Laboratorio di Informatica per Ingegneria elettrica
A.A. 2010/2011
Prof. Sergio Scippacercola

# Analisi numerica

**N.B. le slide devono essere utilizzate solo come riferimento agli argomenti trattati a lezione: LO STUDENTE DEVE INTEGRARE LO STUDIO CON IL LIBRO DI TESTO [Cavallo et. al., La nuova guida a MATLAB, Liguori editore (2002)].**

# Derivazione

- Operatore `diff(x)` calcola le differenze fra gli elementi adiacenti di un vettore
- `Es.: diff(x)= [x(2)-x(1), x(3)-x(2), …, x(n)-x(n-1)]`
- Data una funzione `y=f(x)`la derivata approssimata al primo ordine è:
- `dy= diff(y)./diff(x);`

# Derivazione

- `Es.: derivata di seno(x)`

- x = linspace(0, 2*pi, 1000);
- y = sin(x);
- yd = diff(y) ./ diff(x);
- plot(x(1:999), yd)

# Derivazione

Calcolo differenza n-esima

- `diff(x,n)`
- `Es.:`
- x=[4 -5 9 8];
- >> diff(x,1)
- ans =
- -9 14 -1
- >> diff(x,2)
- ans =
- 23 -15
- >> diff(x,3)
- ans =
- -38
- >> diff(x,4)
- ans =
- []

# Derivazione

- Gradiente di funzioni a due variabili

```
[dZx, dZy] = gradient (Z,dx,dy)
```

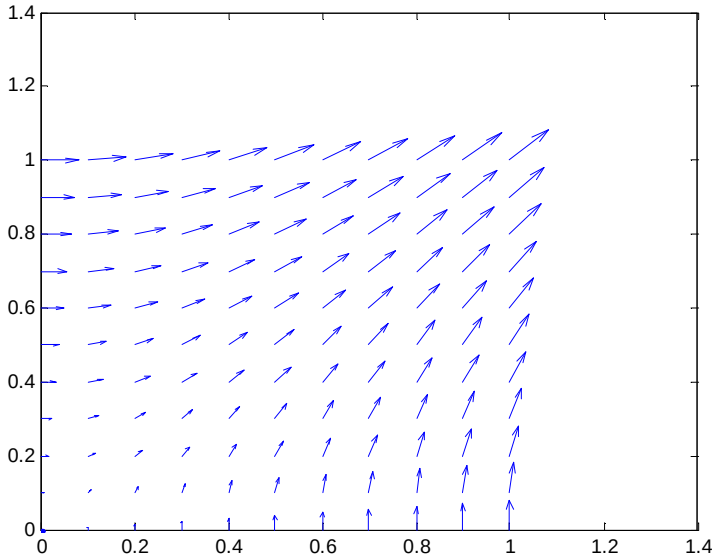
Z contiene i valori della funzione da derivare
dx, dy definiscono la spaziatura uniforme tra i campioni di Z
dZx, dZy sono le derivate parziali di Z

- Utile anche per stimare derivata prima di una funzione in una variabile: `[dY] = gradient (Y,dx)`

- Esempio: calcolare la derivata di log(x) mediante il gradient
- `>> passo = 0.01;`
- `>> x = [0.5:passo:10];`
- `>> y = log(x);`
- `>> yd = gradient(y, passo);`
- `>> plot(x, yd)`

# Derivazione

Esercizio:
Calcolo del gradiente della funzione Z=(X*Y)



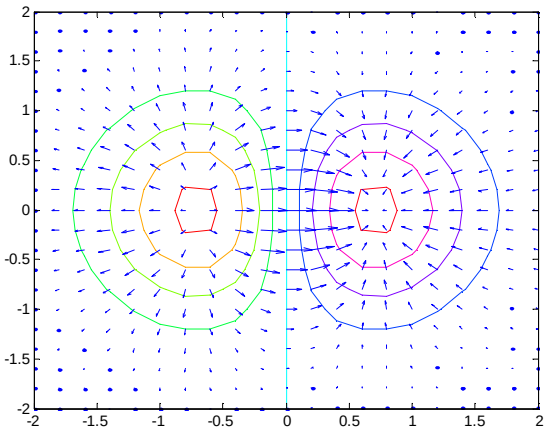

- >> passo = 0.1;
- >> x = [0:passo:1];
- >> y = [0:passo:1];
- >> [X, Y] = meshgrid(x, y);
- >> Z = X .* Y;
- >> [dFx, dFy] = gradient(Z, passo, passo);
- >> quiver(X, Y, dFx, dFy)  %funzione che disegna frecce vettori con componenti (u,v) ai punti (x,y)

# Derivazione

Esercizio:
Calcolo del gradiente della funzione $z = xe^{(-x^2-y^2)}$

[dZx, dZy] = gradient (Z,dx,dy)

- [X,Y] = meshgrid(-2:.2:2);
- Z = X.*exp(-X.^2 - Y.^2);
- [DX,DY] = gradient(Z,.2,.2);
- contour(X,Y,Z)  % disegna curve di livello
- hold on
- quiver(X,Y,DX,DY)
- colormap (hot)  %colora il grafico
- hold off



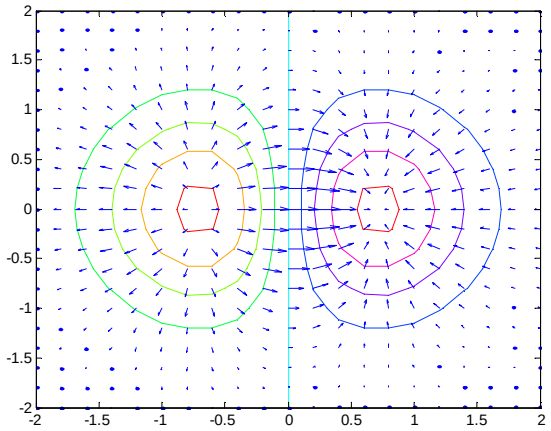
# Derivazione

Esempio senza fare uso della funzione meshgrid,
Calcolo del gradiente della funzione $z = xe^{(-x^2-y^2)}$

[dZx, dZy] = gradient (Z,dx,dy)

- X=[-2:.2:2];
- [k,l]=size(X);  %funzione che calcola k=righe di X ed l=colonne di X
- X=repmat(X,l,1); %replica X per l righe
- Y=X';  %le matrici X e Y formano la griglia per il calcolo della funzione Z=f(X,Y)

- Z = X.*exp(-X.^2 - Y.^2);
- [DX,DY] = gradient(Z,.2,.2);
- contour(X,Y,Z)   % disegna curve di livello
- hold on
- quiver(X,Y,DX,DY)
- colormap (hot)      %colora il grafico
- hold off



# Integrazione

- Matlab valuta l'integrale numerico definito di una funzione mediante l'algoritmo iterativo di Quadratura di Simpson
  - `q = quad(funzione,a,b)`
  - `q = quad(funzione,a,b,tol)`
- Essendo adattativo, è possibile specificare l'accuratezza dell'approssimazione impostando la tolleranza (valore di default è 1.0e-6).

# Integrazione

- Il primo parametro di quad deve essere il nome della funzione, non i suoi valori.
- Si deve specificare con una stringa la forma della funzione, oppure si usano puntatori a funzioni (@).
- Es:
  >> quad('sin(x)+cos(x)', 0, pi/2)
- ans =
- 2.0000
- Se non si possiede la forma analitica ma soltanto i valori (x, y) della funzione, si può usare un algoritmo differente di integrazione: trapz(x,y)

# Integrazione

- Es:
  inserire in un M-file
- `>> function y = funzione(x)`
- `        y = 1./(x.^3-2*x-5);`

- Richiamare la funzione con:

- `quad(@funzione,0,2)`

- `ans =`
- `    -0.4605`

# Equazioni non lineari

- Zero di una funzione continua in una variabile
- `x = fzero('funzione', x0)`
- x0 è un punto vicino al quale si stima esserci uno zero.

- Es.:
- `>> fzero('cos', 0)`
- `ans =`
- `   -1.5708`

- Es.: con funzione in M-file `function y = funzione(x)`
- `                    y = 1./(x.^3-2*x-5);`

- `fzero(@funzione, 0)`
- `ans =`

- `    2.0946`

# Equazioni non lineari

- Zero di funzioni di più variabili
- x = fsolve(‘funzione’, X0)
- X0 è un punto n-dimensionale, vicino al quale si stima esserci uno zero.
- 
- Es.: $X^2=A$  con A=[2.25 7;3.5 11]
- `s = fsolve(‘x*x-[2.25 7; 3.5 11]', ones(2, 2))`
- `s = 0.5000    2.0000`
- `    1.0000    3.0000`

# Equazioni differenziali ordinarie

- L'equazione da integrare deve essere in forma normale: sistema di equazioni del primo ordine

$$dx/dt = f(t, x)$$

- Con x un vettore di cardinalità pari all'ordine dell'equazione e
- t la variabile indipendente (es. il tempo)

# Equazioni differenziali ordinarie

- Esistono vari algoritmi (ode23, ode45, ode113, ode15s, ode23s, ode23t, ode23tb) a seconda della natura del problema, tutti con stessa interfaccia:
  - `[t,x] = odexxx(@funzione, [t0, tf],x0)`
- **Parametri:**
  - `funzione = function file di tipo M che realizza l'equazione differenziale e che prevede i due argomenti di ingresso t e x e fornisce in uscita la derivata temporale di x come vettore colonna.`
- `[t0, tf]  estremi dell'intervallo di integrazione`
- `x0, vettore di valori iniziali`

- **Uscite:**
- x contiene la soluzione dell'equazione
- t contiene i corrispondenti valori della variabile indipendente

- Il passo di integrazione degli algoritmi è variabile, essi riportano in output sia le ascisse che le ordinate (T, Y).

# Equazioni differenziali ordinarie

- Equazioni differenziali “non stiff” quando la soluzione non evolve su scale temporali molto diverse tra loro.
- Equazioni “stiff” in caso contrario.

- Tale  classificazione è dovuta a difficoltà nella scelta del passo di integrazione

# Equazioni differenziali ordinarie

- ode23, ode45 ode113 adatti per “non stiff”
- Ode45 di tipo generale
- ode15s, ode23s, etc. per problemi “stiff”

# Equazioni differenziali ordinarie

- Es: equazione

$$\ddot{x} + (x^2 - 1)\dot{x} + x = 0$$

- Fase a) Portare in forma normale

$$\dot{x}_1 = x_2$$
$$\dot{x}_2 = x_2(1 - x_1^2) - x_1$$

- Fase B) Creare un function file .m

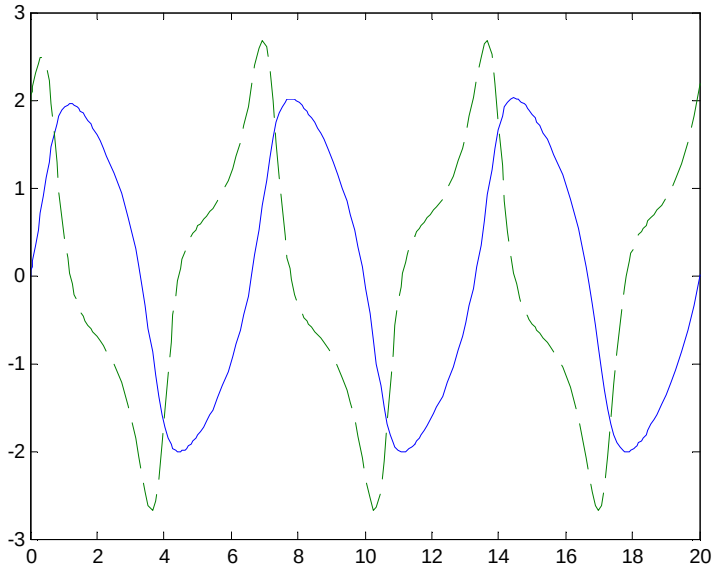
```
function xPunto = vdpol(t, x)
    x1 = x(2);
    x2 = x(2) .* (1 - x(1) .^2) - x(1);
    xPunto = [x1; x2];
end
```

- Fase C) Simulare l'equazione differenziale in [0; 20] con condizione iniziale x(0)=(0,2)

- `[t,x]=ode45(@vdpol,[0,20],[0,2]);`

# Equazioni differenziali ordinarie

- **Fase D) grafico**
- `plot(t,x(:,1),'-',t,x(:,2),'--');`



# Problema stiff

- Modello motore elettrico in continua con eccitazione costante

$$\dot{x}_1 = -\frac{\beta}{J} x_1 + \frac{K}{J} x_2$$

$$\dot{x}_2 = -\frac{K}{L} x_1 + \frac{R}{L} x_2 + \frac{V}{L}$$

$x_1 = \omega$ ***velocità di rotazione del motore***

$x_2 =$ ***i la corrente di armatura***

$J$ ***il momento di inerzia del motore***

$\beta$ ***il coefficiente di attrito viscoso***

$K$ ***una costante*** $R$ ***la resistenza*** $L$ ***induttanza di armatura***

$V$ ***la tensione di alimentazione***

# Problema stiff

- Modello motore elettrico in continua con eccitazione costante

$$\dot{x}_1 = -\frac{\beta}{J} x_1 + \frac{K}{J} x_2$$

$$\dot{x}_2 = -\frac{K}{L} x_1 + \frac{R}{L} x_2 + \frac{V}{L}$$

- function xdot=motore(t,x)
- global beta J K R L V
- xdot=[-beta/J*x(1)+K/J*x(2);-K/L*x(1)-R/L*x(2)+1/L*V];

- global beta J K R L V
- % calcola la soluzione per un motore elettrico a partire dalle
- % condizioni iniziali nulle con tensione di alimentazione V
- % e tempo finale t forniti in input.
- J=1e-2;  R=10;  K=0.01; beta=1e-3;  L=10e-3;
-
- V=input('dammi tensione es. 10Volt:  ')
- t0=input('dammi tempo finale es. 60 s:   ')

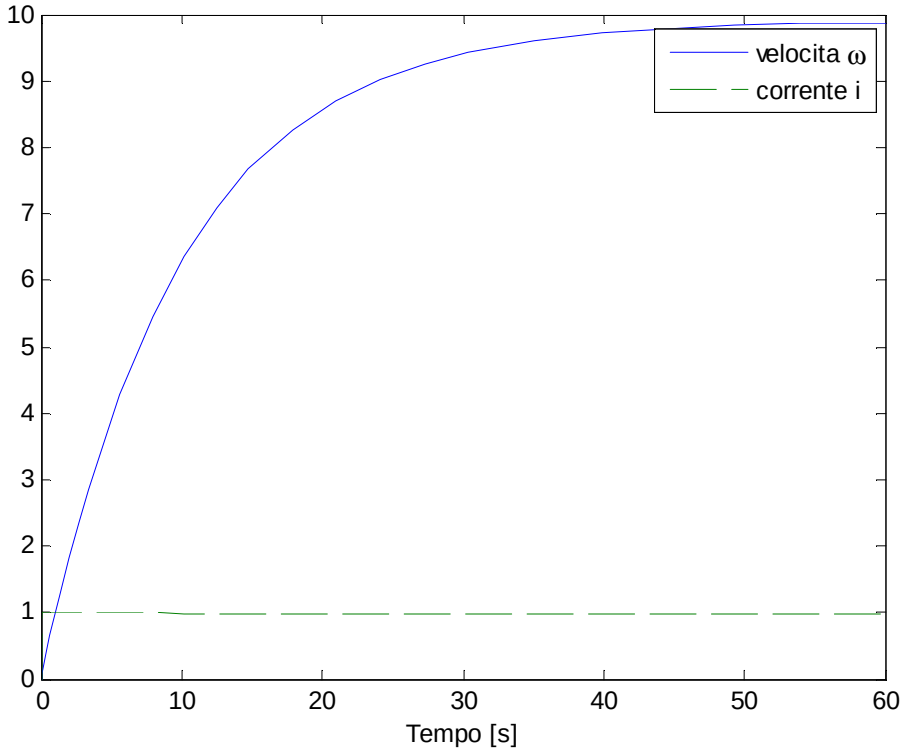
- [t,x]=ode15s(@motore,[0 t0],[0;0]);

- figure(1),plot(t,x(:,1),'-',t,x(:,2),'--')
- legend('velocita \omega','corrente i'),xlabel('Tempo [s]')

# Problema stiff

- Modello motore elettrico in continua con eccitazione costante:

è problema stiff perché la componente elettrica è più veloce di quella meccanica



# Esercizi

- Calcolare le radici dell'equazione

$$x^2 - 3\sin x + 0.1 = 0$$

```
Equazione non lineare:
fzero('x.^2 -3 * sin(x) +0.1', 0)
```

ans =

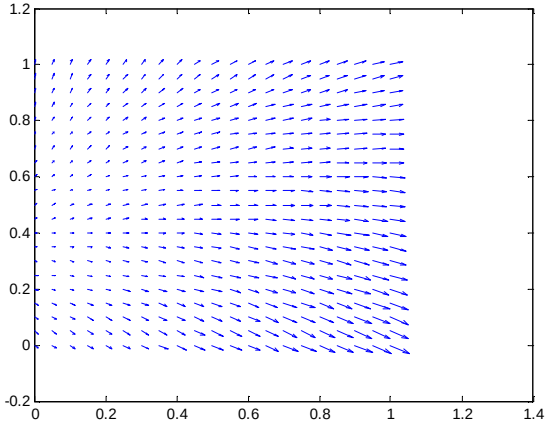
0.0337

# Esercizi

- Trovare il gradiente della funzione nel dominio

  0 <= x <= 1, 0 <= y <= 1

$$f(x,y) = e^{x-y} + x^2 + y^2$$




```
>> passo = 0.05;
>> t = [0:passo:1];
>> [X, Y] = meshgrid(t, t);
>> Z = exp(X - Y) + X.^2 + Y.^2;
>> [dFx, dFy] = gradient(Z, passo, passo);
>> quiver(X, Y, dFx, dFy)
```

# Esercizi

- Integrare la funzione nell'intervallo [0; pi/2]

$$\sin x \cos x$$

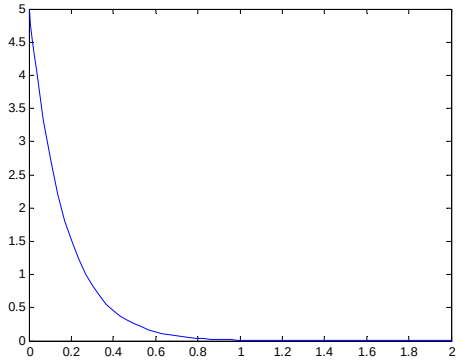
```
>> quad('sin(x).*cos(x)', 0, pi/2)
ans =

    0.5000
```

# Esercizi

- Integrare la funzione differenziale ordinaria del primo ordine nell'intervallo [0, 2] con condizione iniziale x0 = 5

$$\frac{1}{2}\dot{x} + 3x = 0$$

# Soluzione




- Integrare la funzione differenziale ordinaria

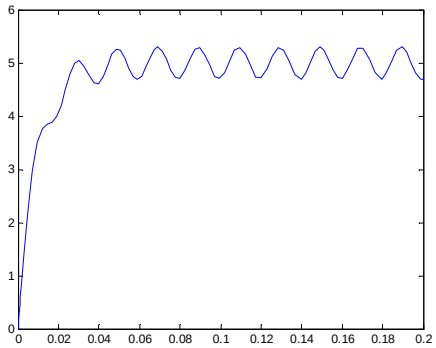
$$\frac{1}{2}\dot{x} + 3x = 0; \quad \dot{x} = -6x$$

```
function xPunto = eqDiff(t, x)
     xPunto = -6 * x;
end

[t, x] = ode45('eqDiff', [0 2], 5);
plot(t, x)
```

# Esercizio

- Simulazione circuito RC con resistenza di 1 kOhm e capacità di 10 microFarad con generatore di tensione a componente continua di 5 V e alternata di ampiezza unitaria e frequenza di 50 Hertz. Il condensatore inizialmente è scarico

```
function xPunto = rcEsercizio(t, x)
f = 50;
amp = 1;
R = 1e3;
C = 10e-6;
DC = 5;
v=DC+amp*sin(2*pi*f*t);
xPunto = -1/(R*C)*(x-v);
end

[t x] = ode45(@rcEsercizio, [0 0.2], 0);
plot(t,x)
```

$$RC\dot{x} + x = (5 + \sin(2\pi 50 t))u(t)$$

$$\dot{x} = -\frac{1}{RC}(x - (5 + \sin(2\pi 50 t)u(t))$$