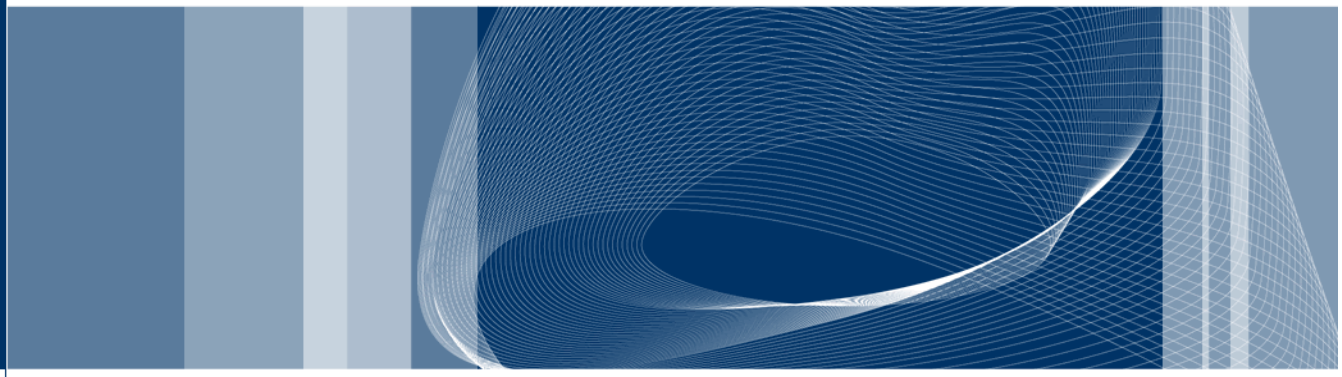
POLITECNICO DI MILANO



Matlab: Gestione avanzata dei file

Informatica B

File

- Contenitori di informazione permanenti
- Sono memorizzati su memoria di massa
- Possono continuare ad esistere indipendentemente dalla vita del programma che li ha creati
- Possono essere acceduti da più programmi
- Il sistema operativo si occupa della loro gestione e offre ai programmi una serie di operazioni per
  - creazione/cancellazione di file
  - scrittura/lettura
  - controllo dei casi di errore
- Nei linguaggi di programmazione abbiamo a disposizione meccanismi per usare I file attraverso il sistema operativo ed ottenere

textread

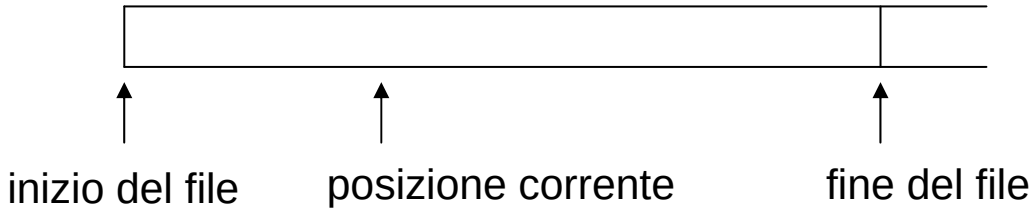
- Legge file ASCII organizzati in tabelle
- Ogni colonna della tabella può essere di tipo diverso (questo non è possibile con load)
- Funzione molto utile per importare dati generati con altre applicazioni
- Forma della funzione
  - [a b c …] = textread (filename, format, n)
  - filename: nome del file da leggere
  - format: come fprintf
  - n: numero di righe da leggere. Se omesso, textread legge fino alla fine del file
  - a, b, c, … vettori colonna in cui verranno caricati i dati

textread: esempio

- Si consideri il file chiamato test_input.dat contenente i seguenti dati
  - James Jones 0+ 3.51 22 yes
  - Sally Smith A+ 3.28 23 No
- Questi dati possono essere letti con l'istruzione
  - [nome cognome gruppo gpa eta risposta] = textread('test_input.dat', '%s %s %s %f %d %s');
- Se si desidera saltare una colonna, per esempio, quella dell'età
  - [nome cognome gruppo gpa risposta] = textread('test_input.dat', '%s %s %s %f %*d %s');

Operazioni di basso livello per l'accesso ai file: approccio operativo

- Simile ad un nastro video/audio



- La posizione corrente indica il punto da dove si leggerà (dove si scriverà)
- L'indicatore di posizione corrente si sposta in avanti dopo ogni lettura/scrittura
- Esistono le operazioni per spostare la posizione corrente in avanti o indietro senza leggere o scrivere (per esempio, rewind)
- Il file non ha dimensioni prefissate. La sua dimensione massima dipende dalla dimensione della memoria di massa

Descrittore del file

- Per ogni file aperto il sistema operativo gestisce un *descrittore di file*
- Il descrittore contiene informazioni su
  - Modalità di utilizzo di un file (lettura, scrittura, append...)
  - Posizione corrente nel file
  - Verificarsi di un errore di lettura/scrittura
  - Coincidenza tra posizione corrente e l'indicatore di fine file
- Tutti i descrittori di file sono memorizzati nella tabella dei file aperti

Sequenza di operazioni di basso livello da eseguire per usare i file

- Apertura di un "flusso di comunicazione" con il file
  - *fid = fopen(nome file, modalità di apertura)*
  - fid è un intero
- Scrittura/lettura nel/dal file
  - *fwrite, fprintf, fread, fscanf*
- Chiusura del flusso di comunicazione
  - *status = fclose(fid)*

fopen

fid = fopen(nome file, modalità di apertura)

- Apre un flusso di comunicazione con il file il cui nome viene specificato come parametro
- Il nome del file può includere il percorso nell'albero delle directory.
  - Se non lo include, si assume che il file si trovi nella directory corrente
  - Si possono specificare anche percorsi relativi rispetto alla directory corrente
- Se il file non esiste e la modalità di apertura è "w" il file viene creato
- Restituisce -1 ed un messaggio di errore se il flusso di comunicazione non è stato aperto, e cioè:
  - Se un file che deve essere aperto in lettura non esiste
  - Se si verifica un errore nell'interazione con il supporto di memorizzazione su cui il file risiede
- fopen('all') restituisce un vettore riga che contiene gli ID di tutti i file aperti dal programma

Modalità di apertura del flusso di comunicazione

- “r”: apre un file esistente in lettura
- “w”: apre un file esistente o crea un nuovo file in scrittura con distruzione di quanto già presente nel file
- “a”: apre un file esistente o crea un nuovo file in scrittura con posizionamento alla fine del file. Quanto già presente nel file viene mantenuto
- “rt”: come “r” ma in modalità testuale
- “wt”: come “w” ma in modalità testuale
- “at”: come “a” ma in modalità testuale
- “r+” e ”rt+”: come “r”/”rt” ma si può anche scrivere nel file
- “w+” e “wt+”: come “w”/”wt” ma si può anche leggere dal file
- “a+” e “at+”: come “a”/”at” ma si può anche leggere dal file

fclose

status = fclose(fid)

- Determina la chiusura del flusso di comunicazione con il file identificato da fid
- Restituisce 0 se la chiusura è avvenuta senza errori, il valore -1 in caso di problemi

status=fclose('all')

- chiude tutti i file

fwrite

- cont=fwrite(fid,array,formato)
  - cont: indica il numero di valori effettivamente scritti nel file
  - fid: identificatore del file su cui scrivere (nb: il file deve essere stato aperto in precedenza)
  - array: array contenente i dati da salvare
  - formato: specifica il formato in cui i dati verranno salvati. Formati principali
    - char, int8, int16, int32, int64, float32, float64 (i numeri indicano il numero di bit usati per rappresentare i valori)

fread

- [array cont]=fread(fid,size,formato)
  - Per il significato di cont e fid, e formato si veda la slide precedente
  - I dati letti vengono memorizzati in array
  - size è la dimensione dei dati da leggere. Tre possibilità
    - n: legge esattamente n valori. Dopo l'esecuzione di questa istruzione array sarà un vettore colonna contenente questi n valori
    - Inf: legge fino alla finel del file. Dopo l'esecuzione di questa istruzione array sarà un vettore colonna contenente tutti i valori letti
    - [n m]: legge esattamente nxm valori. Dopo l'esecuzione di questa istruzione array sarà una matrice nxm contenente tutti i valori letti

Esempio: salvataggio dati su file

```
% genera un vettore riga contenente numeri casuali
a=rand(1,1000);
%richiede all'utente il nome del file
filename=input('inserisci un nome di file ');
%apre il file
[fid msg]=fopen(filename, 'w');
%se il file e` stato aperto con successo...
if(fid>0)
    %scrive il vettore a su file
    cont=fwrite(fid,a,'float64');
    %informa l'utente dell'avvenuta scrittura
    disp([num2str(cont) ' valori scritti...']);
    %chiude il file
    fclose(fid);
else %il file non e` stato aperto...
    disp(msg);
end
```

Esempio: caricamento dati da file

```
%richiede all'utente il nome del file
filename=input('inserisci un nome di file ');
%apre il file
[fid msg]=fopen(filename, 'r');
%se il file e` stato aperto con successo...
if(fid>0)
    %legge i dati da file e li memorizza in un vettore
    [vett cont]=fread(fid,[1 1000],'float64');
    %informa l'utente dell'avvenuta lettura
    disp([num2str(cont) ' valori letti...']);
    %chiude il file
    fclose(fid);
else %il file non e` stato aperto...
    disp(msg);
end
```

Input/output formattato

- cont=fprintf(fid,format,val1, val2, ….)
  - A parte fid, è come la printf in C
- [array cont] = fscanf(fid, format, size)
  - I dati letti vengono memorizzati in array
  - size come in fread

Esempio

- Si consideri il file x.dat che contiene i seguenti dati

  10.00 20.00

  30.00 40.00

- Come utilizzare la scanf per costruire diverse strutture dati con questi valori
- [z cont] = fscanf(fid, '%f');

$$\longrightarrow \begin{bmatrix} 10 \\ 20 \\ 30 \\ 40 \end{bmatrix}$$

cont 4

- [z cont] = fscanf(fid, '%f', [2 2]);

$$\longrightarrow \begin{bmatrix} 10 & 20 \\ 30 & 40 \end{bmatrix}$$

cont 4