Prova d'esame di Laboratorio di Calcolo I
per il corso di laurea in Matematica
19 Luglio 2012

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

Descrizione del metodo di calcolo

I polinomi di Chebyshev di prima specie T_n vengono comunemente utilizzati in analisi numerica. Infatti, le loro radici (cioè le soluzioni delle equazioni $T_n(x) = 0$) sono usati come nodi per costruire delle opportune approssimazioni polinomiali di funzioni che sono almeno continue.

I polinomi Chebyshev di prima specie T_n possono essere definiti (e, quindi, *calcolati*) in modo ricorsivo nel modo seguente. Siano

$$T_0(x) = 1 \ , \qquad T_1(x) = x \tag{1}$$

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

$$T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x) \ . \tag{2}$$

Dalle definizioni riportate nelle formule (1)–(2), si dimostra facilmente (per induzione) che $T_n(x)$ 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 $T_n(x)$ per $2 \leq n \leq 9$:

$$\begin{aligned}
T_2(x) &= 2x^2 - 1 \ , \\
T_3(x) &= 4x^3 - 3x \ , \\
T_4(x) &= 8x^4 - 8x^2 + 1 \ , \\
T_5(x) &= 16x^5 - 20x^3 + 5x \ , \\
T_6(x) &= 32x^6 - 48x^4 + 18x^2 - 1 \ , \\
T_7(x) &= 64x^7 - 112x^5 + 56x^3 - 7x \ , \\
T_8(x) &= 128x^8 - 256x^6 + 160x^4 - 32x^2 + 1 \ , \\
T_9(x) &= 256x^9 - 576x^7 + 432x^5 - 120x^3 + 9x \ .
\end{aligned} \tag{3}$$

Si può dimostrare che la formula (2) (cioè quella che permette di calcolare ricorsivamente i polinomi di Chebyshev di prima specie T_n) è equivalente a quella seguente, che include anche le derivate di T_{n+1} e T_{n-1}:

$$2(n^2 - 1)T_n(x) = (n-1)T'_{n+1}(x) - (n+1)T'_{n-1}(x) \ . \tag{4}$$

I polinomi di Chebyshev godono di varie proprietà; una delle più importanti è sicuramente la seguente relazione di ortogonalità:

$$\int_{-1}^{1} T_m(x)T_n(x)\,\frac{\mathrm{d}x}{\sqrt{1-x^2}} = \begin{cases} 0 & \text{se } m \neq n\,, \\ \pi & \text{se } m = n = 0\,, \\ \dfrac{\pi}{2} & \text{se } m = n \neq 0\,. \end{cases} \tag{5}$$

A proposito della formula precedente, per completezza, si ricorda che l'applicazione binaria $\langle f\,,\,g\rangle = \int_{-1}^{1} f(x)\,g(x)\,\mathrm{d}x/\sqrt{1-x^2}$ è definita per ogni coppia di funzioni $f:[-1,1]\mapsto \mathbf{R}$ e $g:[-1,1]\mapsto \mathbf{R}$ 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. Di fatto, per dimostrare la relazione di ortogonalità (5), si utilizza un'altra proprietà assai importante dei polinomi di Chebyshev di prima specie, cioè

$$T_n\big(\cos(\vartheta)\big) = \cos(n\vartheta) \qquad \forall\ n\geq 0\,,\ \vartheta\in\mathbf{R}\ . \tag{6}$$

Anche i polinomi di Chebyshev di seconda specie U_n possono essere definiti (e, quindi, *calcolati*) in modo ricorsivo, così come descritto dalla formula seguente:

$$U_0(x)=1\ , \qquad U_1(x)=2x\ , \qquad U_n(x)=2xU_{n-1}(x)-U_{n-2}(x)\ \ \forall\ n\geq 2\ . \tag{7}$$

I polinomi di Chebyshev di seconda specie sono uguali alle derivate di quelli di prima specie a meno di un fattore moltiplicativo; infatti, vale la relazione $T'_{n+1}=(n+1)U_n$, che può essere riformulata come segue:

$$\int_a^b U_n(x)\,\mathrm{d}x = \frac{T_{n+1}(b)-T_{n+1}(a)}{n+1}\ . \tag{8}$$

Rappresentazione dei polinomi di Chebyshev

Quando si effettua il calcolo dei polinomi di Chebyshev 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 Chebyshev di prima specie T_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 $T_n(x)=a_0+a_1x+\ldots+a_nx^n$. Ovviamente, tale rappresentazione è valida solo se $n\leq N$; 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 Chebyshev di prima specie (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 $T_n(x)$ 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, $N=50$ 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 $T_{\bar{n}}(\bar{x})$;

(B) l'*input da tastiera* (da effettuarsi nella *main function*) di due fissati valori di n e ϑ ; essi devono essere sottoposti a test in modo tale che $n \in \mathbf{N}$ *deve essere compreso* nell'intervallo $[0, N]$, mentre ϑ *deve essere compreso* nell'intervallo $[-\pi, \pi]$, se ciò non sarà verificato per almeno uno di essi, allora tale valore dovrà essere *reinserito correttamente* [⊙] ;

(C) il calcolo dell'errore assoluto $\varepsilon = \left|T_n\big(\cos(\vartheta)\big) - \cos(n\vartheta)\right|$, dove il valore approssimato di $T_n\big(\cos(\vartheta)\big)$ viene determinato effettuando un'opportuna *chiamata della function descritta al punto (A)*;

(D) la stampa sul video dei valori di $\cos(n\vartheta)$ e, *in formato esponenziale*, dell'errore assoluto ε , così come esso è stato definito al punto (C); ovviamente, la verifica numerica dell'equazione (6) *è da considerarsi ben riuscita* se il valore di ε è circa 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 dell'equazione (8), che istituisce una relazione tra i polinomi di Chebyshev di prima e seconda specie. A tal fine, si proceda come segue:

(A) si scriva una *function* che ha due argomenti: un intero $\bar{n}$ e un numero reale $\bar{x}$; utilizzando *iterativamente* la formula (7) (in modo simile a quanto richiesto al punto (A) dell'obiettivo 1), tale *function (alla fine della chiamata)* deve restituire il valore di $U_{\bar{n}}(\bar{x})$;

(B) si scriva una *function* che ha 5 argomenti: gli estremi a e b di un intervallo di integrazione, il numero di sotto-intervalli *numsubint* , un indice intero n e, infine, un *puntatore a una function* f , la quale, a sua volta, dipenderà da altri due argomenti (il primo dei quali è un `int` e l'ultimo è un `double`); *(alla fine della chiamata)* tale *function* deve restituire il valore approssimato dell'integrale definito $\int_a^b f(n,x)\,\mathrm{d}x$, che viene calcolato utilizzando il *metodo del punto medio* basato su una griglia di *numsubint* sotto-intervalli;

[⋆] 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 $T_n(x)$ con dei vettori, così come era invece stato descritto in precedenza.

[⊙] In verità, imponiamo le condizioni $n \in [0, N]$ e $\vartheta \in [-\pi, \pi]$ solo per ridurre drasticamente gli errori numerici. Ricordiamo che, in effetti, l'uguaglianza in (6) vale per ogni $n \in \mathbf{N}$ e $\vartheta \in \mathbf{R}$.

(C) all'interno della *main function*, si effettui l'*input da tastiera* dei valori dell'intero n e dei due numeri reali a , b ; ciascuno di questi valori deve essere sottoposto a un test in modo tale che, se l'indice n *non è compreso* nell'intervallo $[0, N]$, deve essere *reinserito correttamente*; invece, le quantità a e b (che rappresentano gli estremi dell'intervallo di integrazione) *devono essere tali che* $a < b$ per poter superare il test, a cui esse vengono sottoposte;

(D) all'interno della *main function*, si effettui *un'opportuna chiamata della function descritta al punto (B)*, in modo da calcolare l'integrale $\int_a^b U_n(x)\,\mathrm{d}x$; il valore di tale integrale deve essere approssimato utilizzando $1\,000\,000$ di sotto-intervalli dell'insieme di integrazione $[a, b]$; 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 $\int_a^b U_n(x)\,\mathrm{d}x$, 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 (8).

Obiettivo (intermedio) 3:
si integri il programma richiesto dall'obiettivo 2, in modo tale da effettuare il calcolo *iterativo* delle espansioni dei polinomi di Chebyshev di prima specie, 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 Chebyshev di prima specie T_n , quindi *(alla fine della chiamata)* nel vettore $\underline{a}$ dovranno essere riportati i coefficienti di $T_n(x)$, cioè $T_n(x) = a_0 + a_1 x + \ldots + a_n x^n$; 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 $R(x) = x$ (cioè $Q(x)$ e $R(x)$ vengono *inizialmente posti uguali* a $T_0(x)$ e $T_1(x)$);

(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 $S(x) = 2x \cdot R(x) - Q(x)$, cioè $R(x)$ viene posto uguale a $T_i(x)$;

(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 $T_{i-1}(x)$;

(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 $T_i(x)$;

(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 $T_0(x)$; 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 $T_1(x)$; 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 Chebyshev (di prima specie) richiesto, cioè $T_n(x)$;

(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 $P(x) = a_0 + a_1 x + \ldots + a_n x^n$; 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 per la 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 $[0, N]$, 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 $T_n(x)$; ovviamente, si consiglia di confrontare i risultati ottenuti con qualcuna delle espressioni di $T_2(x)$, ... , $T_9(x)$, 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 $P(x) = a_0 + a_1 x + \ldots + a_n x^n$; *(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 quarta 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 $[1, N-1]$, deve essere *reinserito correttamente*;

(C) all'interno della *main function* deve essere allocata la memoria per almeno cinque vettori, i quali rappresentano dei polinomi che indichiamo con $Q(x)$, $R(x)$, $S(x)$, $T_{n+1}(x)$ e $T_{n-1}(x)$ (oltre a uno relativo a $T_n(x)$, che probabilmente era già stato definito in precedenza);

(D) all'interno della *main function*, si effettui il calcolo dell'espansione polinomiale di $2(n^2-1)T_n(x) - (n-1)T'_{n+1}(x) + (n+1)T'_{n-1}(x)$; tale calcolo può essere realizzato in modo da tradurre in linguaggio di programmazione la procedura descritta ai seguenti punti (D1)–(D6);

  (D1) si calcolino le espansioni dei polinomi $T_n(x)$, $T_{n+1}(x)$ e $T_{n-1}(x)$, effettuando *tre opportune chiamate della function descritta al punto (E) dell'obiettivo 3*;

  (D2) si calcoli l'espansione del polinomio $Q(x) = 2(n^2-1)T_n(x)$, effettuando *le opportune chiamate delle function descritte ai punti (A) e (B) dell'obiettivo 3*;

  (D3) si calcoli l'espansione del polinomio $R(x) = -(n-1)T'_{n+1}(x)$, effettuando *le opportune chiamate della function descritta al punto (A) del presente obiettivo 4 e di quella al punto (B) dell'obiettivo 3*;

  (D4) si calcoli l'espansione del polinomio $S(x) = 2(n^2-1)T_n(x) - (n-1)T'_{n+1}(x)$ e, successivamente, si ridefinisca $Q(x)$ in modo tale che $Q(x) = S(x)$, effettuando *le opportune chiamate delle function descritte ai punti (A) e (D) dell'obiettivo 3*;

  (D5) si calcoli l'espansione del polinomio $R(x) = (n+1)T'_{n-1}(x)$, effettuando *le opportune chiamate della function descritta al punto (A) del presente obiettivo 4 e di quella al punto (B) dell'obiettivo 3*;

  (D6) si calcoli l'espansione del polinomio $S(x) = 2(n^2-1)T_n(x) - (n-1)T'_{n+1}(x) + (n+1)T'_{n-1}(x)$, effettuando *un'opportuna chiamata della function descritta al punto (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 $S(x) = 2(n^2-1)T_n(x) - (n-1)T'_{n+1}(x) + (n+1)T'_{n-1}(x)$ 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 $T_n(x_i)$, a partire dalla quale è possibile tracciare il grafico della funzione $x \mapsto T_n(x)$. 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 $n \in [0, N]$;

(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 $x_0 = -1$, $x_{10000} = 1$, $x_i \in [x_0, x_{10000}]$ 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 $T_n(x_i)$, 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 $T_n(x_i)$;

(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 $x \mapsto T_n(x)$, in modo tale che il suddetto grafico sia tracciato a partire dai dati scritti nel **file** richiesto dal precedente punto (C).