Prova d'esame di Laboratorio di Calcolo I
per il corso di laurea in Matematica
21 Luglio 2010

Tema d'esame: studio di alcune proprietà dei polinomi di Hermite.

Descrizione del metodo di calcolo

I polinomi di Hermite vengono introdotti in modo assai naturale per poter risolvere dei problemi relativi ad alcuni famosi argomenti di probabilità, fisica, etc. Per esempio, essi sono utilizzati per esprimere gli autostati dell'oscillatore armonico in meccanica quantistica.

I polinomi di Hermite H_n possono essere definiti (e, quindi, *calcolati*) in modo ricorsivo, come segue. Siano

$$H_0(x) = 1 \ , \qquad H_1(x) = 2x \tag{1}$$

e, $\forall\ n \geq 2$,

$$H_n(x) = 2xH_{n-1}(x) - 2(n-1)H_{n-2}(x) \ . \tag{2}$$

Dalle definizioni riportate nelle formule (1)–(2), si dimostra facilmente (per induzione) che $H_n(x)$ è un polinomio di grado n , a coefficienti interi e con la stessa parità di n stesso, $\forall\ n \geq 0$. A titolo di esempio, riportiamo qui di seguito l'espressione di $H_n(x)$ per $2 \leq n \leq 8$:

$$\begin{aligned}
H_2(x) &= 4x^2 - 2 \ , \\
H_3(x) &= 8x^3 - 12x \ , \\
H_4(x) &= 16x^4 - 48x^2 + 12 \ , \\
H_5(x) &= 32x^5 - 160x^3 + 120x \ , \\
H_6(x) &= 64x^6 - 480x^4 + 720x^2 - 120 \ , \\
H_7(x) &= 128x^7 - 1344x^5 + 3360x^3 - 1680x \ , \\
H_8(x) &= 256x^8 - 3584x^6 + 13440x^4 - 13440x^2 + 1680 \ .
\end{aligned} \tag{3}$$

Si può dimostrare che la formula (2) (cioè quella che permette di calcolare ricorsivamente i polinomi di Hermite) è equivalente a quella seguente, che non dipende da H_{n-2} , ma include anche la derivata di H_{n-1} :

$$H_n(x) = 2xH_{n-1}(x) - H'_{n-1}(x) \ . \tag{4}$$

I polinomi di Hermite godono di varie proprietà, la più importante delle quali è, probabilmente, quella di costituire una base ortogonale completa nello spazio funzionale $L^2\big(\mathbf{R}\ , \exp(-x^2)\mathrm{d}x\big)$. Ciò è dovuto anche alla seguente relazione di ortogonalità:

$$\int_{-\infty}^{+\infty} H_m(x)H_n(x)\exp(-x^2)\,\mathrm{d}x = \begin{cases} n!\,2^n\sqrt{\pi} & \text{se } m = n\,, \\ 0 & \text{se } m \neq n\,. \end{cases} \tag{5}$$

Per completezza, introduciamo $\langle f, g\rangle = \int_{-\infty}^{+\infty} f(x)g(x)\exp(-x^2)\,\mathrm{d}x$, che è un'applicazione binaria definita per ogni coppia di funzioni $f : \mathbf{R} \mapsto \mathbf{R}$ e $g : \mathbf{R} \mapsto \mathbf{R}$ tali che $\langle f, f\rangle < \infty$ e $\langle g, g\rangle < \infty$; si può dimostrare che $\langle\cdot,\cdot\rangle$ soddisfa le proprietà di un prodotto scalare.

Inoltre, i polinomi di Hermite compaiono nella seguente espansione in serie di Taylor:

$$\exp\left(2xt - t^2\right) = \sum_{n=0}^{+\infty} H_n(x) \frac{t^n}{n!} \ ; \tag{6}$$

quest'ultima equazione sussiste per ogni $x \in \mathbf{R}$ e $t \in \mathbf{R}$.

Rappresentazione dei polinomi di Hermite
Quando si effettua il calcolo dei polinomi di Hermite tramite un programma in linguaggio **C**, bisogna per prima cosa adottare una buona tecnica per *rappresentare* in modo conveniente questi stessi polinomi sul computer. Ai fini della soluzione degli esercizi qui proposti, la seguente strategia è adeguata, senza essere troppo complicata.

Nel programma in linguaggio **C**, si identifichi ciascun polinomio di Hermite H_n con un vettore di tipo `double` di dimensione $N+1$; i primi $n+1$ elementi di tale vettore ospiteranno i valori $a_0 \ \ldots \ , \ a_n$ dei coefficienti del polinomio $H_n(x) = a_0 + a_1 x + \ldots a_n x^n$. Ovviamente, tale rappresentazione è valida solo se $n \le N$; al fine di garantire che questa condizione sia sempre verificata durante l'esecuzione del programma, dovranno essere presenti gli opportuni controlli laddove verrà inserito il valore di n. Inoltre, precisiamo che la scelta del tipo `double` a proposito dei coefficienti dei polinomi di Hermite (nonostante, come abbiamo già osservato in precedenza, essi siano tutti interi) è dovuta al fatto che ciò permette di rappresentare più numeri interi rispetto al tipo `int` ed è coerente con il tipo che viene utilizzato per valutare $H_n(x)$ quando $x \notin \mathbf{Z}$.

È sicuramente opportuno stabilire il valore del parametro N per mezzo di una direttiva `#define`. Tale valore dovrà essere abbastanza grande, senza però essere enorme al punto di rallentare eccessivamente l'esecuzione del programma; per fissare le idee, $N = 100$ dovrebbe essere una scelta opportuna per garantire il buon funzionamento del programma in ognuna delle parti che verranno descritte in seguito.

Obiettivo (intermedio) 1:
si scriva un programma in linguaggio **C** che verifica *numericamente* l'equazione (6). Il programma deve contenere:

(A) una *function* che ha due argomenti: un intero $\bar{n}$ e un numero reale $\bar{x}$; dopo aver utilizzato le definizioni iniziali in (1) e (*iterativamente*) la formula (2), tale *function (alla fine della chiamata)*[⋆] deve restituire il valore di $H_{\bar{n}}(\bar{x})$;

(B) una *function* che ha come due argomenti i numeri reali x e t; tale *function (alla fine della chiamata)* deve restituire il valore del membro di destra dell'equazione (6), il cui calcolo deve essere effettuato così come descritto ai seguenti punti (B1)–(B3);

  (B1) si definiscano i valori *iniziali* delle variabili $n \in \mathbf{Z}$, $\sigma \in \mathbf{R}$, $\mathcal{F} \in \mathbf{R}$ e $\mathcal{P} \in \mathbf{R}$ nel modo seguente: $n = 0$, $\sigma = 0$, $\mathcal{F} = 1$ e $\mathcal{P} = 1$;

  (B2) si iterino le operazioni descritte ai seguenti punti (B21)–(B24) mentre è verificata la condizione (di permanenza nel ciclo), che è espressa al punto (B3);

[⋆] Siccome nella *function* richiesta al punto (A) dell'obiettivo 1 "non si lavora con x variabile, ma con $x = \bar{x}$ fissato", si osservi che, in questa parte del programma, è assolutamente **non necessario** rappresentare i polinomi $H_n(x)$ con dei vettori, così come era invece stato descritto in precedenza.

(B21) si ponga $\bar{\sigma} = \sigma$;

(B22) si (ri)aggiorni il valore di σ , ponendo $\sigma = \bar{\sigma} + H_n(x) \cdot \mathcal{P}/\mathcal{F}$, dove il valore di $H_n(x)$ viene determinato effettuando un'opportuna *chiamata della function descritta al punto (A)*;

(B23) si (ri)aggiorni il valore di n , incrementandolo di 1 ;

(B24) si (ri)aggiornino i valori di $\mathcal{F}$ e $\mathcal{P}$, ponendo $\mathcal{F} = \mathcal{F} \cdot n$ e $\mathcal{P} = \mathcal{P} \cdot t$;

(B3) se $\sigma \neq \bar{\sigma}$, allora si torni a ripetere le operazioni descritte ai precedenti punti (B21)–(B24);

(C) l'*input da tastiera* (da effettuarsi nella *main function*) di due fissati valori di x e t ; essi devono essere sottoposti a test in modo tale che *entrambi devono essere compresi* nell'intervallo $[-1, 1]$, e se ciò non sarà verificato per almeno uno di essi, allora tale valore dovrà essere *reinserito correttamente* [⊙] ;

(D) il calcolo dell'errore relativo $\varepsilon = \left| \exp\left(2xt - t^2\right) - \sum_{n=0}^{+\infty} H_n(x) \frac{t^n}{n!} \right| / \exp\left(2xt - t^2\right)$, dove il valore approssimato della serie $\sum_{n=0}^{+\infty} H_n(x) \frac{t^n}{n!}$ viene determinato effettuando la *chiamata della function descritta al punto (B)*;

(E) la stampa sul video *in formato esponenziale* dei valori di $\exp\left(2xt - t^2\right)$ e dell'errore relativo ε così come definito al punto (D); ovviamente, la verifica numerica dell'equazione (6) *è da considerarsi ben riuscita* se ε è al più dell'ordine di grandezza dell'*errore di macchina.*

Un consiglio

È sicuramente comodo (e *prudente*) utilizzare delle *functions* o parti di programma, che sono incluse in altri programmi precedentemente scritti dagli studenti stessi o dal docente (e reperibili in rete). Ciò è vero non solo al fine di conseguire l'obiettivo 1, ma soprattutto riguardo a quelli che verranno descritti in seguito.

Obiettivo (intermedio) 2:

si integri il programma richiesto dall'obiettivo 1, in modo tale da effettuare la verifica numerica della relazione di ortogonalità (5). A tal fine, si proceda come segue:

(A) si scriva una *function* che ha tre argomenti: gli indici interi m , n e la variabile reale x ; *(alla fine della chiamata)* tale *function* deve restituire il valore di $H_m(x)H_n(x)\exp(-x^2)$, cioè proprio la funzione integranda che compare nell'equazione (5); si ricordi che i valori di $H_m(x)$ e $H_n(x)$ possono essere determinati con *due opportune chiamate della function descritta al punto (A) dell'obiettivo 1*;

(B) si scriva una *function* che ha 6 argomenti: gli estremi a e b di un intervallo di integrazione, il numero di sotto-intervalli *numsubint*, due indici interi m , n e infine un *puntatore a una function f*, la quale, a sua volta, dipenderà da altri tre argomenti (i primi due dei quali sono degli `int` e l'ultimo è un `double`); *(alla fine della chiamata)* tale *function* deve restituire il valore approssimato dell'integrale definito $\int_a^b f(m, n, x)\,\mathrm{d}x$, che viene calcolato utilizzando il *metodo del punto medio* basato su una griglia di *numsubint* sotto-intervalli;

[⊙] In verità, imponiamo le condizioni $x \in [-1, 1]$ e $t \in [-1, 1]$ solo per avere una convergenza delle somme parziali al risultato che sia abbastanza veloce da ridurre drasticamente gli errore numerici. Ricordiamo che, in effetti, l'uguaglianza in (6) vale per ogni $x \in \mathbf{R}$ e $t \in \mathbf{R}$.

(C) all'interno della *main function*, si effettui l'*input da tastiera* dei valori degli interi m, n e del numero reale L; ciascuno di questi valori deve essere sottoposto a un test in modo tale che, se l'indice m *non è compreso* nell'intervallo $[0, N]$, deve essere *reinserito correttamente*; altrettanto si faccia proposito di n; invece, la quantità L (che rappresenta la metà dell'intervallo di integrazione) deve essere *positiva* per poter superare il test, a cui essa viene sottoposta;

(D) all'interno della *main function*, si effettui *un'opportuna chiamata della function descritta al punto (B)*, in modo da calcolare l'integrale $\int_{-L}^{+L} H_m(x)H_n(x)\exp(-x^2)\,\mathrm{d}x$; il valore di tale integrale deve essere approssimato utilizzando $100\,000$ sotto-intervalli dell'insieme di integrazione $[-L, L]$; inoltre, la suddetta *chiamata della function relativa al punto (B)* deve essere tale che, tra gli argomenti, viene passato anche l'indirizzo della *function* descritta al punto (A);

(E) all'interno della *main function*, si stampi sul video *in formato esponenziale* il valore approssimato dell'integrale $\int_{-L}^{+L} H_m(x)H_n(x)\exp(-x^2)\,\mathrm{d}x$, così come è stato ottenuto grazie al procedimento descritto al precedente punto (D); inoltre, si stampi (sempre *in formato esponenziale*) la differenza (la quale *dovrà essere piccola, giusto?!*) tra tale valore e quello atteso, cioè quello riportato nel membro di destra dell'equazione (5).

Obiettivo (intermedio) 3:
si integri il programma richiesto dall'obiettivo 2, in modo tale da effettuare il calcolo *iterativo* delle espansioni dei polinomi di Hermite utilizzando le formule (1)–(2). A tal fine, si proceda come segue:

(A) si scriva una *function* che ha tre argomenti: un vettore $\underline{a}$, un intero n e un secondo vettore $\underline{b}$, tali che $\underline{a}$ e n sono, rispettivamente, i coefficienti e il grado di un polinomio $P(x) = a_0 + a_1x + \ldots a_nx^n$; *(alla fine della chiamata)* nel vettore $\underline{b}$ dovranno essere riportati i coefficienti del polinomio $Q(x) = P(x)$ (proprio così, $Q(x)$ altro non è che una copia di $P(x)$) cioè $P(x) = Q(x) = b_0 + b_1x + \ldots b_nx^n$;

(B) si scriva una *function* che ha tre argomenti: un vettore $\underline{a}$, un intero n e un numero reale c, tali che $\underline{a}$ e n sono, rispettivamente, i coefficienti e il grado di un polinomio $P(x) = a_0 + a_1x + \ldots a_nx^n$; *(alla fine della chiamata)* nel vettore $\underline{a}$ dovranno essere riportati i coefficienti del nuovo polinomio $cP(x)$, cioè $cP(x) = a_0 + a_1x + \ldots a_nx^n$;

(C) si scriva una *function* che ha due argomenti: un vettore $\underline{a}$ e un intero n, tali che essi sono, rispettivamente, i coefficienti e il grado di un polinomio $P(x) = a_0 + a_1x + \ldots a_nx^n$; *(alla fine della chiamata)* nel vettore $\underline{a}$ dovranno essere riportati i coefficienti del prodotto polinomiale $xP(x)$, cioè $xP(x) = a_0 + a_1x + \ldots a_{n+1}x^{n+1}$;

(D) si scriva una *function* che ha cinque argomenti: un vettore $\underline{a}$, un intero n, un secondo vettore $\underline{b}$, un altro intero m e un terzo vettore $\underline{c}$; tali che $\underline{a}$ e n sono, rispettivamente, i coefficienti e il grado di un polinomio $P(x) = a_0 + a_1x + \ldots a_nx^n$, mentre $\underline{b}$ e m sono, rispettivamente, i coefficienti e il grado di un polinomio $Q(x) = b_0 + b_1x + \ldots b_mx^m$; a tale *function* è richiesto di funzionare correttamente solo se $n \geq m$, altrimenti l'esecuzione dell'intero programma può essere arrestata, dopo aver stampato su video un messaggio di errore; *(alla fine della chiamata)* nel vettore $\underline{c}$ dovranno essere riportati i coefficienti della somma polinomiale $P(x) + Q(x)$, cioè $P(x) + Q(x) = c_0 + c_1x + \ldots c_nx^n$;

(E) si scriva una *function* che ha due argomenti: un intero n e un vettore $\underline{a}$; tale *function* effettua il calcolo dell'espansione del polinomio di Hermite H_n , quindi *(alla fine della chiamata)* nel vettore $\underline{a}$ dovranno essere riportati i coefficienti di $H_n(x)$, cioè $H_n(x) = a_0 + a_1x + \ldots a_nx^n$; questa *function* può essere realizzata in modo da tradurre in linguaggio di programmazione la procedura descritta ai seguenti punti (E1)–(E4);

(E1) all'interno della *function* deve essere allocata la memoria per almeno tre vettori che rappresentano i polinomi $Q(x)$, $R(x)$ e $S(x)$ (inoltre, saranno necessari alcuni polinomi che ospiteranno i risultati di qualche operazione intermedia);

(E2) si definiscano opportunamente i valori dei coefficienti dei vettori, rispettivamente, corrispondenti a $Q(x)$ e $R(x)$ in modo tale che $Q(x) = 1$ e $R(x) = 2x$ (cioè $Q(x)$ e $R(x)$ vengono *inizialmente posti uguali* a $H_0(x)$ e $H_1(x)$);

(E3) si iterino le istruzioni descritte ai seguenti punti (E31)–(E33) in modo tale da costituire un ciclo, che viene eseguito per i (che è un *contatore* intero) che va da 2 fino a n ;

(E31) utilizzando ripetutamente *le function descritte ai punti (A)–(D) del presente obiettivo 3*, si effettui il calcolo del polinomio $S(x) = 2x \cdot R(x) - 2(i-1) \cdot Q(x)$, cioè $R(x)$ viene posto uguale a $H_i(x)$;

(E32) con un'opportuna *chiamata della function relativa al punto (A)*, si ridefinisca $Q(x)$ in modo tale che $Q(x) = R(x)$, cioè $Q(x)$ viene ora posto uguale a $H_{i-1}(x)$;

(E33) con un'opportuna *chiamata della function relativa al punto (A)*, si ridefinisca $R(x)$ in modo tale che $R(x) = S(x)$, cioè $R(x)$ viene ora posto uguale a $H_i(x)$;

(E4) si effettuino alcuni test alla fine della *function*, i quali devono essere strutturati come segue: se $n = 0$, allora si ridefinisca $S(x)$ in modo tale che $S(x) = Q(x)$, cioè $S(x)$ viene ora posto uguale a $H_0(x)$; altrimenti, se $n = 1$, allora si ridefinisca $S(x)$ in modo tale che $S(x) = R(x)$, cioè $S(x)$ viene ora posto uguale a $H_1(x)$; si osservi che, dopo l'esecuzione dei suddetti test e del ciclo descritto ai punti (E3)–(E33), $S(x)$ è sicuramente uguale al polinomio di Hermite richiesto, cioè $H_n(x)$;

(F) si scriva una *function* che ha due argomenti: un vettore $\underline{a}$ e un intero n , tali che essi sono, rispettivamente, i coefficienti e il grado di un polinomio $P(x) = a_0 + a_1x + \ldots a_nx^n$; tale *function* non deve fare altro che stampare ordinatamente sul video i coefficienti del polinomio $P(x)$;

(G) all'interno della *main function*, si ripeta una seconda volta l'*input da tastiera* del valore di n che deve essere sottoposto a un test in modo tale che, se esso *non è compreso* nell'intervallo $[0, N]$, deve essere *reinserito correttamente*;

(H) all'interno della *main function* si effettuino *le opportune chiamate delle function descritte ai punti (E) e (F)* per visualizzare l'espansione calcolata del polinomio $H_n(x)$; ovviamente, si consiglia di confrontare i risultati ottenuti con qualcuna delle espressioni di $H_2(x)$, ... , $H_8(x)$, riportate in formula (3).

Obiettivo (intermedio) 4:
si integri il programma richiesto dall'obiettivo 3, in modo tale da effettuare, $\forall\ x \in \mathbf{R}$, la verifica dell'equazione differenziale (4) (che durante questo esercizio *non* usiamo ricorsivamente). A tal fine si proceda come segue:

(A) si scriva una *function* che ha tre argomenti: un vettore $\underline{a}$, un intero n e un secondo vettore $\underline{b}$, tali che $\underline{a}$ e n sono, rispettivamente, i coefficienti e il grado di un polinomio $P(x) = a_0 + a_1x + \ldots a_nx^n$; *(alla fine della chiamata)* nel vettore $\underline{b}$ dovranno essere riportati i coefficienti della derivata di $P(x)$, cioè $P'(x) = b_0 + b_1 x + \ldots b_{n-1} x^{n-1}$;

(B) all'interno della *main function*, si ripeta una terza volta l'*input da tastiera* del valore di n che deve essere sottoposto a un test in modo tale che, se esso *non è compreso* nell'intervallo $[2, N]$, deve essere *reinserito correttamente*;

(C) all'interno della *main function* deve essere allocata la memoria per almeno quattro vettori che rappresentano dei polinomi $Q(x)$, $Q'(x)$, $R(x)$ e $S(x)$ (oltre a uno relativo a $H_n(x)$, che probabilmente era già stato definito in precedenza);

(D) all'interno della *main function*, si effettui il calcolo dell'espansione polinomiale di $H_n(x) - 2xH_{n-1}(x) + H'_{n-1}(x)$; tale calcolo può essere realizzato in modo da tradurre in linguaggio di programmazione la procedura descritta ai seguenti punti (D1)–(D4);

- (D1) si calcolino le espansioni del polinomio $H_n(x)$ e $Q(x) = H_{n-1}(x)$, effettuando *due opportune chiamate della function descritta al punto (E) dell'obiettivo 3*;
- (D2) si calcoli l'espansione del polinomio $R(x) = -2xQ(x)$, effettuando *le opportune chiamate delle function descritte ai punti (A), (B) e (C) dell'obiettivo 3*;
- (D3) si calcoli l'espansione della derivata $Q'(x)$, effettuando una *chiamata della function descritta al punto (A) del presente obiettivo 4*;
- (D4) si calcoli l'espansione del polinomio $S(x) = H_n(x) + R(x) + Q'(x)$, effettuando *le opportune chiamate delle function descritte ai punti (A) e (D) dell'obiettivo 3*;

(E) all'interno della *main function* si effettui la *chiamata della function descritta al punto (F) dell'obiettivo 3* per visualizzare l'espansione calcolata del polinomio $S(x)$; ovviamente, la verifica dell'equazione differenziale (4) è da considerarsi ben riuscita se la suddetta espansione polinomiale di $S(x) = H_n(x) - 2xH_{n-1}(x) + H'_{n-1}(x)$ risulta essere uguale a zero.

Obiettivo (finale) 5:

al programma richiesto dagli obiettivi precedenti si aggiunga la scrittura ordinata su **file** di una successione di coppie di valori di x_i e $H_n(x_i)$, a partire dalla quale è possibile tracciare il grafico della funzione $x \mapsto H_n(x)$. Inoltre, si compiano le operazioni necessarie al fine di visualizzare tale grafico, utilizzando il software **gnuplot**. A tali scopi si può procedere come segue:

(A) all'interno della *main function*, si effettui un ultimo *input da tastiera* dell'intero $n \in [0, N]$ e del numero reale *positivo* L;

(B) si esegua un ciclo all'interno del quale la variabile indipendente x assume tutti i valori appartenenti all'insieme $\{x_i\}_{i=0}^{10000}$, dove $x_0 = -L$, $x_{10000} = L$, $x_i \in [x_0, x_{10000}]$ e gli x_i costituiscono una "griglia di punti equidistanti", cioè $x_{j+1} - x_j = x_j - x_{j-1}$ $\forall\ j = 1, \ldots, 9999$; inoltre, per ciascun punto dell'insieme $\{x_i\}_{i=0}^{10000}$ si calcoli il corrispondente valore di $H_n(x_i)$, con un'opportuna *chiamata della function descritta al punto (A) dell'obiettivo 1*;

(C) all'interno del suddetto ciclo si proceda alla stampa su **file**, in modo tale che su ciascuna delle sue righe compaia prima il valore di x_i e poi quello corrispondente di $H_n(x_i)$;

(D) si scriva un **file** contenente i comandi necessari al software **gnuplot**, al fine di far apparire sullo schermo un grafico (esteticamente apprezzabile) della funzione $x \mapsto H_n(x)$, in modo tale che il suddetto grafico sia tracciato a partire dai dati scritti nel **file** richiesto dal precedente punto (C).