Calcolo Numerico - Corso B: Laboratorio Lezione 1

Luca Gemignani <luca.gemignani@unipi.it>

27 Febbraio 2019

1 Analisi degli errori

Iniziamo con un semplice esempio. Vogliamo analizzare gli effetti delle approssimazioni introdotte in macchina nel calcolo della funzione

$$f(x) = \frac{x-1}{x} = 1 - \frac{1}{x}, \quad x \neq 0.$$

Detta $\tilde{x} = x(1+\epsilon_x)$, $|\epsilon_x| \leq u$, l'approssimazione di macchina del dato x, indichiamo con

$$g_1(\tilde{x}) = (\tilde{x} \ominus 1) \oslash \tilde{x}, \quad g_2(\tilde{x}) = 1 \ominus (1 \oslash \tilde{x})$$

le funzioni effettivamente calcolate in macchina. Ponendo $x \odot y = x \cdot y(1+\epsilon)$, $|\epsilon| \leq u$, $\cdot \in \{+, -, \times, /\}$, otteniamo

$$g_1(\tilde{x}) = \frac{(x(1+\epsilon_x)-1)(1+\epsilon_1)}{x(1+\epsilon_x)}(1+\epsilon_2) \tag{1}$$

$$\doteq \frac{(x(1+\epsilon_x)-1)(1+\epsilon_1)(1-\epsilon_x)}{x}(1+\epsilon_2) \tag{2}$$

$$\doteq \frac{x(1+\epsilon_1+\epsilon_2) - 1(1+\epsilon_1+\epsilon_2-\epsilon_x)}{x}, \tag{3}$$

da cui

$$\epsilon_{TOT} = \frac{g_1(\tilde{x}) - f(x)}{f(x)} \tag{4}$$

$$\doteq \epsilon_1 + \epsilon_2 + \frac{1}{xf(x)}\epsilon_x. \tag{5}$$

Da questa espressione ricaviamo che l'errore totale si esprime in un'analisi al primo ordine –abbiamo trascurato le componenti di ordine superiore al primo negli errori–

come somma di due componenti distinte dovute rispettivamente all'amplificazione sul risultato finale dell'errore di rappresentazioned ϵ_x e degli errori locali delle operazioni ϵ_1, ϵ_2. Questa considerazione è generale. Consideriamo ad esempio il calcolo di $f(x)$ con l'algoritmo $g_2(\tilde{x})$. Abbiamo

$$g_2(\tilde{x}) = (1 - \frac{1}{x(1+\epsilon_x)}(1+\delta_1))(1+\delta_2) \tag{6}$$

$$\doteq 1(1+\delta_2) - \frac{1}{x}(1+\delta_1+\delta_2-\epsilon_x) \tag{7}$$

$$\tag{8}$$

da cui

$$\epsilon_{TOT} = \frac{g_2(\tilde{x}) - f(x)}{f(x)} \tag{9}$$

$$\doteq \delta_2 - \frac{1}{xf(x)}\delta_1 + \frac{1}{xf(x)}\epsilon_x. \tag{10}$$

Si osserva che l'errore totale si scrive nuovamente come somma di due componenti di cui quella relativa all'amplificazione dell'errore di rappresentazione risulta inalterata mentre la rimanente dipende dall'algoritmo considerato. In particolare se $\epsilon_x = 0$ il primo algoritmo può fornire risultati significativamente migliori. A titolo di esempio si consideri il caso seguente trattato in MatLab.

```
>> x=1+2^(-31)

x =

    1.000000000465661e+00

>> g1(x)

ans =

    4.656612870908988e-10

>> g2(x)

ans =

    4.656612873077393e-10

>> digits(32); a=sym(2^(-31)/(1+2^(-31)), 'd')

a =

0.00000000046566128709089882331539911319851
```

Passiamo ora ad un esempio meno accademico. Consideriamo il calcolo del prodotto scalare di due vettori $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ definito da

$$f(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x}^T \boldsymbol{y} = \sum_{i=1}^{n} x_i y_i.$$

La seguente funzione MatLab implementa l'algoritmo usuale.

```
function [s] = prod_sc(x,y)
n=length(x); s=0;
for k=1:n
    s=s+x(k)*y(k);
end
end
```

Studiamo separatamente le componenti dell'errore dovute all'amplificazione degli errori di rappresentazione e degli errori locali delle operazioni. Per la prima componente otteniamo che

$$f(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}) = \sum_{i=1}^{n} x_i y_i (1 + \epsilon_{x_i})(1 + \epsilon_{y_i}) \tag{11}$$

$$\doteq \sum_{i=1}^{n} x_i y_i (1 + \epsilon_{x_i} + \epsilon_{y_i}) \tag{12}$$

da cui otteniamo

$$\epsilon_{IN} = \frac{f(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}) - f(\boldsymbol{x}, \boldsymbol{y})}{f(\boldsymbol{x}, \boldsymbol{y})} \tag{13}$$

$$\doteq \frac{1}{f(\boldsymbol{x}, \boldsymbol{y})} \sum_{i=1}^{n} x_i y_i (\epsilon_{x_i} + \epsilon_{y_i}) \tag{14}$$

e quindi

$$|\epsilon_{IN}| \leq 2u \sum_{i=1}^{n} \frac{|x_i y_i|}{|f(\boldsymbol{x}, \boldsymbol{y})|}.$$

Per la seconda componente assumendo $\boldsymbol{x}$ e $\boldsymbol{y}$ vettori formati da numeri di macchina si ha che la funzione $g = g(\boldsymbol{x}, \boldsymbol{y})$ calcolata da `prod_sc` soddisfa

$$g(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}) = g(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{n} x_i y_i (1 + \epsilon_i) \prod_{j=i}^{n} (1 + \delta_j) \tag{15}$$

$$\doteq \sum_{i=1}^{n} x_i y_i (1 + \epsilon_i)(1 + \sum_{j=i}^{n} \delta_j) \tag{16}$$

dove ϵ_i sono glli errori locali commessi nell'operazione $x_i \otimes y_i = x_i y_i(1+\epsilon_i)$ e δ_i, $\delta_1 = 0$, sono gli errori locali commessi nell'operazione di somma

$$fl(s) \oplus x_i \otimes y_i = (fl(s) + x_i \otimes y_i)(1+\delta_i).$$

Si ricava dunque che

$$\epsilon_{ALG} = \frac{g(\boldsymbol{x},\boldsymbol{y}) - f(\boldsymbol{x},\boldsymbol{y})}{f(\boldsymbol{x},\boldsymbol{y})} \tag{17}$$

$$\doteq \frac{1}{f(\boldsymbol{x},\boldsymbol{y})} \sum_{i=1}^{n} x_i y_i (\epsilon_i + \sum_{j=i}^{n} \delta_j) \tag{18}$$

da cui

$$|\epsilon_{ALG}| \leq nu \sum_{i=1}^{n} \frac{|x_i y_i|}{|f(\boldsymbol{x},\boldsymbol{y})|}.$$

Si conclude quindi che la *stabilità dell'algoritmo* è assicurata per *problemi ben condizionati*. Alla stessa conclusione si perveniva con una tecnica differente detta di analisi dell'errore all'indietro (*backward error analysis*). In particolare si deduce che

$$g(\boldsymbol{x},\boldsymbol{y}) \doteq \sum_{i=1}^{n} x_i y_i (1 + \epsilon_i + \sum_{j=i}^{n} \delta_j) = f(\boldsymbol{x}, \widehat{\boldsymbol{y}}),$$

con

$$(\widehat{\boldsymbol{y}})_i = y_i(1 + \epsilon_i + \sum_{j=i}^{n} \delta_j) = y_i(1+\gamma_i), \quad |\gamma_i| \leq (n-i+1)u.$$

In altri termini il valore della funzione effettivamente calcolata coincide con il valore assunto dalla funzione esatta valutata su dati perturbati ("all'indietro" si riferisce al fatto che si vuole stimare la perturbazione sui dati in ingresso). La valutazione di ϵ_{ALG} è dunque ricondotta allo studio della sensibilità di f rispetto a perturbazioni dei dati iniziali, i.e.,

$$\epsilon_{ALG} = \frac{g(\boldsymbol{x},\boldsymbol{y}) - f(\boldsymbol{x},\boldsymbol{y})}{f(\boldsymbol{x},\boldsymbol{y})} = \frac{f(\boldsymbol{x},\widehat{\boldsymbol{y}}) - f(\boldsymbol{x},\boldsymbol{y})}{f(\boldsymbol{x},\boldsymbol{y})}.$$

Questo tipo di analisi all'indietro ben si adatta allo studio della stabilità degli algoritmi dell'algebra lineare numerica in cui tipicamente si hanno molti dati in ingresso. A tal proposito lo studente consideri il problema della risoluzione di un sistema triangolare 2×2 mediante la procedura –di sostituzione all'indietro– descritta di seguito

```
function [x1,x2] = solve_tri2(a,b,c,e1,e2)
%%risolve [a,b; 0, c][x1; x2]=[e1; e2]
x2=e2/c;
x1=(e1-b*x2)/a;
end
```

Si mostri che la soluzione calcolata in macchina a partire da dati in input di macchina è la soluzione esatta di un problema perturbato. Si determinino le maggiorazioni sulle perturbazioni dei dati in ingresso. Lo studente verifichi inoltre che questa proprietà non implica l'accuratezza del risultato provando il seguente set di problemi

```
delta=1.0e-8;
[x1,x2]=solve_tri2(delta, 1, 1-1/delta, 1+delta, 1-1/delta);
```

per differenti valori del parametro δ.