df)
k1=h*df(t0,y0);
k2=h*df(t0+h/2,y0+k1/2);
k3=h*df(t0+h/2,y0+k2/2);
k4=h*df(t0+h,y0+k3);
y=y0+(k1+2*k2+2*k3+k4)/6;
t=t0+h;
return
```

$$y_{i+1} = y_i + \frac{1}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right)$$

$$k_1 = hf\left(x_i, y_i\right)$$

$$k_2 = hf\left(x_i + \frac{h}{2}, y_i + \frac{k_1}{2}\right)$$

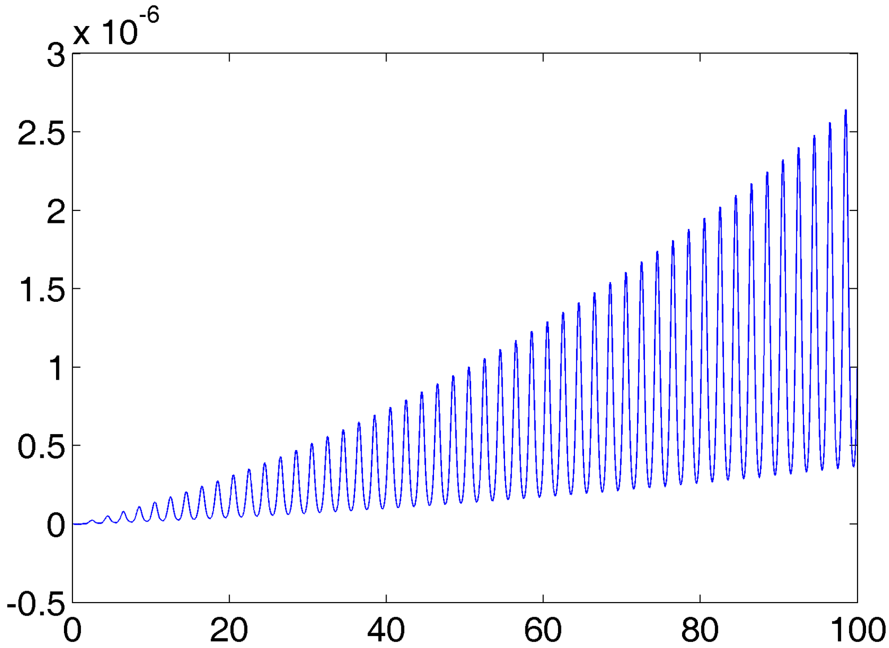
$$k_3 = hf\left(x_i + \frac{h}{2}, y_i + \frac{k_2}{2}\right)$$

$$k_4 = hf\left(x_i + h, y_i + k_3\right)$$

Metodo RK4

```
% \provaRK\provaRK_1.m
L=[0 pi*100];%dominio d'integrazione
M0=[1; 0]; %Condizioni iniziali
h=0.025; %Passo integrazione
N=floor((L(2)-L(1))/h)+1;
x=zeros(N,1);
M=zeros(N,length(M0));
x(1)=L(1);
M(1,:)=M0;
xe=x;
Me=M;
tic
for i=1:N-1
   [x(i+1) , M(i+1,:)]=rk4(x(i),h,M(i,:)',@DerCos);
 %  [xe(i+1) , Me(i+1,:)]=heun(xe(i),h,Me(i,:)',@DerCos);
end
toc
plot(x/pi,M(:,1)-cos(x));
```

Metodo RK4



Errore assoluto (Tempo di calcolo=0.49s)

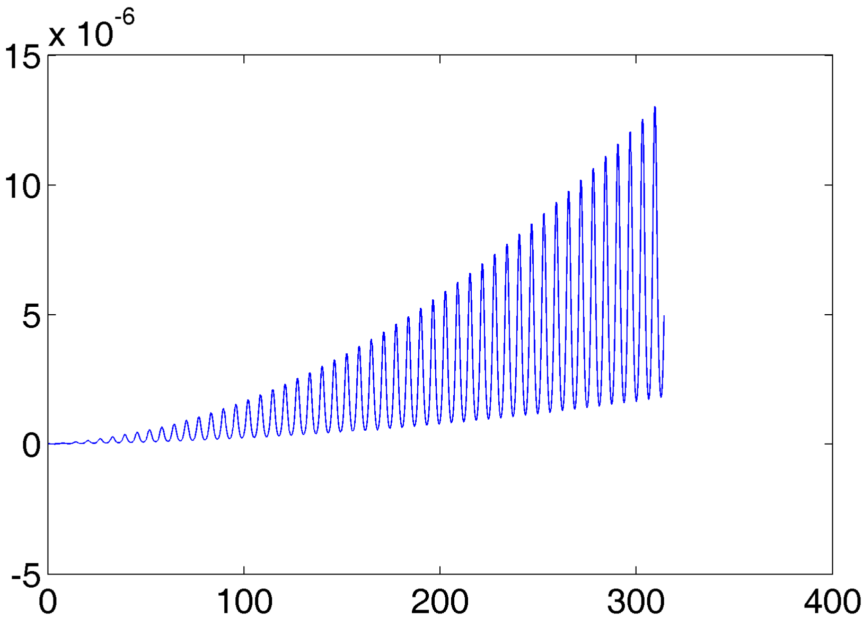
Soluzione delle equazioni

Utilizzando una routine standard di matlab si ottiene un risultato analogo:

```
% \provaRK\provaRK_2.m

L=[0 pi*100]; %dominio d'integrazione
M0=[1; 0]; %Condizioni iniziali
options=odeset('RelTol',1e-8, 'AbsTol', [1e-8]);
tic
[x M]=ode45(@DerCos,L,M0,options);
toc
plot(x,M(:,1)-cos(x), '.-' );
```

Soluzione delle equazioni



Errore assoluto (Tempo di calcolo=0.45s)

Soluzione delle equazioni

Si analizzi ora una funzione non periodica:

$$y(x) = tanh(ax) \qquad y'(x) = a\,sech^2(ax) \qquad y''(x) = -2a^2\,sech^2(ax)tanh(ax)$$

$$y''(x) = -2a^2\,sech^2(ax)tanh(ax)$$

Associata alle condizioni iniziali:

$$y(0) = 0 \qquad y'(0) = a$$

Con le posizioni:

$$y_1 = y \qquad y_2 = y'$$

Si ottiene il sistema d'equazioni differenziali:

$$\begin{cases} y_1' = y_2 \\ y_2' = -2a^2\,sech^2(ax)tanh(ax) \end{cases}$$

Associato alle condizioni:

$$\begin{cases} y_1(0) = 0 \\ y_2(0) = a \end{cases}$$

Soluzione delle equazioni

$$\begin{cases} y_1' = y_2 \\ y_2' = -2a^2 \, sech^2(ax) tanh(ax) \end{cases}$$

```
%\provaRK\DerTanh.m
function DerTanh = DerTanh(x,y)
global a;
DerTanh=[y(2)
  % -y(2)*a*tanh(a*x)-y(1)*a*a*sech(a*x)^2];
    -2*tanh(a*x)*a*a*sech(a*x)^2];

return
```


Soluzione delle equazioni

```
% \provaRK\provaRK_1Tanh.m
global a;
a=10;
L=[0 10];%dominio d'integrazione
M0=[0; a];  %Condizioni iniziali
h=0.025;  %Passo integrazione
N=floor((L(2)-L(1))/h)+1;
x=zeros(N,1);
M=zeros(N,length(M0));
x(1)=L(1);
M(1,:)=M0;
xe=x;
Me=M;
tic
for i=1:N-1
    [x(i+1) , M(i+1,:)]=rk4(x(i),h,M(i,:)',@DerTanh);
    %[xe(i+1) , Me(i+1,:)]=eulero(xe(i),h,Me(i,:)',@DerTanh);
end
```

Soluzione delle equazioni

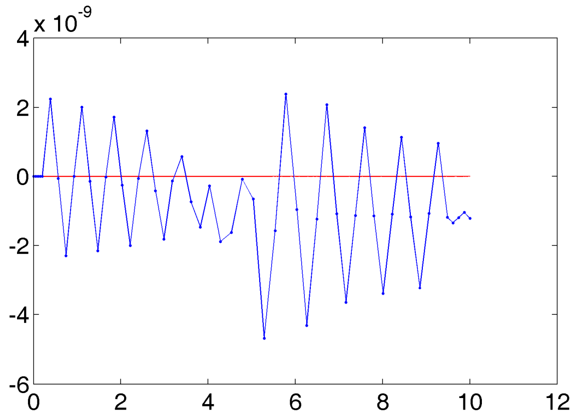
```
toc
options=odeset('RelTol',1e-8, 'AbsTol', [1e-8]);
tic
[xm Mm]=ode45(@DerTanh,L,M0,options);
toc
figure (1)
plot(x,M(:,1)-tanh(a*x), 'r',xm,Mm(:,1)-tanh(a*xm), '.-b' );
figure (2)
plot(xm,tanh(a*xm),'.-' )
```

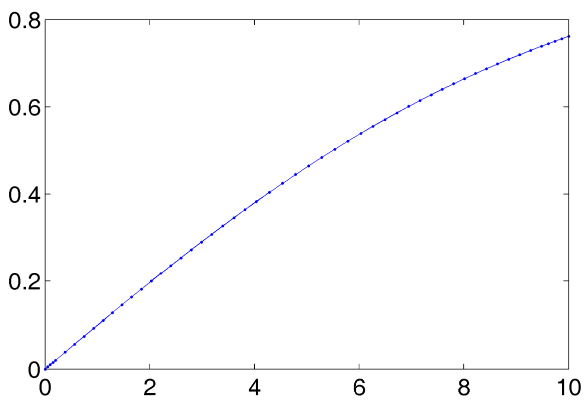

Soluzione delle equazioni

Per a =0.1 i tempi di calcolo sono

Rk4: 0.043s

Ode45: 0.0073s



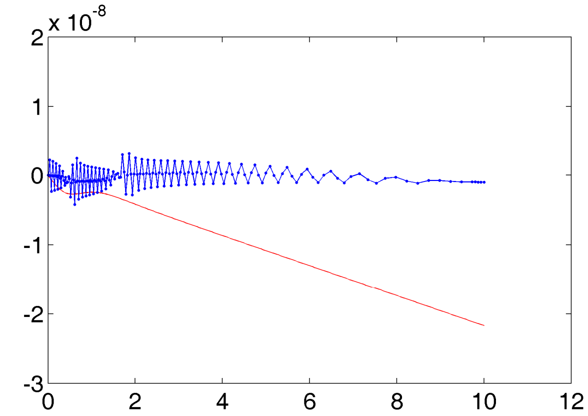


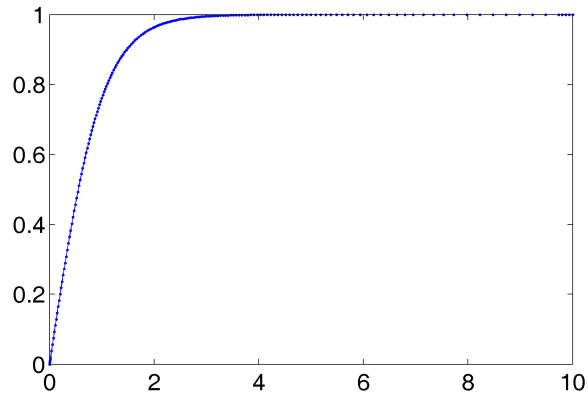
Soluzione delle equazioni

Per a =1 i tempi di calcolo sono

Rk4: 0.043s

Ode45: 0.012s





Soluzione delle equazioni

Per a =10 i tempi di calcolo sono

Rk4: 0.043s

Ode45: 0.015s