

Esame di Laboratorio di Fisica Computazionale

6 maggio 2014, ore 10.00

shell scripting

1. Si utilizzi `sed` per modificare il file `modello.txt`
 - 1) cancellando le prime tre righe che contengono degli asterischi,
 - 2) sostituendo le scritte generiche `cognome` e `nome` con il vostro nome e cognome. Si salvi il risultato in un nuovo file.

Mathematica

1. Si risolva, rispetto alle variabili (y, z) , il seguente sistema di equazioni:

$$\begin{aligned}x^2 + y^2 - z &= 1 \\ z - 2x^2 - 2y^2 &= 4\end{aligned}\tag{1}$$

2. Si disegnino, nello stesso grafico, le due superfici definite dalle due equazioni (??).

3. Si risolva la seguente equazione differenziale, parametrica in λ :

$$\begin{aligned}y'(x) + y(x) &= e^{-\lambda x} + 3 \sin x \\ y(1) &= 2\end{aligned}\tag{2}$$

4. Si disegni in un grafico tridimensionale la soluzione dell'equazione (??) nell'intervallo $x \in [-2, 2]$ e con $\lambda \in [-0.5, 0.5]$

5. Si definisca una funzione che esprime il tensore d'inerzia per una particella puntiforme di massa m e di coordinate $X = (x, y, z)$

$$I = m \begin{pmatrix} y^2 + z^2 & -xy & -xz \\ -xy & x^2 + z^2 & -yz \\ -xz & -yz & x^2 + y^2 \end{pmatrix}\tag{3}$$

6. Si consideri un sistema di tre particelle puntiformi con le seguenti masse e coordinate:

$$\begin{aligned}m_1 &= 1, & X_1 &= (4, 1, 3) \\ m_2 &= 2, & X_2 &= (-1, -1, 1) \\ m_3 &= 4, & X_3 &= (1, 0, -2)\end{aligned}\tag{4}$$

e si definisca il tensore d'inerzia totale $I = \sum_{j=1}^3 I_j$. Si calcolino autovalori a_i e autovettori \vec{v}_i (con $i = 1, 2, 3$) della matrice I .

7. Si verifichi che la matrice degli autovettori diagonalizza la matrice I .

8. Si consideri l'ellissoide d'inerzia, ovvero la superficie definita dall'equazione

$$1 = \frac{a_1}{a_{sum}} \rho_x^2 + \frac{a_2}{a_{sum}} \rho_y^2 + \frac{a_3}{a_{sum}} \rho_z^2\tag{5}$$

dove $a_{sum} = \sum_{j=1}^3 a_j$, con a_j gli autovalori del tensore di inerzia, e dove ρ_i sono delle coordinate cartesiane lungo gli assi principali d'inerzia.

Si disegni l'ellissoide nel caso del tensore I , tenendo conto del fatto che i tre assi principali \vec{v}_j sono ortogonali tra di loro, ma non coincidono con gli assi cartesiani.

LEGGERE il seguente suggerimento:

Come parametrizzazione per il punto nella base (x, y, z) si utilizzi quella seguente:

$(\frac{a_{sum}}{a_1} \sin \theta \cos \phi, \frac{a_{sum}}{a_2} \sin \theta \sin \phi, \frac{a_{sum}}{a_3} \cos \theta)$

Per visualizzare l'ellissoide ruotato, si prenda il primo autovettore (primo asse principale) e si calcolino l'angolo θ_1 che esso forma con l'asse cartesiano z e l'angolo ϕ_1 che la sua proiezione sul piano (x, y) forma con l'asse x ; si scrivano quindi le due rotazioni necessarie per ruotare l'asse z nel primo asse principale.

Come visto a lezione, per determinare il valore di ciascun punto si possono dare più istruzioni all'interno del comando `ParametricPlot3D`; si proceda quindi a ruotare ciascun punto, scritto nella base (x, y, z) , utilizzando le due matrici di rotazione determinate prima.

9. Si consideri la seguente mappa (mappa logistica):

$$x_{n+1} = rx_n(1 - x_n) \quad (6)$$

e si ponga come condizione iniziale $x_0 = 0.8$.

Si scriva una funzione che genera, per r assegnato, i primi 1000 valori della serie.

Utilizzando il comando `Take`, si estraggano gli ultimi 32 elementi della lista precedente; con `Union` ci si riduca agli elementi distinti tra questi 32.

Con `Map` si generi la lista delle coppie $\{r, y\}$ dove y è uno degli elementi ottenuti nel punto precedente.

Si effettui una scansione in $r \in [2.4, 3.8]$ con passo pari a 0.01, e si generi la lista delle liste di coppie del punto precedente; utilizzando il comando `Flatten[lista, 1]` ci si riduca a una sola lista di tutte le coppie $\{r, y\}$, che può infine essere visualizzata con `ListPlot`.

C++

Si risolva l'esercizio proposto. Per facilitare la correzione, se possibile includere tutto in un unico file sorgente. La sufficienza è raggiunta risolvendo correttamente i primi tre punti.

Esercizio

1. Si scriva una classe **Matrix** che rappresenti una matrice 2x2. Tra i membri privati si pongano le quattro componenti (reali); tra i membri pubblici si scriva un opportuno costruttore che richieda gli elementi di matrice, con valori di default impostati in modo da realizzare la matrice identità.
2. Si implementi l'operatore "*" (membro di **Matrix**), che restituisce il prodotto riga per colonna.
3. Si implementi una funzione membro **Print**, che stampi su standard output la generica matrice nella forma
$$\begin{matrix} a & b \\ c & d \end{matrix}$$
4. Si scriva un **main** che istanzi una matrice identità ed una $m = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$ e si controlli, stampando i risultati su **cout**, che il prodotto a sinistra (come anche quello a destra) dell'identità con M sia uguale a M .
5. Tra i membri pubblici, si scriva una funzione membro **Det**, che restituisca il determinante della matrice.
6. Si scriva una classe **RandomMatrix** che erediti pubblicamente da **Matrix**. Il costruttore di default (senza parametri) dovrà generare una matrice simmetrica con elementi random reali compresi tra 0 e 1.
7. Si crei un **std::vector** di puntatori a **Matrix**, e lo si riempia con la matrice identità, con la matrice $\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ e poi con 10 matrici simmetriche random (tutte allocate dinamicamente; ricordarsi di liberare la memoria allocata, alla fine del programma; in alternativa usare degli smart pointer, se si preferisce).
8. Si vuole ora ordinare il vettore per determinanti crescenti, usando l'algoritmo

```
std::sort(it1, it2, pred)
```

che ordina gli elementi compresi tra gli iteratori **it1** (compreso) e **it2** (escluso), confrontandoli attraverso il predicato **pred** (si ricorda che un predicato è una funzione che restituisce un **bool**). Si scriva il predicato opportuno, si ordini il vettore, e si controlli infine (stampandoli su **cout**) che il primo e l'ultimo elemento siano rispettivamente σ_x e l'identità.