Analisi Numerica I, a.a. 2004-2005

Docente: M.Gaviano

Esercitazione n.1

Mediante MatLab

1) Costruisci le seguenti matrici

$$\begin{bmatrix} 0.9501 & 0.8913 & 0.8214 & 0.9218 \\ 0.2311 & 0.7621 & 0.4447 & 0.7382 \\ 0.6068 & 0.4565 & 0.6154 & 0.1763 \\ 0.4860 & 0.0185 & 0.7919 & 0.4057 \end{bmatrix}$$

$$\begin{bmatrix} 0.8913 & 0.4565 & 0.8214 & 0.7643 \\ 0.7621 & 0.0185 & 0.4447 & 0.2134 \end{bmatrix}$$

- Calcola la matrice prodotto C= A*B'
- Mediante la funzione **size** definita in MatLab calcola le dimensioni delle matrici A, B, e C.
- Costruisci la matrice D, 6x4, formata dalle 4 righe che compongono A e le 2 righe che compongono B.
- Costruisci dalla matrice A la matrice E composta dalle prime due colonne di A.
- Costruisci dalla matrice A la matrice F ottenuta eliminando la prima ed ultima riga, e la prima ed ultima colonna.

2)

- Genera la matrice identità di dimensione 2, 4, 6 e 8.
- Genera una matrice casuale (mediante la funzione rand() predefinita in Matlab) con valori compresi tra 0 e 10 di dimensione 2, 4.
- Genera una matrice 4x4 casuale con valore compresi tra 0 e 1. Calcolane il determinante e l'inversa.
- Genera 2 matrici A e B, 4x4, casuali con valori compresi tra -2 e 2.
- Costruisci una matrice C ottenuta moltiplicando gli elementi di A e B elemento per elemento ovvero $c_{ij}= a_{ij}* b_{ij}$
- Costruisci una matrice D ottenuta dividendo gli elementi di A e B elemento per elemento ovvero $d_{ij}= a_{ij}/ b_{ij}$

3) Utizzando l'help verifica l'utilizzo delle funzioni per matrici

rand(), inv(), eye(), zeros(), ones()

- Calcola: rand(6), eye(4), zeros(6), ones(3), inv(A) con A matrice 5x5 casuale con valori compresi tra -10 e 10
- Calcola la soluzione del seguente sistema di 4 equazioni in 4 incognite: Ax=b in cui A è una matrice casuale 4x4 e b un vettore casuale 4x1 di elementi compresi tra 0e 1. Verifica l'esattezza della soluzione.
- Genera i vettori riga
  x1 con 20 elementi scelti equidistanti nell'intervallo $[-\pi, \pi]$
  x2 con 12 elementi equidistanti nell'intervallo [–4, 4]

4) Costruisci un file MatLab che, quando eseguito, chiede come dati un intervallo [a,b] un numero intero n e quindi genera un vettori riga x costituito da n punti equidistanti nell'intervallo [a,b].

5) Costruisci funzioni MatLab di nome fun1, fun2, fun3 e fun4 (memorizzate in files di nome fun1.m, fun2.m, fun3.m e fun4.m) che calcolino le funzioni

$y=2*\sin(x)$, $y=\sin(x)+x+4$, $y=x^2+x^3+5$, $y=1/(x^2+1)$

Fai in modo che se x è un vettore allora viene restituita la tabulazione della funzione.

6) Costruisci un file MatLab di nome grafico.m che tracci il grafico delle funzioni

$y=2*\sin(x)$, $y=\sin(x)+x+4$, $y=x^2+x^3+5$, $y=1/(x^2+1)$

in un intervallo da assegnare ogni volta che il file viene eseguito. Il numero di punti per la tabulazione sia ugualmente assegnato da tastiera.

Analisi Numerica I, a.a. 2004-2005

Docente: M.Gaviano

Esercitazione n.2

1)

Scrivere un programma in MATLAB che calcoli le radici reali ($D \geq 0$, $D=b^2 - 4ac$) dell'equazione di secondo grado

$$ax^2+bx+c=0$$

ovvero calcoli le radici x_1 e x_2 date da

$$x_{12} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Nel caso $D<0$, $D=b^2 - 4ac$ il programma deve solo dare il seguente messaggio: "radici complesse coniugate".
Fare il test sui seguenti problemi

a	b	c
2	6	1
3	3	0
1	3	1

2)

Scrivere un programma in MATLAB che calcoli le radici reali o complesse coniugate dell'equazione di secondo grado

$$ax^2+bx+c=0$$

ovvero per $D \geq 0$ calcoli le radici x_1 e x_2 date da

$$x_{12} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Per $D<0$ calcoli le radici complesse coniugate x_1 e x_2 date da

$$x_{1,2} \equiv (parte\_reale(=\frac{-b}{2a}) \pm parte\_immaginaria(=\frac{(\sqrt{4ac - b^2})}{2a})$$

Fare il test sui seguenti problemi

a	b	c
2	6	1
3	3	0
1	3	1
0	12	-3
3	6	3
2	-4	3

3)

Scrivere un programma in MATLAB che calcoli le radici reali o complesse coniugate dell'equazione di secondo grado in modo da sfruttare la capacità di matlab di operare con numeri complessi.
Fare il test sui seguenti problemi

a	b	c
2	6	1
3	3	0
1	3	1
0	12	-3
3	6	3

Analisi Numerica I, a.a. 2004-2005

Docente: M.Gaviano

Esercitazione n.3

a) Scrivere un programma in Matlab che legga un valore x e ne calcoli il seno utilizzando la serie infinita

$$\sin x = x - x^3/3! + x^5/5! - x^7/7! + ...+ x^n/7!.$$

dove *x* è espresso in radianti ed n è un numero intero introdotto da tastiera.
Visualizzare la differenza tra il valore di sin(x) calcolato dal programma e quello calcolato con la funzione predefinita di MatLab. Rendere il programma interattivo, ovvero l'esecuzione continui assegnando valori successivi di n e termini quando l'utente lo desidera

b) Scrivere un programma in Matlab che legga un valore x e ne calcoli il seno utilizzando la serie infinita

$$\sin x = x - x^3/3! + x^5/5! - x^7/7! + ...$$

dove *x* è espresso in radianti. Nel sommare gli elementi successivi della serie interrompere l'operazione quando la differenza tra gli ultimi 2 termini (calcolati) della serie è minore in valore assoluto di 10^{-8} (accuratezza). Visualizzare la differenza tra il valore di sin(x) calcolato dal programma e quello calcolato con la funzione predefinita di MatLab. Visualizzare anche il numero di iterazioni necessarie a raggiungere l'accuratezza. Rendere il programma interattivo, ovvero l'esecuzione continui assegnando valori successivi di x e termini quando l'utente lo desidera

Fare il test con vari valori di x.

Analisi Numerica I, a.a. 2004-2005

Docente: M.Gaviano

Esercitazione n.4

a) Scrivi un algoritmo che calcoli la precisione macchina di MatLab e del compilatore C della Borland col tipo di dati *float* e *double* (solo per gli informatici). . Lo schema dell'algoritmo può essere

algoritmo(precisione macchina)

```
eps=1.
eps1=eps+1;
while eps1> 1,
      eps=0.5*eps;
      eps1=eps+1
end
stampa 'la precisione macchina è: 2*eps
```

Calcola i valori approssimati di t, L, U per il sistema floating point di MatLab e del compilatore Borland col tipo di dati *float* e *double.*

b) La funzione

$$y_1= x^6-6x^5+15x^4-20x^3 +15x^2-6x+1$$

può essere scritta anche nella forma

$$y_2=(x-1)^6$$

Fai il grafico della funzione utilizzando le 2 rappresentazioni. Ovvero fai il grafico di y_1 e y_2 nell'intervallo [0.994, 1.006] nella stessa figura. Cerca di capire perché il grafico della rappresentazione y_1 ha un andamento non regolare.

c)

Si sa da un qualsiasi libro di Analisi Matematica che per |x|<1 la serie

$$\sum_{i=0}^{n} x^i$$

tende a 1/(1-x) per n che assume valori crescenti. Scrivi un algoritmo in MatLab che legge un numero reale x, controlla se il suo valore assoluto è minore di 1, quindi se la risposta è affermativa, legge un numero reale positivo Acc quale accuratezza, e determina l'intero n* tale che

$$|\sum_{i=0}^{n^*} x^i - \frac{1}{1-x}| \le Acc$$

(Considera che $x^i = x^{i-1} * x$. Quindi ciascun termine della serie può essere ottenuto eseguendo una sola moltiplicazione)

Esempio:

Input:

x = 0.6 Acc = 0.001

Output:

n* = 16

Analisi Numerica I, a.a. 2004-2005

Docente: M.Gaviano

Esercitazione n.5

Un algoritmo per la risoluzione di sistemi triangolari inferiori del tipo Ly=b è il seguente:

1**. algoritmo**

$y_1=b_1/l_{11}$
for i=2,…,n
  $y_i=b_i$
  for j=1,…, i-1
    $y_i=y_i-l_{ij}y_j$
  end
  $y_i=y_i/l_{ii}$
end

Scrivi un algoritmo in Matlab che
a) generi a caso un problema del tipo Ly=b di dimensione n la cui soluzione esatta yy è nota.
b) applichi l'algoritmo 1 per trovare la soluzione approssimata y.
c) stampi la soluzione trovata e la norma del massimo dell'errore yy-y.

Sperimenta l'algoritmo con n=10, 20,40, 60, 80,100,200, 300.

Suggerimento: Determina il problema a caso, generando una matrice A casuale di dimensione n i cui elementi sono tra 0 e 1. Estrai da A mediante la funzione tril() predefinita in MatLab, la triangolare inferiore ottenendo L. Scegli ugualmente a caso (elementi tra 0 e 1) il vettore colonna yy di dimensione n e quindi calcola b=L*yy.
I dati del problema sono n, L e b; yy è la soluzione *esatta* del problema. Rendi l'algoritmo MatLab interattivo.

Analisi Numerica I, a.a. 2004-2005

Docente: M.Gaviano

Esercitazione n.6

Un algoritmo per la risoluzione di sistemi triangolari superiori del tipo Ux=y è il seguente:

1**. algoritmo**

$x_n=y_n/u_{nn}$
for i=n-1,…,1
  $x_i=y_i$
  for j=i+1,…,n
    $x_i=x_i-u_{ij}x_j$
  end
  $x_i=x_i/u_{ii}$
end

Scrivi un algoritmo in Matlab che
a) generi a caso un problema del tipo Ux=y di dimensione n la cui soluzione esatta xx è nota.
b) applichi l'algoritmo 1 per trovare la soluzione approssimata x.
c) stampi la soluzione trovata e la norma euclidea dell'errore xx-x.

Sperimenta l'algoritmo con n=10, 20,40, 60, 80,100, 200,300.

Suggerimento:

Determina il problema a caso in 2 modi diversi (ottenendo problemi con caratteristiche differenti)

1. genera una matrice A casuale di dimensione n i cui elementi sono tra 0 e 1. Estrai da A mediante la funzione triu(), predefinita in MatLab, la triangolare superiore ottenendo U. Scegli ugualmente a caso (elementi tra 0 e 1) il vettore colonna xx di dimensione n e quindi calcola y=U*xx. n, U e y sono i dati del problema; xx è la soluzione *esatta* del problema. Rendi l'algoritmo MatLab interattivo.

2. genera una matrice A casuale di dimensione n i cui elementi sono tra 0 e 1. Estrai da A mediante la funzione triu(), predefinita in MatLab,la triangolare superiore ottenendo U. Modifica U ponendo gli elementi nella diagonale principale uguali a 1.
   Scegli ugualmente a caso (elementi tra 0 e 1) il vettore colonna xx di dimensione n e quindi calcola y=U*xx. n, U e y sono i dati del problema; xx è la soluzione *esatta* del problema. Rendi l'algoritmo MatLab interattivo.

Controlla il valore dell'errore al crescere di n.

Analisi Numerica I, a.a. 2004-2005

Docente: M.Gaviano

Esercitazione n.7

1) L'algoritmo di Gauss naive per la risoluzione di sistemi lineari Ax=b trasforma il sistema in uno triangolare superiore mediante il seguente:

1. **algoritmo**

```
for k=1,2,…,n-1
    for i=k+1,…,n
        m_{i,k} = a^{(k)}_{i,k} / a^{(k)}_{k,k}
        for j=k+1,…,n+1
            a^{(k+1)}_{ij} = a^{(k)}_{i,j} - m_{ik} a^{(k)}_{kj}
        end
    end
end
```

Scrivi un algoritmo in Matlab che risolva il sistema lineare Ax=b operando come segue

a) generi a caso un problema del tipo Ax=b di dimensione n la cui soluzione esatta xx è nota;
b) applichi l'algoritmo 1 per triangolarizzare il sistema;
c) trovi la soluzione approssimata x risolvendo il sistema triangolare superiore;
c) stampi la soluzione trovata e la norma euclidea dell'errore xx-x.

Sperimenta l'algoritmo con n=10, 20,40, 60, 80,100.

Suggerimento: Il problema a caso può essere determinato, generando una matrice A casuale di dimensione n i cui elementi sono tra 0 e 1. Si sceglie, ugualmente a caso, il vettore colonna xx di dimensione n e quindi si calcola b=A*xx. n, A e b sono i dati del problema; xx è la soluzione esatta del problema.

2) Modifica il metodo di Gauss naive in modo che sia applicabile a problemi Ax=b in cui $\det(A) \neq 0$ e qualche minore principale sia uguale a zero.
Suggerimento: Il problema a caso può essere determinato come nel caso precedente ed imponendo successivamente che due righe di qualche matrice principale siano uguali.

Analisi Numerica I, a.a. 2004-2005

Docente: M.Gaviano

Esercitazione n.8

Il seguente algoritmo di Gauss scalato triangolarizza un sistema lineare Ax=b in cui A e b sono memorizzati in un unico array A

1. algoritmo

for i=1,..,n
  $d_i=\max_{1<=j<=n+1}|a_{ij}|$
end
Scala il sistema lineare dividendo la riga i-sima del sistema per d_i.
for k=1,2,…,n-1
  p intero tale che $|a_{pk}^{(k)}| = \max_{k\le i\le n} |a_{ik}^{(k)}|$
  scambio $a_{kj}^{(k)}$ e $a_{pj}^{(k)}$, j=1,…,n+1.
  for i=k+1,…,n

$$m_{i,k} = \frac{a_{i,k}^{(k)}}{a_{k,k}^{(k)}}$$

    for j=k+1,…,n+1

$$a_{ij}^{(k+1)} = a_{i,j}^{(k)} - m_{ik} a_{kj}^{(k)}$$

    end
  end
end

Scrivi un algoritmo in Matlab che risolva il sistema lineare Ax=b operando come segue

Caso 1

a) generi a caso un problema del tipo Ax=b di dimensione n la cui soluzione esatta xx è nota;
b) applichi l'algoritmo 1 per triangolarizzare il sistema;
c) trovi la soluzione approssimata x risolvendo il sistema triangolare superiore;
d) stampi la soluzione trovata e la norma euclidea dell'errore xx-x.

Caso 2

a) generi un problema test Ax =b in cui A è la matrice di Hilbert e la cui soluzione esatta xx è nota;

I passi b), c), d) sono gli stessi del caso 1

Sperimenta l'algoritmo con n=10, 20,40, 60, 80,100.

Suggerimento: Il problema a caso può essere determinato, generando una matrice A casuale di dimensione n i cui elementi sono tra 0 e 1. Si sceglie, ugualmente a caso, il vettore colonna xx di dimensione n e quindi si calcola b=A*xx. n, A e b sono i dati del problema; xx è la soluzione esatta del problema. La matrice di Hilbert si genera utilizzando la funzione hilb( ) predefinita in MatLab.

Analisi Numerica I, a.a. 2004-2005

Docente: M.Gaviano

Esercitazione n.9

Il seguente algoritmo di Gauss Doolitle fattorizza una matrice A nella forma A=LU con L matrice triangolare inferiore avente 1 nella diagonale principale e U triangolare superiore

For k=1,..,n

For i=k,..,n

$$u_{ki} = a_{ki} - \sum_{p=1}^{k-1} l_{kp} u_{pi}$$

end

For i=k+1,..,n

$$l_{ik} = (a_{ik} - \sum_{p=1}^{k-1} l_{ip} u_{pk}) / u_{kk}$$

end

end

Risolvi con MatLab un sistema lineare Ax=b generato a caso, fattorizzando prima A con l'algoritmo su descritto e poi risolvendo i sistemi triangolari. Calcola anche il numero di operazioni aritmetiche (addizioni, sottrazioni, moltiplicazioni e divisioni) eseguite durante l'esecuzione dell'algoritmo.

Suggerimento: Il problema a caso può essere determinato, generando una matrice A casuale di dimensione n i cui elementi sono tra 0 e 1. Si sceglie, ugualmente a caso, il vettore colonna xx di dimensione n e quindi si calcola b=A*xx. n, A e b sono i dati del problema; xx è la soluzione esatta del problema.

Analisi Numerica I, a.a. 2004-2005

Docente: M.Gaviano

Esercitazione n.10

Il seguente algoritmo di Cholesky fattorizza una matrice A, simmetrica definita positiva, nella forma $A=R*R^T$ con R matrice triangolare inferiore.

For k=1,..,n

$$a_{kk} = (a_{kk} - \sum_{p=1}^{k-1} a_{kp}^2)^{0.5}$$

For i=k+1,..,n

$$a_{ik} = (a_{ik} - \sum_{p=1}^{k-1} a_{ip} a_{kp}) / a_{kk}$$

end

end

Risolvi con MatLab un sistema lineare Ax=b generato a caso, con A simmetrica definita positiva, fattorizzando prima A con l'algoritmo su descritto e poi risolvendo i sistemi triangolari..

Suggerimento: Il problema a caso può essere determinato, generando una matrice C casuale di dimensione n i cui elementi sono tra 0 e 1 e ponendo poi $A=C^TC$. Si sceglie, ugualmente a caso, il vettore colonna xx di dimensione n e quindi si calcola b=A*xx. n, A e b sono i dati del problema; xx è la soluzione esatta del problema.

Analisi Numerica I, a.a. 2004-2005

Docente: M.Gaviano

Esercitazione n.11

Calcola il tempo che l'algoritmo di Gauss con pivoting scalato impiega a risolvere sistemi lineari con dimensioni n= 10,20,30, 40, …, 200. Memorizza i tempi ottenuti in un array di nome tempi(), quindi fai il grafico della variabile n e della corrispondente tabulazione tempi()

Suggerimento. Poni l'algoritmo di Gauss dentro un ciclo in cui n varia da 10 a 200 con passo 10 e quindi memorizza i tempi di calcolo. Utilizza le funzioni di Matlab **cputime** che fornisce il tempo dell'orologio in secondi; la differenza tra due chiamate successive di **cputime** fornisce il tempo di utilizzo della cpu tra le due chiamate.

Analisi Numerica I, a.a. 2004-2005

Docente: M.Gaviano

Esercitazione n.12

L'algoritmo di Gauss per la fattorizzazione A=L*U di matrici a banda inferiore p e superiore q è

1. **algoritmo**

```
for k=1,2,…,n-1
    for i=k+1,…,min(k+p,n)
        a_{i,k} = a_{ik} / a_{k,k}
    end
    for i=k+1,…,min(k+p,n)
        for j=k+1,…,min(k+q,n)
            a_{ij} = a_{ij} - a_{ik} a_{kj}
        end
    end
end
```

Scrivi un algoritmo in Matlab che risolva il sistema lineare Ax=b in cui A è una matrice a banda (p,q) determinando prima le matrici U ed L con l'algoritmo su indicato e poi risolvendo Ly=b e Ux=y.

Sperimenta l'algoritmo con n=60, q=p=0, n/4, n/2, n-1. Calcola il tempo di esecuzione per ciascuno di questi parametri mediante la funzione di MatLab *cputime*.

Suggerimento: Il problema a caso può essere determinato, generando prima una matrice A, casuale di dimensione n i cui elementi sono tra 0 e 1, poi utilizzando successivamente le funzioni predefinite tril(A,k) e triu(A,k) (l'help di triu() e tril() indicano quale valore dare a k). Si sceglie, ugualmente a caso, il vettore colonna xx di dimensione n e quindi si calcola b=A*xx. n, A e b sono i dati del problema; xx è la soluzione esatta del problema.

Analisi Numerica I, a.a. 2004-2005

Docente: M.Gaviano

Esercitazione n.13

Risolvi un sistema lineare Ax=b utilizzando sia il metodo iterativo di Jacobi sia il metodo di Gauss-Seidel. Il primo è basato sul seguente schema iterativo

For 1=1,..,n

$$x_i^{(k+1)} = (b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)}) / a_{ii}$$

end

Il secondo sullo schema

For 1=1,..,n

$$x_i^{(k+1)} = (b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)}) / a_{ii}$$

end

Testa i due metodi sia nel caso la matrice A sia data da

$$A = \begin{bmatrix} 1 & -2 & 2 \\ -1 & 1 & -1 \\ -2 & -2 & 1 \end{bmatrix}, \; A = \begin{bmatrix} 2 & -1 & 1 \\ 2 & 2 & 2 \\ -1 & -1 & 2 \end{bmatrix}$$

sia nel caso la matrice A sia generata a caso utilizzando funzioni predefinite di MatLab per costruire un sistema Ax=b per il quale si è sicuri che il metodo di Jacobi converge:

```
B=(1/n)*rand(n);
DD=diag(diag(B));
B=B-DD;
A=DD-DD*(B);
```

Suggerimento. I due metodi sono iterativi; indicando con k l'indice di iterazione si ha che la convergenza è ottenuta per k che tende all'infinito. Pertanto nella formulazione dell'algoritmo è necessario inserire un criterio di stop. Questo può essere: fissata un'accuratezza *acc* termina le iterazioni quando la norma della differenza di due n-ple successive $x^{(k+1)}$, $x^{(k)}$ è minore di *acc*. Inoltre poiché il criterio di stop potrebbe non essere mai soddisfatto (caso della non convergenza), entrando in un ciclo infinito, conviene aggiungere un altro criterio di stop: l'esecuzione dell'algoritmo è interrotta quando viene raggiunta un'iterazione massima prefissata. Nota anche che nell'implementazione non è necessario memorizzare tutta la successione $x^{(k)}$ basta solo operare su due vettori x0 e x1.

Analisi Numerica I, a.a. 2004-2005

Docente: M.Gaviano

Esercitazione n.14

L'algoritmo di Neville data la tabulazione di una funzione x_i, y_i i= 0,..n permette di calcolare il valore del polinomio interpolante fxs in un punto generico xs. Memorizzando i valori di x_i e y_i negli arrays x(i) e y(i) l'algoritmo può scriversi nella forma

```
t(0)=y(0)
for i=1,…,n
    t(i)=y(i)
    for j=i-1,-1,0
        t(j)=t(j+1)+(t(j+1)-t(j))*(xs-x(i))/(x(i)-x(j))
    end
end
fxs=t(0).
```

Scrivi un algoritmo in matlab che dato un intervallo [a,b]

- scelga per una assegnata funzione f(x), n nodi x_i equidistanti nell'intervallo dato e calcoli i valori $y_i=f(x_i)$.
- con l'algoritmo di Neville calcoli i valori pxs_j del polinomio interpolante in m punti xs_j equidistanti nell'intervallo [a,b].
- calcoli i valori di f(x) nei punti xs_j da indicare con fxs_j.
- Usando i valori xs_j e pxs_j e fxs_j faccia i grafici del polinomio interpolante e della funzione f(x).

Come funzioni test f(x) prendi 2sen(x)+2 , e $1/(1+x^2)$

Hint: Scegli n come una variabile input in questo modo si possono facilmente fare prove con un numero di nodi variabile. Usa la definizione di funzione presente in Matlab per creare il file fun.m contenente la definizione di funzione. Per esempio:

```
function y = fun(x)
                    % calcolo di y
y=2*sin(x)+2;
```

Analisi Numerica I, a.a. 2004-2004

Docente: M.Gaviano

Esercitazione n.15

Data la tabulazione di una funzione x_i, y_i i= 0,..n, MatLab permette di costruire il polinomio interpolante di grado n. Ciò si ottiene mediante la funzione *polyfit*(x,y,n) con x≡(x_i), y≡(y_i) e n il grado del polinomio (*polyfit* si trova nel gruppo polyfun). *Polyfit* restituisce un vettore, che possiamo chiamare P i cui elementi sono i coefficienti del polinomio interpolante. Inoltre in matlab esiste, predefinita, la funzione *polyval*(P,z) la quale dati i coefficienti P di un polinomio dà una tabulazione del polinomio in z, con z vettore di numeri.

Scrivi un algoritmo in matlab che dato un intervallo [a,b]

- scelga per una assegnata funzione f(x), n nodi x_i equidistanti nell'intervallo dato e calcoli i valori y_i=f(x_i).
- mediante la funzione *polyfit* calcoli i coefficienti del polinomio interpolante nei punti x_j y_i.
- faccia il grafico di f(x) e del polinomio interpolante scegliendo m punti xs_j equidistanti nell'intervallo [a,b].

Come funzioni test f(x) prendi

- e^{-x}sen(2πx) nell'intervallo [0,4]
- (-3x+1.4)*sen(18x) nell'intervallo [0.96609, 1.48907].
- 1/(1+x^2) nell'intervallo [-5, 5].

Hint: Scegli n come una variabile input in questo modo si possono facilmente fare prove con un numero di nodi variabile. Usa la definizione di funzione presente in Matlab per creare il file fun.m contenente la definizione di funzione. Per esempio:

```
function y = fun(x)
                    % calcolo di y
y=2*sin(x)+2;
```

Analisi Numerica I, a.a. 2004-2005

Docente: M.Gaviano

Esercitazione n.16

Data la tabulazione di una funzione f(x) ,x_i, y_i i= 0,..n, la f(x) può approssimarsi mediante una spline lineare. Scrivi un algoritmo in matlab che dato un intervallo [a,b]

- scelga per una assegnata funzione f(x) n nodi x_i equidistanti nell'intervallo dato e calcoli i valori $y_i=f(x_i)$.
- faccia i grafici della spline interpolante e della funzione f(x).

Come esempio funzione prendi le funzioni

- 2sen(x)+2 (nell'intervallo [-20, 20]) ,
- e^{-x}sen(2πx) nell'intervallo [0,4]
- (-3x+1.4)*sen(18x) nell'intervallo [0.96609, 1.48907].
- $1/(1+x^2)$ (nell'intervallo [-6, 6])

Hint: n deve essere una variabile input in questo modo si possono facilmente fare prove con diversi nodi. Usa la definizione di funzione presente in Matlab:

Crea il file fun.m contenente

% definizione di funzione

```
function y = fun(x)
y=2*sin(x)+2;
```

la scrittura fun(6.0) in un programma o nella finestra di commando permette di calcolare 2sen(x)+2 nel punto x=6.0. Attento che alla funzione può passarsi uno scalare o un vettore. In quest'ultimo caso la funzione restituisce il vettore tabulazione, ma la definizione deve essere fatta opportunamente.

Analisi Numerica I, a.a. 2004-2005

Docente: M.Gaviano

Esercitazione n.17

1) Supponi di conoscere la tabulazione x_i, y_i, i=1,..,m , di una funzione f(x) in un intervallo [a,b]. Utilizzando Matlab costruisci un polinomio di grado n, n<m che approssima la f(x) nel senso dei minimi quadrati (caso discreto). Fai il grafico della f(x) (per punti) e del polinomio approssimante.
2) Ripeti la stessa procedura utilizzata per risolvere 1) supponendo che i valori di f(x) non siano calcolati esattamente ma con un errore casuale che non supera il 10% del valore della funzione. Cioè invece di f(x) si ha g(x)=f(x)+ casuale(x) con f(x) –0.1*|f(x)|<casuale(x)< f(x) +0.1*|f(x)|.

Considera come esempi di funzioni

- 2sen(x)+2 (nell'intervallo [-20, 20]) ,
- e^{-x}sen(2πx) nell'intervallo [0,4]
- (-3x+1.4)*sen(18x) nell'intervallo [0.96609, 1.48907].
- $1/(1+x^2)$ (nell'intervallo [-6, 6])

Hint: Approssimare nel senso dei minimi quadrati (caso discreto) con un polinomio di grado n significa:

Data la tabulazione x_i, y_i, i=1,..,m , di una funzione f(x)εC[a,b] e il sottospazio W dei polinomi di grado n, $w(a,x)= a_0+a_1x+\ldots+a_nx^n$, trovare il polinomio w(a*,x) tale che

$$\sum_{k=1}^{m}(w(a^*,x_k)-y_i)^2 \le \sum_{k=1}^{m}(w(a,x_k)-y_i)^2$$

Il vettore $a^* = (a_0^*, a_1^*, ..., a_n^*)$ è dato da

$$Ma^*=r$$

Con

$$M = (m_{i,j}) = \sum_{k=1}^{m} x_k^i x_k^j \quad \text{i=0,1,…,n e j=0,1,…,n}$$

$$r = (r_i) = \sum_{k=1}^{m} x_k^i y_k \quad \text{i=0,1,…,n}$$

Analisi Numerica I, a.a. 2004-2005

Docente: M.Gaviano

Esercitazione n.18

3) Supponi di conoscere la tabulazione x_i, y_i, i=1,..,m , di una funzione f(x) in un intervallo [a,b]. Dato un intero n, $n \geq 0$, utilizzando la funzione predefinita di Matlab *polyfit*(x,y,n) (data la tabulazione $x \equiv (x_i)$, $y \equiv (y_i)$ ed n il grado del polinomio, *Polyfit* restituisce un vettore, che possiamo chiamare P i cui elementi sono i coefficienti del polinomio interpolante) costruisci un polinomio di grado n, che approssima la f(x) nel senso dei minimi quadrati (caso discreto). Fai il grafico della f(x) (per punti) e del polinomio approssimante.
4) Ripeti la stessa procedura utilizzata per risolvere 1) supponendo che i valori di f(x) non siano calcolati esattamente ma con un errore casuale che non supera il 10% del valore della funzione. Cioè invece di f(x) si ha g(x)=f(x)+ casuale(x) con f(x) –0.1*|f(x)|<casuale(x)< f(x) +0.1*|f(x)|.

Considera come esempi di funzioni

- 2sen(x)+2 (nell'intervallo [-20, 20]) ,
- e^{-x}sen(2πx) nell'intervallo [0,4]
- (-3x+1.4)*sen(18x) nell'intervallo [0.96609, 1.48907].
- $1/(1+x^2)$ (nell'intervallo [-6, 6])

Hint: per ottenere la tabulazione del polinomio interpolante utilizza la funzione *polyval*(P,z) la quale dati i coefficienti P di un polinomio dà una tabulazione del polinomio in z, con z scalare o vettore di numeri.