Lezione 6 (ore 9,10) - 2 ottobre 2025, 8:30 - 10:30 F3

(1.31) Definizione (funzioni predefinite).

Sia $M = F(\beta,m)$ l'insieme dei numeri di macchina del calcolatore in esame, e rd la funzione arrotondamento in M. L'insieme FP delle *funzioni predefinite*, ovvero delle funzioni che il calcolatore sa calcolare operando con gli elementi di M è costituito da tre classi.

- L'insieme delle funzioni predefinite corrispondenti ad *operazioni aritmetiche*. Se $\cdot$ è una delle operazioni aritmetiche tra numeri reali +, -, $\times$, / allora la funzione predefinita corrispondente si indica con il simbolo $\odot$ (un cerchietto contenente il simbolo dell'operazione considerata) ed è definita, per ogni coppia ξ, ϑ di elementi di $F(\beta,m)$ facenti parte del dominio dell'operazione $\cdot$, da

$$\xi \odot \vartheta = rd(\xi \cdot \vartheta)$$

- L'insieme delle funzioni predefinite corrispondenti alle usuali *funzioni elementari* (sen, cos, arcsen, arccos, ln, exp ...). Se $f:A \to R$ è una delle funzioni elementari allora la funzione predefinita corrispondente si indica con il simbolo F ed è definita, per ogni elemento ξ di $F(\beta,m)$ facente parte del dominio A della funzione elementare f, da

$$F(\xi) = rd(f(\xi))$$

- L'insieme delle funzioni predefinite corrispondenti ai *confronti* tra numeri reali ($<$, $\leqslant$, $=$, $\neq$, $\geqslant$, $>$). In questo caso, poiché gli elementi di $F(\beta,m)$ *sono* numeri reali, essi vengono confrontati come tali. Quindi le funzioni predefinite corrispondenti ai confronti sono semplicemente le restrizioni a $F(\beta,m) \times F(\beta,m)$ dei confronti tra numeri reali (e non è necessario introdurre simboli nuovi per indicarle).

(1.32) Definizione (algoritmo, algoritmo ingenuo).

Siano $f_1,...,f_k$ funzioni elementari o operazioni aritmetiche e sia $f:A \to R$, con A un opportuno sottoinsieme di R, la funzione ottenuta *componendo* $f_1,...,f_k$:

$$f(x) = f_1 \circ ... \circ f_k(x)$$

(ad esempio: $f(x) = sen(x) + cos(x)$, dove $f_3(x) = sen(x)$, $f_2(x) = cos(x)$ e $f_1(x_1,x_2) = x_1 + x_2$). Se chiediamo a *Scilab* di valutare la funzione f con l'istruzione

```
> f(x)
```

il valore restituito sarà

$$F_1 \circ ... \circ F_k(rd(x))$$

dove $F_1,...,F_k(x)$ sono, rispettivamente, le funzioni predefinite corrispondenti a $f_1,...,f_k(x)$.

L'espressione $F_1 \circ ... \circ F_k(rd(x))$ definisce una funzione φ: $A \to M$ detta *algoritmo ingenuo* per f (per la funzione dell'esempio: φ(x) = SEN(rd(x)) $\oplus$ COS(rd(x)), definita per ogni x in R). Con il termine *algoritmo* ci si riferisce, in generale, ad una sequenza *finita* di operazioni di calcolo di funzioni predefinite.

Salvo casi molto particolari, ci saranno valori di x per i quali f(x) $\neq$ φ(x). In questi casi si utilizza φ(x) per approssimare f(x) ed è interessante avere *informazioni sull'errore commesso*.

Per ottenere queste informazioni introduciamo le nozioni di *algoritmo accurato*, *algoritmo stabile* e di *calcolo ben condizionato del valore di una funzione*.

(1.33) Definizione (algoritmo accurato).

Siano f:A → R una funzione, φ:A → M l'algoritmo utilizzato per approssimare i valori di f e x $\in$ A.

L'algoritmo φ si dice *accurato* (quando utilizzato per approssimare il valore di f in x) se esiste un numero reale ε tale che:

(1) $\varphi(x) = (1 + \varepsilon)\, f(x)$
(2) ε 'piccolo'

Se l'algoritmo è accurato per ogni x $\in$ B $\subset$ A, si dirà che l'algoritmo è accurato in B. In tal caso ε dipenderà da x.

(1.34) Osservazione.

- Siano f ed x tali che f(x) $\neq$ 0. La (1) della Definizione precedente è *equivalente* alla seguente:

$$\varepsilon = \frac{\varphi(x) - f(x)}{f(x)}$$

  In questo caso dunque, l'algoritmo è accurato equivale a dire che l'errore relativo commesso approssimando f(x) con φ(x) è 'piccolo'.

- Se l'algoritmo è accurato si ha: f(x) = 0 $\Leftrightarrow$ φ(x) = 0.

- La definizione di algoritmo accurato è *qualitativa* perché non si quantifica il termine 'piccolo' relativo ad ε. Il significato concreto del termine 'piccolo' dipende caso per caso. Ad esempio, se, come nel caso del metodo di bisezione, interessa soltanto che φ(x) e f(x) abbiano lo stesso segno, ε 'piccolo' significa $\varepsilon > -1$.

  Esercizio: Si approssima una L > 0 con λ. Che errore relativo ε si commette utilizzando λ = 0? Quale valore di λ si deve usare per ottenere un errore relativo ε = 1?

(1.35) Definizione (algoritmo stabile).

Siano f:A → R una funzione, φ:A → M l'algoritmo utilizzato per approssimare i valori di f e x $\in$ A.

L'algoritmo φ si dice *stabile* (quando utilizzato per approssimare il valore di f in x) se esistono numeri reali ε_a, ε_v tali che:

(1) $\varphi(x) = (1 + \varepsilon_v)\ f((1 + \varepsilon_a)x)$

(2) ε_a, ε_v 'piccoli'

Se l'algoritmo è stabile per ogni $x \in B \subset A$, si dirà che l'algoritmo è stabile in B. In tal caso ε_a, ε_v dipenderanno da x.

(1.36) Osservazione.

- Se un algoritmo è accurato allora è stabile ($\varepsilon_a = 0$, $\varepsilon_v = \varepsilon$) ma *non* viceversa.
- Informalmente: un algoritmo stabile restituisce una *buona approssimazione* (ε_v 'piccolo') del valore di f in un punto *vicino* ad x (ε_a 'piccolo').

(1.37) Osservazione (algoritmo 'buono').

La nozione di stabilità formalizza l'idea di algoritmo '*buono*' per approssimare i valori di una data f. Ad esempio, se f è una funzione elementare e φ è l'algoritmo ingenuo per f allora, detta F la funzione predefinita corrispondente ad f, si ha:

$$\varphi(x) = F(\ rd(x)\ ) = rd(\ f(\ rd(x)\ )\ )$$

(1.38) Teorema (errore relativo e perturbazione).

Ricordando la definizione di errore relativo commesso approssimando un numero reale t con l'arrotondato rd(t) ed il Teorema (1.28) della Lezione 5 sulla limitazione dell'errore relativo, si ottiene:

Siano x un numero reale e rd la funzione arrotondamento in $F(\beta,m)$. Esiste un numero reale ε tale che:

$$rd(x) = (1 + \varepsilon)x \quad \text{e} \quad |\varepsilon| < u$$

L'uguaglianza esprime l'arrotondato di x come (piccola) *perturbazione moltiplicativa* di x.

(Dimostrazione: se $x \neq 0$ allora ε è l'errore relativo commesso approssimando x con rd(x); se x = 0 (e quindi rd(x) = 0) l'uguaglianza sussiste, ad esempio, con $\varepsilon = 0$.)

(1.39) Osservazione (continuazione della precedente).

Utilizzando due volte il Teorema precedente si ottiene infine:

$$\varphi(x) = (1 + \varepsilon_2)f(\ (1 + \varepsilon_1)x\ ) \quad \text{con} \quad |\varepsilon_1| < u \ \text{ e } \ |\varepsilon_2| < u$$

L'algoritmo φ restituisce la *migliore approssimazione possibile* del valore di f nel punto *più vicino possibile* ad x. In questo senso φ è l'algoritmo 'migliore possibile' che il calcolatore possa utilizzare per approssimare f(x). Da qui, generalizzando, l'idea che un algoritmo 'buono' per approssimare il valore di una funzione in un punto assegnato sia un algoritmo che restituisce una buona approssimazione del valore della funzione in un punto vicino a quello in cui si voleva calcolarla.

(1.40) Definizione (calcolo ben condizionato del valore di una funzione).

Siano $f:A \to R$ una funzione e $x \in A$. Il calcolo del valore di f in x è *ben condizionato* se: *per ogni* numero reale α 'piccolo' *esiste* un numero reale ε_V 'piccolo' tale che

$$f(\,(1+\alpha)x\,) = (1+\varepsilon_V)f(x)$$

Informalmente: il calcolo del valore di f in x è ben condizionato se il valore di f in *ogni* punto 'vicino' ad x è una 'buona' approssimazione del valore di f in x.

(1.41) Osservazione.

- La proprietà che il calcolo del valore di f in x sia ben condizionato riguarda *esclusivamente* la funzione f. In particolare, non è legata a *quale algoritmo* si sceglie per approssimare i valori di f.
- Se $f(x) \neq 0$, il valore di ε_V, una volta assegnato α, è *determinato*. Precisamente, ε_V risulta:

$$\varepsilon_V = \frac{f(\,(1+\alpha)x\,) - f(x)}{f(x)}$$

(1.42) Teorema (stabilità + buon condizionamento => accuratezza).

Siano $f:A \to R$ una funzione, $x \in A$ e φ l'algoritmo utilizzato per approssimare f(x). Se l'algoritmo è *stabile* e il calcolo di f in x è *ben condizionato* allora l'algoritmo è *accurato*.

Dimostrazione. Per la stabilità dell'algoritmo esistono ε_1 e ε_2 tali che:

$$\varphi(x) = (1+\varepsilon_2)f(\,(1+\varepsilon_1)x\,) \quad \text{con} \quad \varepsilon_1 \text{ e } \varepsilon_2 \text{ 'piccoli'}$$

Per il buon condizionamento del calcolo di f in x esiste ε_3 tale che:

$$f(\,(1+\varepsilon_1)x\,) = (1+\varepsilon_3)f(x) \quad \text{e} \quad \varepsilon_3 \text{ 'piccolo'}$$

Allora possiamo riscrivere:

$$\varphi(x) = (1+\varepsilon_2)(1+\varepsilon_3)f(x)$$

e, posto $(1+\varepsilon_2)(1+\varepsilon_3) = 1+t$, ovvero $t = \varepsilon_2+\varepsilon_3+\varepsilon_2\varepsilon_3$, si ottiene infine:

$$\varphi(x) = (1+t)f(x) \quad \text{con} \quad t \text{ 'piccolo'}$$

dunque l'algoritmo è accurato.

(1.43) <u>Osservazione</u> (stabilità degli algoritmi ingenui nei casi elementari).

- Per quanto ricavato nelle Osservazioni (1.37) e (1.39), se $f:A \to R$ è una funzione elementare e φ è l'algoritmo ingenuo per f, φ è stabile su A: *l'algoritmo ingenuo per ciascuna funzione elementare è stabile.*
- Sia $f(x_1,x_2) = x_1 + x_2$. L'algoritmo ingenuo per f è:

$$\varphi(x_1,x_2) = rd(x_1) \oplus rd(x_2)$$

Ricordando la definizione di $\oplus$ (vedi Definizione (1.31)) ed utilizzando tre volte il Teorema (1.38) si riscrive:

$$\varphi(x_1,x_2) = (1 + \varepsilon_3)(\ (1 + \varepsilon_1)x + (1 + \varepsilon_2)x\ ) \quad , \quad \text{con } |\varepsilon_j| \leqslant u\ ,\ j = 1,2,3$$

Dunque, l'algoritmo ingenuo per la somma è stabile.

Allo stesso modo si dimostra che *l'algoritmo ingenuo per ciascuna delle operazioni aritmetiche è stabile.*