Laboratorio Didattico di Matematica Computazionale

a.a. 2018-2019

Esercitazione 5

Polinomi

1. Polinomi in Matlab

In Matlab, un polinomio è definito dal vettore (riga o colonna) dei suoi coefficienti, cominciando dal termine di grado più alto. Per esempio, il polinomio

$$p(x) = x^5 + 2x^4 - 5x^3 + x - 6$$

si può rappresentare come

```
p=[1 2 -5 0 1 -6]
```

Alcuni comandi utili per lavorare con i polinomi sono i seguenti (cercate uso e sintassi nell'help):

```
polyval
roots
poly
conv
deconv
polyder
polyvalm
```

- Per fare pratica, provate a definire in Matlab il polinomio $p(x) = x^2 - x - 1$, calcolate le sue radici e verificate che il polinomio si annulla numericamente in corrispondenza delle radici calcolate:

  ```
  p=[1 -1 -1]
  r=roots(p)
  polyval(p,r)
  ```

  Viceversa, scegliamo un vettore `r` contenente le radici e calcoliamo i coefficienti del polinomio `p` corrispondente, quindi verifichiamo che `p` si annulli sulle radici assegnate e che il calcolo numerico delle radici di `p` restituisca gli stessi valori assegnati all'inizio:

  ```
  r=[0.1 0.5 1 -0.5]
  p=poly(r)
  polyval(p,r)
  roots(p)
  ```

- Il comando `poly` permette anche di calcolare il polinomio caratteristico di una matrice. Per esempio, definiamo $p(x)$ come il polinomio caratteristico della matrice

  $$A = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}.$$

  Il teorema di Cayley-Hamilton implica che $p(A) = 0$. Verifichiamolo numericamente:

  ```
  A=[0 1; 1 1]
  p=poly(A)
  polyvalm(p,A)
  ```

  Dovreste ottenere una matrice numericamente nulla.

- Grazie alle istruzioni `conv` e `deconv` possiamo moltiplicare e dividere polinomi. Per esempio, definiamo $p(x) = x^3 - 6x^2 + 12 - 8$, dividiamo $p(x)$ per $q(x) = x - 2$, poi moltiplichiamo il risultato di nuovo per $q(x)$ e verifichiamo di aver ottenuto proprio $p(x)$:

  ```
  p=[1 -6 12 -8]
  q=[1 -2]
  [g,r]=deconv(p,q)
  conv(g,q)
  ```

- Per tracciare il grafico di una funzione polinomiale definita su un intervallo $[a, b]$ possiamo valutare il polinomio su una discretizzazione dell'intervallo e applicare il comando `plot` ai risultati ottenuti. Per esempio, supponiamo di voler tracciare il grafico di $p(x) = x^4 - 9x^3 + 21x^2 - x - 30$ definito sull'intervallo $[-2, 6]$:

  ```
  p=[1 -9 21 1 -30]
  t=linspace(-2,6,500);
  pv=polyval(p,t);
  plot(t,pv)
  ```

Esercizio 1 *Scrivere una function* `perturbed_poly(p,t,n)` *che prenda in ingresso un vettore* `p` *contenente i coefficienti di un polinomio, un numero reale* `t` *e un intero positivo* `n`*, e disegni sul piano complesso gli zeri dei polinomi ottenuti sommando* `h*t` *al coefficiente costante di* `p`*, per* `h=0:n`.

Scrivere poi uno script che applichi la function appena definita agli esempi seguenti:

1. $p(x) = x^4 - 1$, $t = 0.05$, $n = 20$,
2. $p(x) = (x - 1)^4$, $t = 0.02$, $n = 40$,
3. $p(x) = \prod_{k=1}^{20}(x - 1/k)$, $t = 0.05$, $n = 20$.

Che cosa si può constatare? In particolare, che differenze notate tra il comportamento dell'esempio 1 e dell'esempio 2? Che cosa succede nell'esempio 3?

Esercizio 2 (Polinomi di Legendre). *I polinomi di Legendre sono definiti in modo ricorsivo come*

$$\begin{aligned}
p_1(x) &= 1, \\
p_2(x) &= x, \\
p_n(x) &= \frac{(2n-1)xp_{n-1}(x) - (n-1)p_{n-2}(x)}{n}, \qquad n \geq 3.
\end{aligned}$$

Scrivere una function `legendre(K)` *che prenda in ingresso un intero positivo* `K` *e disegni il grafico dei primi* `K` *polinomi di Legendre sull'intervallo* $[-1,1]$.

Facoltativo: *scrivere uno script che verifichi sperimentalmente la formula*

$$p_n(x) = \frac{1}{2^n n!}\frac{d^n}{dx^n}(x^2-1)^n$$

per alcuni valori di n. *Potrà essere utile il comando* `diff`.

Esercizio 3 (Polinomi di Chebyshev). *I polinomi di Chebyshev di prima specie sono definiti in modo ricorsivo come*

$$\begin{aligned}
p_1(x) &= 1, \\
p_2(x) &= x, \\
p_n(x) &= 2xp_{n-1}(x) - p_{n-2}(x), \qquad n \geq 3.
\end{aligned}$$

Scrivere una function `cheby(K)` *che prenda in ingresso un intero positivo* `K` *e disegni il grafico dei primi* `K` *polinomi di Chebyshev sull'intervallo* $[-1,1]$.

2. Iterazione di Graeffe

Sia $p(x)$ un polinomio di grado n e definiamo $q(x) = p(x)p(-x)$. Non è difficile vedere che il polinomio $q(x)$, di grado $2n$, ha tutti i coefficienti di grado dispari nulli. Verifichiamo numericamente questa proprietà su un esempio:

```
p=[1 -6 12 -8];
degree=length(p)-1;
pminus=p.*((-1).^[degree:-1:0])
q=conv(p,pminus)
```

Di conseguenza, possiamo vedere $q(x)$ come un polinomio in x^2, cioè scrivere

$$q(x) = p_1(x^2),$$

dove $p_1(x)$ è un polinomio di grado n.

Sfruttando questa osservazione, definiamo in modo ricorsivo una successione di polinomi di grado n:

$$\begin{aligned}
p_0(x) &= p(x), \\
p_{i+1}(x^2) &= p_i(x)p_i(-x)/r_i, \qquad i = 0,1,2,\ldots
\end{aligned}$$

dove r_i è il massimo modulo dei coefficienti di $p_i(x)p_i(-x)$. La divisione per r_i ha lo scopo di non far divergere i coefficienti dei polinomi.

La successione ha le seguenti proprietà:

1. Se il polinomio iniziale $p(x)$ ha s radici di modulo minore di 1 e $n-s$ radici di modulo maggiore di 1, la successione converge al polinomio x^s.

2. Se $p(x)$ ha almeno una radice di modulo 1, la successione può non convergere.

Esercizio 4 *Scrivere una function che prende in input il vettore dei coefficienti di un polinomio $p(x)$ e un intero positivo K, e restituisca in output una matrice W di dimensione $(K+1) \times (n+1)$ la cui riga i-esima contenga i coefficienti del polinomio $p_i(z)$, per $i = 0, \ldots, K$. Ricordiamo che n denota il grado di $p(x)$.*

Scrivere uno script che sfrutti la function appena definita per verificare sperimentalmente le proprietà 1 e 2 elencate sopra. Naturalmente dovrete scegliere polinomi con radici disposte in modo opportuno.

3. Esponenziale di una matrice e polinomi di Taylor (facoltativo)

Data una matrice A di dimensioni $n \times n$, l'esponenziale di A è la matrice

$$e^A = \sum_{k=0}^{\infty} \frac{A^k}{k!} = I + A + \frac{1}{2}A^2 + \frac{1}{6}A^3 + \ldots.$$

In Matlab l'esponenziale di `A` si calcola con il comando
`expm(A)`
(Attenzione: i comandi `expm(A)` e `exp(A)` sono entrambi ben definiti, ma calcolano due cose diverse! Provate a confrontare i risultati).

Vogliamo capire se l'approssimazione di e^A data dai polinomi di Taylor, cioè dalle serie troncate $p_m(A) = I + A + \frac{1}{2}A^2 + \ldots + \frac{1}{m!}A^m$ converge rapidamente ed è numericamente valida. In altre parole, vogliamo studiare numericamente la successione $\{s_m\}_{m\in\mathbb{N}}$, dove

$$s_m = \|e^A - p_m(A)\|_2.$$

È utile sapere che in Matlab la norma 2 di una matrice si calcola con il comando `norm` e il fattoriale di un numero intero con il comando `factorial`.

Esercizio 5 *Scrivere una function* `err=convergenza_exp(A,k)` *che prenda in ingresso una matrice quadrata* `A` *e un intero positivo* `k`*, e restituisca il vettore* `err` *dei primi* `k` *elementi della successione $\{s_m\}_{m\in\mathbb{N}}$.*

Scrivere poi uno script che, facendo uso della function appena definita, disegni in modo opportuno l'andamento degli errori di approssimazione di e^A nei casi seguenti:

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0.2 & 0 \\ 0 & 0 & 0.5 \end{bmatrix},$$

$$A = \begin{bmatrix} 1.5 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 5 \end{bmatrix},$$

$$A = \begin{bmatrix} 40 & 32 \\ 16 & 18 \end{bmatrix},$$

$$A = \begin{bmatrix} -49 & 24 \\ -64 & 31 \end{bmatrix}.$$

Che cosa osservate?