CALCOLO NUMERICO

(C.L. Ing. delle Comunicazioni - AA 2013-14)

Proff. F. Pitolli, A. Pascarella

Progetti

1. Il procedimento iterativo

$$P_i^{(k+1)} = \sum_{j\in\mathbb{Z}} a_{i-2\,j}\, P_j^{(k)}\,, \qquad k = 0, 1, \ldots$$

dove $\{P_i^{(k)}, i \in \mathbb{Z}\}$ è una sequenza di punti in $\mathbb{R}^2$ o $\mathbb{R}^3$ e $\{a_i\}$ è una sequenza di numeri reali data, è un metodo efficiente per costruire curve regolari che approssimano la sequenza iniziale $\{P_i^{(0)}\}$.

Eseguire 8 passi degli algoritmi

$$\text{Chaikin's algorithm: } \begin{cases} P_{2i}^{(k+1)} = \dfrac{3}{4}\, P_i^{(k)} + \dfrac{1}{4}\, P_{i+1}^{(k)} \\ \\ P_{2i+1}^{(k+1)} = \dfrac{1}{4}\, P_i^{(k)} + \dfrac{3}{4}\, P_{i+1}^{(k)} \end{cases} \qquad k = 0, 1, \ldots$$

$$\text{4-point scheme: } \begin{cases} P_{2i}^{(k+1)} = P_{2i} \\ \\ P_{2i+1}^{(k+1)} = -\omega\, P_{i-1}^{(k)} + \left(\dfrac{1}{2} + \omega\right) P_i^{(k)} + \left(\dfrac{1}{2} + \omega\right) P_{i+1}^{(k)} - \omega\, P_{i-1}^{(k)} \end{cases} \qquad k = 0, 1, \ldots$$

con $0 \le \omega \le \frac{\sqrt{5}-1}{8}$, a partire dalla sequenza iniziale

$$\begin{array}{lllll} P_0 = (5.45, 4.31) & P_1 = (4.30, 4.40) & P_2 = (4.48, 5.00) & P_3 = (4.00, 5.50) & P_4 = (3.52, 5.00) \\ P_5 = (3.71, 4.40) & P_6 = (2.55, 4.31) & P_7 = (3.52, 3.85) & P_8 = (3.16, 1.94) & \\ P_9 = (4.00, 3.50) & P_{10} = (4.84, 1.94) & P_{11} = (4.48, 3.85) & P_{12} = (5.45, 4.31) & \end{array}$$

Utilizzare condizioni periodiche per gli estremi della sequenza. Confrontare le curve ottenute con i due algoritmi dati con quella che si ottiene con la function di Matlab `spline`. Per il 4-point scheme usare, ad esempio, i valori $\omega = 0, 1/64, 1/32, 1/16$.

Il procedimento iterativo dato è noto come *algoritmo di suddivisione binario*. Gli algoritmi di suddivisione (subdivision schemes) sono utilizzati nella computer graphics per costruire superfici in $\mathbb{R}^3$ a partire da un insieme di punti assegnato (vedi `http://graphics.pixar.com/opensubdiv`).

2. Scrivere una function che implementi il prodotto matrice-vettore $A\,X$ con A matrice sparsa. Si memorizzi la matrice sparsa utilizzando tre vettori:

   - il vettore `A` contenente solo gli elementi non nulli della matrice A;
   - il vettore `Icol` contenente gli indici di colonna degli elementi del vettore `A`;
   - il vettore `Irow` contenente l'indice del vettore `A` dove inizia una nuova riga della matrice A.

   Esempio: per la matrice $A = \begin{bmatrix} 0 & 1 & -2 & 0 \\ -2 & 0 & 0 & 3 \\ -1 & 0 & 0 & 0 \\ 2 & 0 & -2 & 0 \end{bmatrix}$

   si ha

   `A = [1,-2,-2,3,-1,2,-2]  Icol = [2,3,1,4,1,1,3]  Irow = [1,3,5,6]`

   Utilizzare la function per implementare i metodi iterativi di Jacobi e Gauss-Seidel.

   Risolvere il sistema lineare $A\,X = B$ dove A è una matrice tridiagonale a blocchi

$$A = \begin{bmatrix} C & D & 0 & 0 & \cdots \\ D & C & D & 0 & \cdots \\ 0 & D & C & D & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & 0 & D & C \end{bmatrix} \quad C = \begin{bmatrix} 4 & -1 & 0 & \cdots \\ -1 & 4 & -1 & \cdots \\ 0 & -1 & 4 & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & -1 & 4 \end{bmatrix} \quad D = \begin{bmatrix} -1 & 0 & 0 & \cdots \\ 0 & -1 & 0 & \cdots \\ 0 & 0 & -1 & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & 0 & -1 \end{bmatrix}$$

   e $B = [1110 \cdots 0111]^T$.

   Confrontare la velocità e l'occupazione di memoria dei metodi implementati con quella del solutore di Matlab.

   Il sistema lineare dato è il sistema risultante dalla discretizzazione dell'equazione di Poisson con il metodo alle differenze finite.

3. Il problema differenziale non lineare

$$\begin{cases} y' = -\beta\, y + z\, w \\ z' = -\sigma\, z + \sigma\, w \\ w' = -z\, y + \rho\, z - w \end{cases}$$

rappresenta un sistema dinamico, noto come *attrattore di Lorentz*, la cui soluzione non può essere calcolata per via analitica.

Si assegnino ai parametri i valori

$$\sigma = 10\,, \qquad \rho = 28\,, \qquad \beta = 8/3\,, \qquad \eta = \sqrt{\beta(\rho - 1)}$$

e le condizioni iniziali

$$x_0 = 0, \quad y_0 = \rho - 1, \quad z_0 = \eta, \quad w_0 = \eta + 3.$$

Approssimare la soluzione con i metodi di Eulero, Heun e Runge-Kutta con vari passi di integrazione. Provare diverse condizioni iniziali e diversi valori del parametro ρ. Graficare la soluzione nello spazio delle fasi. Che tipo di comportamento ha? E' possibile individuare uno stato stazionario?

4. Il problema differenziale lineare, elaborato dal matematico Steven Strogatz negli anni Ottanta,

$$\begin{cases} R' = \alpha\, R + \beta\, J \\ J' = \sigma\, R + \omega\, J \end{cases}$$

descrive come varia nel tempo l'atteggiamento amoroso di Romeo, espresso da $R(t)$ nei confronti di Giulietta, $J(t)$, e viceversa.

I parametri α, β, σ, γ, possono avere entrambi i segni. Il parametro α descrive la misura con cui Romeo è incoraggiato dal suo stesso sentimento, mentre β è la misura in cui egli è incoraggiato dai sentimenti di Giulietta. Analogo significato hanno i parametri σ e ω.

Approssimare la soluzione del problema differenziale con i metodi di Eulero, Heun e Runge-Kutta con vari passi di integrazione. Provare diverse condizioni iniziali e diversi valori dei parametri. Graficare la soluzione nello spazio delle fasi. Che tipo di comportamento ha la soluzione al variare dei parametri? E' possibile individuare uno stato stazionario?

5. Scrivere una funzione MatLab che implementi l'algoritmo di Canny per la individuazione di contorni in un'immagine. La funzione riceva in input una matrice di numeri *I* e tre numeri reali e positivi: *sigma*, t_1 e t_2, con $t_2<t_1$, Se t_2 non viene data in input ponga $t_2=t_1/2$; se *sigma* non viene data in input ponga *sigma* = 1.

La funzione

- esegua il prodotto di convoluzione tra *I* e un filtro bidimensionale gaussiano con deviazione standard *sigma* (consultare le funzioni fspecial e conv2 di MatLab);
- indicando con *(i,j)* rispettivamente gli indici di riga e di colonna di ogni elemento di *I*, calcoli le componenti x e y del gradiente di I, rispettivamente Ix e Iy, in corrispondenza di ogni punto *(i,j)* ;
- calcoli il valore assoluto *M(i,j)* e la direzione *D(i,j)* (angolo) del gradiente di *I* ;
- approssimi *D(i,j)* con il valore più prossimo appartenente all'insieme {0, π/4, π/2, 3π/4};
- se *M(i,j)* non è un massimo locale lungo la direzione perpendicolare a *D(i,j)*, ponga *M(i,j)=0;*
- per ogni punto *(i,j)* tale che $M(i,j)\geq t_1$ determini tutti i punti *(ic,jc)* connessi a *(i,j)* lungo la direzione *D(i,j)* e tali che $t_2 \leq M(ic,jc) < t_1$*;*
- costruisca la matrice *A* così definita:
  - *A(i,j)=1* se $M(i,j) \geq t_1$ oppure se *(i,j)=(ic,jc)*, con *(ic,jc)* determinati al punto precedente,
  - *A(i,j)=0* altrimenti;
- restituisca in output la matrice *A*
- visualizzi la matrice *A* usando il comando imshow di MatLab

Testare la funzione su matrici *I* corrispondenti a immagini a livelli di grigio (usare la funzione Matlab imread.m per leggere l' immagine. Per esempio: I = double(imread('cameraman.tif'));) variando i parametri di input. Confrontare i risultati con quelli ottenuti usando la funzione edge di MatLab. Scrivere una breve relazione che includa i risultati più significativi.

6. Dato il file data_meg.mat si scriva una funzione Matlab che risolva il problema inverso della MagnetoEncefaloGrafia (MEG)

$$B = G*S$$

dove B è la matrice dei dati di dimensione n_sensori x n_istanti, G è la cosiddetta matrice di guadagno di dimensione n_sensori x n_punti e S è la matrice dell'intensità di corrente di dimensione n_punti x n_istanti. Il problema inverso MEG consiste nel ricostruire la matrice S a partire dalle misure note del campo magnetico B.

Il file data_meg.mat contiene le seguenti variabili:

- **data** è la matrice dei dati B
- **LF_soc** è la cosiddetta matrice di guadagno G
- **source_space** è lo spazio delle sorgenti P
- **times** è il vettore dei tempi

La funzione deve avere come **parametri di input**: la matrice dei dati B, la matrice di guadagno G, lo spazio delle sorgenti P, un istante T. La funzione deve restituire come **output** la matrice S dell'intensità di corrente all'istante T.
In particolare la funzione deve:

1) visualizzare la matrice delle misure di campo magnetico B (i valori delle ascisse sono gli istanti temporali contenuti in times)
2) risolvere il sistema B=G*S mediante l'uso dei minimi quadrati
3) rappresentare la soluzione S all'istante T (per la visualizzazione del risultato usare lo spazio delle sorgenti P)
4) salvare il grafico ottenuto al punto 3) in un file.

Abbellire i grafici con le dovute etichette e commentare la funzione.

Scrivere una breve relazione in cui si spieghi l'algoritmo implementato e in cui si riportino i grafici ottenuti. Includere una breve descrizione sulla MagnetoEncefaloGrafia .

7. Dato il file meg_data_raw.mat si scriva una funzione matlab che implementi un algoritmo che individui i trials rumorosi.
Tale funzione deve avere come **parametro di input**: una stringa data_filename (che contiene il nome del file dei dati da importare) e deve restituire come **output** il vettore trials_rumorosi contenente l'indice dei trials rumorosi e la matrice B_m dei dati ottenuta mediando solo i dati relativi ai trials non rumorosi.
In particolare la funzione deve:

1) importare i file data_filename e memorizzare il suo contenuto nella variabile Braw che rappresenta la matrice dei dati di dimensione n_sensori x n_istanti x n_trials: fissato il terzo indice, la riga i-esima contiene il segnale registrato dall'i-esimo sensore (le colonne rappresentano gli istanti temporali in cui è stato registrato il segnale);
2) determinare le dimensioni della matrice dei dati Braw e stamparle a video
3) disegnare la matrice dei dati Braw rispetto al tempo per diversi valori del terzo indice (che rappresenta i vari trials dell'esperimento)
4) costruire il vettore dei trials_rumorosi a partire da Braw: un trial è definito come rumoroso se la matrice dei dati relativa a tale trial contiene valori maggiori di una certa soglia (=0.6)
5) calcolare la nuova matrice dei dati B_m che si ottiene mediando il segnale registrato da ogni sensore considerando solo i trials non rumorosi
6) disegnare in una stessa finestra grafica la nuova matrice dei dati B_m così ottenuta e la matrice ottenuta mediando su tutti i trials
7) salvare il grafico ottenuto al punto 6) in un file.

Abbellire i grafici con le dovute etichette e commentare la funzione.

Scrivere una breve relazione in cui si spieghi l'algoritmo implementato e in cui si riportino i grafici della matrice dei dati mediati B. Includere una breve descrizione sulla MagnetoEncefaloGrafia (la tecnica con la quale sono stati registrati questi dati).

8. Dato il file clown.bmp si scriva una funzione matlab che abbia come **parametri di input**: una striga file_name (che contiene il nome dell'immagine da importare) e un valore di soglia T. La funzione deve

1) importare il file file_name dell'immagine, e memorizzare il suo contenuto in una matrice I
2) determinare le dimensione dell'immagine I importata
3) disegnare l'immagine usando le apposite funzioni di Matlab
4) disegnare l'istogramma della scala dei grigi dell'immagine
5) definire una nuova immagine I_T contenente solo i pixel i cui valori sono superiori alla soglia T
6) disegnare in una stessa finestra grafica l'immagine originale I e quella sogliata I_T
7) scrivere la nuova immagine I_T in un nuovo file

Abbellire i grafici con le dovute etichette e commentare la funzione.

Scrivere una breve relazione in cui si spieghi l'algoritmo implementato e in cui si riportino i grafici dell'immagine originale I e della sua versione sogliata I_T per diversi valori di T.

9. Sia dato un bersaglio formato da un quadrato di lato 2L contenente una circonferenza C di raggio L. Detto N il numero di freccette lanciate al bersaglio, si supponga che le freccette siano lanciate casualmente all'interno del bersaglio e che quindi colpiscano il quadrato in ogni posizione con uguale probabilità.
Dopo molti lanci la frazione di freccette Nc che ha colpito il cerchio sarà proporzionale al rapporto tra l'area della circonferenza e quella del quadrato

$$\frac{\pi L^2}{4L^2} = \frac{1}{4}\pi \cong \frac{N_c}{N}$$

e può essere usato per stimare il valore di pi greco.

Scrivere una funzione Matlab che calcoli il valore di pi greco col metodo Monte Carlo. In particolare la funzione deve avere come **parametri di input** il numero N_r dei lanci, il valore L del lato del bersaglio e deve restituire come **output** il vettore pi_greco contenente la stima di π ottenuta ad ogni lancio. In particolare, la funzione deve

1) disegnare in uno stesso grafico il valore di π ottenuto all'i-esimo lancio
2) calcolare il valore medio delle stime ottenute
3) ripetere l'esperimento per diversi valori di N
4) memorizzare i risultati ottenuti al punto 3) in un file

Abbellire i grafici con le dovute etichette e commentare la funzione.

Scrivere una breve relazione in cui si spieghi l'algoritmo implementato.

10. Dato il file mri.mat si scriva una funzione matlab che abbia come **parametro di input** una stringa file_name (che contiene il nome del file da importare) e un valore di soglia T e come **parametro di output** la matrice contenente i valori della correlazione tra coppie di immagini.

La funzione deve

1. importare il file file_name e memorizzare il suo contenuto in una matrice D
2. visualizzare le dimensioni della matrice D
3. rappresentare in un'unica finestra grafica le varie immagini contenute in D
4. calcolare la correlazione tra le immagini contenute in D; in particolare, detto N il numero delle immagini contenute in D, la matrice di correlazione C avrà dimensione NxN e sarà tale che C(i,j) è la correlazione tra l'immagine i e l'immagine j
5. rappresentare in figure separate le coppie di immagini la cui correlazione è maggiore di T

Abbellire i grafici con le dovute etichette e commentare la funzione.

Scrivere una breve relazione in cui si spieghi l'algoritmo implementato.

11. L'orbita ellittica di un pianeta è descritta dall'equazione:

$$a\,x^2 + b\,x\,y + c\,y^2 + d\,x + e\,y + f = 0$$

Si consideri la tabella

x_i	1.02	0.95	0.87	0.77	0.67	0.56	0.44	0.30	0.16	0.01
y_i	0.39	0.32	0.27	0.22	0.18	0.15	0.13	0.12	0.13	0.15;

dove sono riportate 10 osservazioni relative alle posizioni di un pianeta.

Dividere l'equazione per uno qualunque dei coefficienti e calcolare l'ellisse che approssima i dati nel senso dei minimi quadrati. Graficare l'orbita approssimata sovrapposta ai dati della tabella.

Perturbare i dati aggiungendo ad ogni punto un numero random distribuito uniformemente nell'intervallo $[0.0005, 0.0005]$ e calcolare il nuovo polinomio. Sovrapporre anche questa approssimazione alle precedenti.

Spiegare il comportamento delle approssimazioni ottenute.

12. Dato il file contrabbasso.m4a si scriva una funzione matlab che abbia come **parametro di input** una stringa file_name (che contiene il nome del file da importare), due interi t1,t2 e come **parametri di output il** vettore y contenente il segnale campionato nella finestra temporale [t1, t2] e la trasformata di Fourier Y del segnale y effettuata sulla finestra temporale [t1, t2]. In particolare, la funzione deve:

- importare il file file_name usando le opportune funzioni di Matlab
- memorrizzare in una variabile y il segnale di uno dei due canali
- visualizzare y nella finestra temporale [t1,t2]
- calcolare la trasformata di Fourier Y del segnale y nella finestra temporale [t1,t2]
- disegnare in uno stesso grafico il segnale nel dominio dei tempi e in quello delle frequenze
- salvare il grafico ottenuto nel punto precendente

Richiamare la funzione per diverse intervalli temporali: un secondo, due secondi, dieci secondi, un minuto, tutta la durata del file.

Riprodurre il file usando le apposite funzioni di Matlab.

Abbellire i grafici con le dovute etichette e commentare la funzione.

Scrivere una breve relazione in cui si spieghi l'algoritmo implementato.

13. Dato il file flute.wav, si scriva una funzione matlab che abbia come **parametro di input** una stringa file_name (che contiene il nome del file da importare), un intero N e una stringa (opzionale) play che può assumere i valori 'y' o 'r' e come **parametri di output il** vettore y contenente il segnale digitale, il vettore ye contenente l'eco del segnale e la frequenza di campionamento fs. In particolare, la funzione deve:

- importare il file file_name usando le opportune funzioni di Matlab
- visualizzare il numero di campioni, la frequenza di campionamento del segnale digitale, la durata del segnale
- disegnare il segnale nel dominio del tempo (sec) per diverse finestre temporali
- se il valore della stringa opzionale play è 'y' riprodurre il segnale, se è 'r' riprodurlo all'indietro
- aggiungere un eco al segnale aggiungendo ad ogni campione il campione di un tempo precedente N: si crea così il nuovo segnale ye

Richiamare mediante uno script la funzione per diversi valori di N e riprodurre nello script sia il segnale y che il suo eco ye.

Effettuare i controlli sulle variabili di input (usare varargin per gestire il numero di input opzionali)

Scrivere una breve relazione in cui si spieghi l'algoritmo implementato.

14. Si scriva una funzione matlab che abbia come **parametro di input** un vettore x e un intero H e come **parametri di output** il vettore AC contenente la funzione di autocorrelazione del segnale x. In particolare, la funzione deve:

- visualizzare il numero di campioni del segnale e
- disegnare il segnale campionato usando la funzione stem
- calcolare la funzione di autocorrelazione per H lag (NON usare la funzione predefinita di matlab!)
- disegnare la funzione di autocorrelazione di x

Dato il segnale tempo discreto

$$x(n) = \begin{cases} |n| & |n| \leq 4 \\ 4 & 5 \leq n \leq 8 \\ 0 & \text{altrimenti} \end{cases}$$

si scriva uno script in cui si definisca il nuovo segnale y:

$$y(n) = x[n/2]x(n-1)$$

e si usi la funzione precedentemente implementata per calcolare la funzione di autocorrelazione di y. Si confrontino i risultati ottenuti con quelli della funzione predefinita di matlab.

Effettuare i controlli sulle variabili di input e abbellire opportunatamente i grafici.

Scrivere una breve relazione in cui si spieghi l'algoritmo implementato.

15. Dato il file tre_idrometri.dat si scriva una funzione matlab che abbia come **parametro di input** una stringa file_name (che contiene il nome del file da importare) e un intero NL e come **parametro di output la** matrice di crosscorrelazione C.

La funzione deve

1. leggere (con le appropriate funzioni di Matlab) il file file_name e memorizzare il suo contenuto in una matrice T (ogni colonna della matrice rappresenta i valori misurati da un idrometro)
2. rappresentare in un'unica finestra grafica le tre serie temporali
3. calcolare la crosscorrelazione (usando la relativa funzione di Matlab) tra le diverse coppie di serie temporali su NL lag
4. per ogni coppia di serie temporali calcolare il valore massimo della crosscorrelazione e il relativo lag, memorizzare tali valori nella matrice C, disegnare la coppia di serie temporali in uno stesso grafico e stampare sul grafico il valore massimo della crosscorrelazione e il lag al quale si ottiene
5. salvare i grafici del punto 4 in un file

Effettuare i controlli sui paramentri di input.

Abbellire i grafici con le dovute etichette e commentare la funzione.

Scrivere una breve relazione in cui si spieghi l'algoritmo implementato.