Esame di Calcolo Numerico — 5 novembre 2022

Corso di Laurea in Ingegneria Chimica
Tempo a disposizione: 2 ore. È consentito consultare appunti e testi (cartacei).

Esercizio 1 (15 punti) Consideriamo la funzione $F : \mathbb{R}^2 \to \mathbb{R}^2$ data da

$$F(\mathbf{x}) = \begin{bmatrix} x_1^3 + 2x_2 - 1 \\ 2x_1^2 + x_2 + 2, \end{bmatrix}. \tag{1}$$

Vogliamo risolvere il sistema di due equazioni in due incognite dato da $F(\mathbf{x}) = 0$ utilizzando il metodo di Newton multivariato.

1. Trovare un'espressione per la matrice Jacobiana $J_F(\mathbf{x})$ di F.
2. Per $\mathbf{x} = \left[\begin{smallmatrix} 1 \\ 1 \end{smallmatrix}\right]$ e $\mathbf{h} = 10^{-6}\left[\begin{smallmatrix} 1 \\ -2 \end{smallmatrix}\right]$ calcolare $\|F(\mathbf{x}+\mathbf{h}) - F(\mathbf{x}) - J_F(\mathbf{x})\mathbf{h}\|_1$. Riportare sul foglio il codice Matlab utilizzato e il valore ottenuto. Quale dovrebbe essere l'ordine di grandezza della norma ottenuta? In base a quale risultato di analisi?
3. Scrivere una `function [x, n] = multivariate_newton(x0, epsilon)` che applica il metodo di Newton multivariato per calcolare una soluzione di $F(\mathbf{x}) = 0$, a partire dal valore iniziale $\mathbf{x}_0 \in \mathbb{R}^2$, arrestandosi quando è verificato il criterio di arresto $\|F(\mathbf{x}_n)\|_1 \leq \varepsilon$, e restituendo l'ultima iterata $\mathbf{x}_n$ e l'indice corrispondente n. Per risolvere i sistemi lineari che compaiono nel metodo, utilizzare l'operatore backslash (`\`) di Matlab. Se possibile, fare in modo che il valore di $F(\mathbf{x}_n)$ venga calcolato una volta sola per ogni n. Riportare sul foglio il codice della funzione.
4. Per $\mathbf{x}_0 = \left[\begin{smallmatrix} 1 \\ 1 \end{smallmatrix}\right]$, riportare i risultati $(\mathbf{x}_n, n)$ ottenuti eseguendo la funzione con i tre valori $\varepsilon = 10^{-3}, 10^{-6}, 10^{-12}$.

Esercizio 2 (15 punti) Consideriamo il problema ai valori iniziali

$$y' = -2ty, \quad y(0) = y_0 = 1, \quad [a, b] = [0, 1]. \tag{2}$$

1. Scrivere una `function [t, Y] = rkvariable(alpha, N)` che, dato in input il numero di intervalli N e il valore di $\alpha \in [0, 1]$, risolve il problema (2) utilizzando il metodo di Runge–Kutta con tavola di Butcher

$$\begin{array}{c|cc} 0 & 0 & 0 \\ \alpha & \alpha & 0 \\ \hline & 0 & 1 \end{array}$$

   La funzione deve restituire il vettore con gli estremi $t = [t_0, t_1, \ldots, t_N]$ degli N intervalli (uguali) di discretizzazione, e il valore delle approssimazioni corrispondenti $Y = [y_0, y_1, \ldots, y_N]$.
2. Chiamare la funzione con $\alpha \in \{\frac{1}{3}, \frac{1}{2}\}$ e $N \in \{30, 60\}$, riportando per ognuna delle quattro combinazioni dei parametri l'errore globale di discretizzazione $E_{N,\alpha} = \max_{n=1,2,\ldots,N} |y(t_n) - Y_n|$ tra la soluzione numerica calcolata e quella esatta (la soluzione esatta si può calcolare in Matlab con l'istruzione `y = exp(-t.^2)`). Cosa indicano i risultati ottenuti sull'ordine di convergenza del metodo per questi valori di α?
3. Qual è, in funzione di α, la funzione di stabilità $R(q)$ di questo metodo di Runge–Kutta?
4. Esistono valori di α per cui il metodo è A-stabile?

Soluzioni

Esercizio 1 (15 punti)

1. Con le usuali regole di derivazione, si ha

$$J_F(\mathbf{x}) = \begin{bmatrix} 3x_1^2 & 2 \\ 4x_1 & 1 \end{bmatrix}.$$

2.
```
>> F = @(x) [x(1)^3 + 2*x(2) - 1;   2*x(1)^2 + x(2) + 2];
>> JF = @(x) [3*x(1)^2  2;   4*x(1)   1];
>> x = [1; 1];
>> h = 1e-6 * [1; -2];
>> norm(F(x + h) - F(x) - JF(x) * h, 1)
ans =
   5.0006e-12
```

Il risultato ottenuto dovrebbe essere dell'ordine di $\|\mathbf{h}\|^2$, visto che da uno sviluppo di Taylor multivariato si ha

$$F(\mathbf{x}+\mathbf{h}) = F(\mathbf{x}) + JF(\mathbf{x})\mathbf{h} + O(\|\mathbf{h}\|^2).$$

3. Una possibile soluzione è la seguente.

```
function [x, n] = multivariate_newton(x0, epsilon)

F = @(x) [x(1)^3 + 2*x(2) - 1;   2*x(1)^2 + x(2) + 2];
JF = @(x) [3*x(1)^2  2;   4*x(1)   1];

x = x0;
Fx = F(x); % evita di dover ricalcolare F(x) due volte
n = 0;
while(norm(Fx, 1) > epsilon)
    x = x - JF(x) \ Fx;
    Fx = F(x);
    n = n + 1;
end
```

4. I risultati ottenuti sono i seguenti.

```
>> [x, n] = multivariate_newton([1;1], 1e-3)
x =
    4.2737
  -38.5299
n =
    19
>> [x, n] = multivariate_newton([1;1], 1e-6)
x =
    4.2737
  -38.5299
n =
    20
>> [x, n] = multivariate_newton([1;1], 1e-12)
x =
    4.2737
  -38.5299
n =
    21
```

Esercizio 2 (15 punti)

1. Una possibile soluzione è la seguente.

```
function [t, Y] = rkvariable(alpha, N)

f = @(t, x) -2*t*x;
a = 0;
b = 1;
h = (b-a) / N;
t = a:h:b;
Y = zeros(1, N+1);
Y(1) = 1;

for n = 1:N
    k1 = f(t(n), Y(n));
    k2 = f(t(n) + alpha*h, Y(n) + alpha*h*k1);
    Y(n+1) = Y(n) + h*k2;
end
```

2. I risultati ottenuti sono i seguenti.

```
>> [t, Y] = rkvariable(1/3, 30);
>> E30 = max(abs(Y - exp(-t.^2)))
E30 =
    0.0036
>> [t, Y] = rkvariable(1/3, 60);
>> E60 = max(abs(Y - exp(-t.^2)))
E60 =
    0.0018
>> E60 / E30
ans =
    0.5013
>> [t, Y] = rkvariable(1/2, 30);
>> E30 = max(abs(Y - exp(-t.^2)))
E30 =
  1.1607e-04
>> [t, Y] = rkvariable(1/2, 60);
>> E60 = max(abs(Y - exp(-t.^2)))
E60 =
  2.8545e-05
>> E60 / E30
ans =
    0.2459
```

   L'ordine di convergenza quindi è $p = 1$ per $\alpha = \frac{1}{3}$, e $p = 2$ per $\alpha = \frac{1}{2}$.

3. Si ha

$$\kappa_1 = \lambda y_n,$$
$$\kappa_2 = \lambda(y_n + h\alpha\kappa_1) = \lambda(y_n + h\alpha\lambda y_n),$$
$$y_{n+1} = y_n + h\kappa_2 = y_n + h\lambda(y_n + h\alpha\lambda y_n) = (1 + h\lambda(1 + \alpha h\lambda))y_n = (1 + q(1 + \alpha q))y_n.$$

   La funzione di stabilità quindi è

$$R(q) = 1 + q(1 + \alpha q) = 1 + q + \alpha q^2.$$

4. No: un teorema enunciato a lezione afferma che non esistono metodi di Runge–Kutta espliciti A-stabili, e il metodo che stiamo studiando è esplicito.