

## ESERCITAZIONE 5

### Polinomi

Per questa esercitazione vi viene chiesto di consegnare **uno a scelta tra gli esercizi 3 e 4**. Create un file `.tar` o `.zip` contenente il codice che risolve l'esercizio e caricatelo sulla pagina di e-learning del corso.

#### 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

$$\mathbf{p} = [1 \ 2 \ -5 \ 0 \ 1 \ -6]$$

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  $\mathbf{r}$  contenente le radici e calcoliamo i coefficienti del polinomio monico  $\mathbf{p}$  corrispondente, quindi verifichiamo che  $\mathbf{p}$  si annulli sulle radici assegnate e che il calcolo numerico delle radici di  $\mathbf{p}$  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)` che prenda in ingresso un vettore `p` contenente i coefficienti di un polinomio e un numero reale `t`, e disegni sul piano complesso gli zeri di `p` e gli zeri del polinomio ottenuto sommando `t` al coefficiente costante di `p`.

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

1.  $p(x) = x^4 - 1$ ,  $t = 0.02$ ,

2.  $q(x) = (x - 1)^4$ ,  $t = 0.02$ ,

e tracci (in una terza figura) i grafici dei due polinomi in un intorno di 1, per esempio nell'intervallo  $[0, 1.5]$ .

Che cosa si può constatare confrontando le radici dei polinomi di partenza e dei polinomi perturbati? In particolare, che differenze notate tra il comportamento dell'esempio 1 e dell'esempio 2? Che legame c'è con i grafici disegnati?

Nell'Esercizio 1 abbiamo verificato sperimentalmente come le radici multiple di un polinomio siano in generale mal condizionate: se  $p(x)$  ha una radice con molteplicità  $k$  e applico ai coefficienti di  $p(x)$  una perturbazione dell'ordine di  $\epsilon$ , la radice multipla si “spezza” in  $k$  radici distinte a distanza circa  $\epsilon^{\frac{1}{k}}$ .

Tuttavia, se ci restringiamo a opportune classi di perturbazioni sui coefficienti, la situazione migliora, come vedremo nel prossimo esercizio.

**Esercizio 2** Sia  $p(x) = (x-2)^3(x-1)$ . Vogliamo perturbare  $p(x)$  in due modi diversi e vedere sperimentalmente come cambiano le radici. Si scriva uno script in MATLAB che faccia quanto segue.

- (a) Si definisca il polinomio  $q(x) = p(x) + 0.05$  e se ne calcolino numericamente le radici. Rappresentare sul piano complesso, in una stessa figura, le radici di  $p(x)$  e le radici di  $q(x)$ .
- (b) Sia  $r(x) = (x-2)^3(x+1)$ . Si definisca il polinomio  $s(x) = p(x) + \frac{0.05}{\|r(x)\|_2} r(x)$ , dove  $\|r(x)\|_2$  denota la norma euclidea del vettore dei coefficienti di  $r(x)$ , e se ne calcolino numericamente le radici. Rappresentare sul piano complesso, in una stessa figura, le radici di  $p(x)$  e le radici di  $s(x)$ .

Che cosa osservate? La norma della perturbazione è la stessa nei due casi, ma dovrete constatare che il comportamento delle radici è piuttosto diverso.

Nell'esercizio che segue costruiamo due note famiglie di polinomi: Legendre e Chebyshev.

**Esercizio 3** (a) Polinomi di Legendre. I polinomi di Legendre sono definiti in modo ricorsivo come

$$\begin{aligned} p_0(x) &= 1, \\ p_1(x) &= x, \\ p_n(x) &= \frac{(2n-1)xp_{n-1}(x) - (n-1)p_{n-2}(x)}{n}, \quad n \geq 2. \end{aligned}$$

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  $[-1, 1]$ .

- (b) (Polinomi di Chebyshev). I polinomi di Chebyshev di prima specie sono definiti in modo ricorsivo come

$$\begin{aligned} p_0(x) &= 1, \\ p_1(x) &= x, \\ p_n(x) &= 2xp_{n-1}(x) - p_{n-2}(x), \quad n \geq 2. \end{aligned}$$

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  $n$ :

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

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.

Ogni iterazione ha l'effetto di incrementare la separazione delle radici interne ed esterne alla circonferenza unitaria. Al crescere di  $i$ , le radici interne tendono a 0, quelle esterne tendono all'infinito.

Se il polinomio iniziale  $p(x)$  ha  $s$  radici di modulo minore di 1 e  $n - s$  radici di modulo maggiore di 1, a quale polinomio convergerà la successione?

**Esercizio 4** Scrivere una function **graeffe** che prenda 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  $p_{i-1}$ , per  $i = 0, \dots, K$ . Ricordiamo che  $n$  denota il grado di  $p(x)$ .

Scrivere uno script che sfrutti la function appena definita per verificare sperimentalmente la proprietà di convergenza enunciata sopra:

- scegliete  $n$  ed  $s$  a vostro piacimento,
- costruite un polinomio  $p$  di grado  $n$  con  $s$  radici di modulo minore di 1 e  $n - s$  radici di modulo maggiore di 1,
- applicate la function **graeffe** definita sopra, per un opportuno valore di  $K$ ,
- stampate l'ultima riga della matrice  $W$  e una frase che commenti il risultato (per esempio "Il polinomio limite è ..."). Può esservi utile il comando **disp**.

Ricordiamo che l'iterazione di Graeffe è alla base di un metodo per il calcolo delle radici di polinomi, noto come methodo di (Dandelin-Lobachevsky-)Graeffe. Ne trovate una descrizione ad esempio su Wikipedia o nel sito Wolfram MathWorld.

### 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

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

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  $p_m(A) = I + A + \frac{1}{2}A^2 + \dots + \frac{1}{m!}A^m$  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?