5. Esempi di instabilità della sottrazione

Nel precedente capitolo abbiamo analizzato la risposta delle operazione aritmetiche agli errori sui dati. In particolare, dati $x, y \in \mathbb{R}$ con $\tilde{x} \approx x, \tilde{y} \approx y$, i risultati ottenuti si possono sintetizzare con la disuguaglianza $\epsilon_{x \star y} \leq w_1 \epsilon_x + w_2 \epsilon_y$ dove $\epsilon_{x \star y}$ è l'errore relativo al risultato dell'operazione $\star$ e ϵ_x, ϵ_y gli errori relativi sui dati.

Ci ricordiamo che *i pesi* $w_1, w_2 > 0$ possono dipendere da x, y, ma non dagli errori.
Nel caso di moltiplicazione, divisione e addizione si ha $w_1, w_2 \approx 1$ oppure $w_1, w_2 \leq 1$, quindi tali operazioni risultano **stabili**.
Nel caso della sottrazione i pesi

$$w_1 = \frac{|x|}{|x+y|}, \; w_2 = \frac{|y|}{|x+y|}$$

con $\text{sgn}(x) = -\text{sgn}(y)$ *possono* essere grandi.
Questo accade in particolare se x, y sono "vicini" in termini relativi, ovvero $|x+y| \ll |x|, |y|$, quindi la sottrazione è **potenzialmente instabile** e in grado di diminuire in modo consistente nel risultato la precisione dei dati e anche di distruggerla completamente, rendendo il risultato praticamente privo di significato.
Ad esempio quando $w_1, w_2 > \max \left\{ \frac{1}{\epsilon_x}, \frac{1}{\epsilon_y} \right\}$ ci si può attendere un errore $> 100\%$.

Faremo ora alcuni esempi significativi di perdita di precisione dovuta ad instabilità della sottrazione, lavorando come sempre per semplicità in sistemi floating-point virtuali in base 10.

Esempio 1

Consideriamo $\mathbb{F}(10, 4, L, U)$ con L, U sufficienti per rappresentare i numeri che ci interessano.
Siano $x = 0.10016$ e $y = -0.10012$. Allora

$$\tilde{x} = fl^4(x) = 0.1002$$
$$\tilde{y} = fl^4(y) = -0.1001$$

Eseguendo l'operazione-macchina di somma algebrica, che è una sottrazione visto che x e y hanno segno opposto, si ottiene:

$$x \oplus y = fl^4(fl^4(x) + fl^4(y)) = fl^4(0.1002 - 0.1001) = 10^{-4}$$

Invece $x + y = 4 \cdot 10^{-5}$.
Quindi l'errore relativo nel risultato è

$$\frac{|(x+y) - (\tilde{x} + \tilde{y})|}{|x+y|} = \frac{|4 \cdot 10^{-5} - 10^{-4}|}{|4 \cdot 10^{-5}|} = \frac{6 \cdot 10^{-5}}{4 \cdot 10^{-5}} = \frac{3}{2} = 150\%$$

A fronte di $\epsilon_x, \epsilon_y \leq \epsilon_M = \frac{10^{-3}}{2}$ abbiamo un errore del 150% e una perdita di precisione di ben 3 ordini di grandezza rispetto alla precisione di macchina.
Qui il problema sta nella sottrazione tra numeri vicini, visto che $x + y = 4 \cdot 10^{-5}$ ma $|x|, |y| \approx 10^{-1}$.
Infatti se calcoliamo i pesi

$$w_1, w_2 = \frac{|x|}{|x+y|} \approx \frac{10^{-1}}{4 \cdot 10^{-5}} = \frac{10^4}{4} = 2500$$

Questi fattori di amplificazione degli errori sui dati sono dell'ordine di 10^3 e spiegano come si arrivi ad un errore finale $> 100\%$, che rende nella pratica inaccettabile il risultato.
In questo caso i fattori di amplificazione sono enormi, ma non comunque $> \frac{1}{\epsilon_M}$.
Osserviamo che sarebbe bastata una cifra di mantissa in più per avere il risultato esatto, dal momento che non si necessitava più di fare arrotondamenti.
Questo apre la strada all'analisi del prossimo esempio, un pò più sofisticato.

Esempio 2

Consideriamo $\mathbb{F}(10, 8, L, U)$ con L, U opportuni e quindi $\epsilon_M = \frac{10^{-7}}{2}$. Si tratta di calcolare in questa aritmetica floating-point la somma algebrica $a + b + c$ dove

$$\begin{aligned} a &= +0.23371258 \cdot 10^{-4} \\ b &= +0.33678429 \cdot 10^{2} \\ c &= -0.33677811 \cdot 10^{2} \end{aligned}$$

Osserviamo che $fl^8(a + b + c) = 0.64137126 \cdot 10^{-3}$. Vedremo ora che in questo esempio *non* vale la *proprietà associativa*.
Infatti

$$I = (a \oplus b) \oplus c = 0.33678452 \cdot 10^2 \oplus (-0.33677811 \cdot 10^2) = 0.64100000 \cdot 10^{-3}$$

Invece

$$II = a \oplus (b \oplus c) = 0.3371258 \cdot 10^{-4} \oplus 0.61800000 \cdot 10^{-3} = 0.64137126 \cdot 10^{-3}$$

Quindi $I \neq II$; inoltre si verifica, ad esempio utilizzando Matlab, che utilizza 16 cifre decimali, che $II = fl^8(a + b + c)$. Cioè il risultato II è il meglio che si può ottenere in questa aritmetica floating-point, mentre l'errore relativo di I è

$$\frac{|I - (a + b + c)|}{|a + b + c|} \approx \frac{4 \cdot 10^{-7}}{0.6 \cdot 10^{-3}} = \frac{2}{3} \cdot 10^{-3} \approx 0.07\%$$

Qui riusciamo subito a spiegare la perdita di precisione di 4 ordini di grandezza rispetto a $\epsilon_M = \frac{10^{-7}}{2}$, calcolando i pesi w_1 e w_2 associati alla sottrazione $\underbrace{(a+b)}_{=x} + \underbrace{c}_{=y}$,

visto che x e y sono vicini in termini relativi, si noti che hanno le prime 4 cifre significative coincidenti

$$w_1 = \frac{|x|}{|x+y|} = \frac{|a+b|}{|a+b+c|} \approx \frac{0.3 \cdot 10^2}{0.6 \cdot 10^{-3}} = \frac{1}{2} \cdot 10^5 = 50000$$

> Si osservi che in queste stime abbiamo lavorato con una sola cifra significativa, sia con gli erori sia con i pesi, perchè di queste quantità non ci interessa il valore accurato ma solo l'ordine di grandezza.

Naturalmente, anche nel caso II c'è una sottrazione che viene fatta subito, ovvero $b+c$, quindi posto $x = b, y = c$ si ha

$$w_1 = \frac{|b|}{|b+c|} = \frac{0.3 \cdot 10^2}{0.6 \cdot 10^{-3}} = 50000$$

Ma allora perché con l'espressione di calcolo II non c'è perdita di precisione?
La spiegazione è sottile e sta nel fatto che i numeri a, b, c entrano con 8 cifre significative e non occorre arrotondarli, cioè $\epsilon_a = \epsilon_b = \epsilon_c = 0$.
Quindi la sottrazione $b+c$ non perde precisione. Invece nell'espressione di calcolo I la sottrazione avviene con uno dei due dati arrotondato, ovvero $a+b$, che è un'addizione il cui risultato viene comunque arrotondato perché ha più di 8 cifre significative e quell'errore di arrotondamento viene amplifcato dal peso w_1.
Osserviamo che in una sottrazione instabile *basta che uno dei due dati sia affetto da errore* per vedere la perdita di precisione.

Siamo in grado a questo punto di discutere l'esempio portato alla fine del capitolo 4, risponderemo alla domanda: *perché $f(10^{-15})$ in Matlab ha un errore $> 11\%$ e $f(2^{-50})$ è esatto, pur essendo $10^{-15} = 2^{-50}$?*

Esempio 3

Si consideri il calcolo in Matlab della funzione $f(x) = \dfrac{(1+x)-1}{x}, x \neq 0$. Chiaramente $f(x) = 1$, in Matlab però con le operazioni-macchina si calcola

$$\tilde{f}(x)\Big((1 \oplus x) \oplus (-1)\Big) \oslash x$$

e si ha $\tilde{f}(10^{-15}) = 1.11\ldots$. Cioè l'errore relativo nel calcolo di f è

$$\epsilon_{f(x)} = \frac{|f(x) - \tilde{f}(x)|}{|f(x)|} = |1 - 1.11\ldots| = 0.11\cdots > 11\%$$

La spiegazione sta nella sottrazione che avviene tra due numeri estremamente vicini, cioè $1+x$ e 1, le altre operazioni invece non danno problemi in quanto *stabili*.
Mentre 1 è un reale-macchina e non viene quindi arrotondato, $x = 10^{-15}$ viene arrotondato (non lo sarebbe se la base fosse 10, ma non bisogna mai dimenticare che in Matlab la base vera è 2), così come $1 + 10^{-15}$.

Questi piccolissimi errori di arrotondamento vengono però amplificati dal peso w_1 nella sottrazione $(1+x)-1, x=10^{-15}$.
Calcoliamo $w_1 = \dfrac{|1+x|}{|(1+x)-1|} = \dfrac{1+x}{x} \approx 10^{15}$ che spiega perfettamente come si siano persi 15 ordini di grandezza del calcolo di f.
D'altra parte il peso w_1 nel caso della sottrazione $(1+x)-1, x=2^{-50} \approx 10^{-15}$ è sempre dell'ordine di 10^{-15}. Per quale motivo allora $\tilde{f}(2^{-50})$ e $1+2^{-50}$ sono entrambi reali-macchina in base 2 e perciò *non* vengono arrotondati, cioè

$$\epsilon_{1+x} = \frac{|(1+x)-(1 \oplus x)|}{(1+x)} = 0$$

e il fattore di amplificatione non ha effetto.

Come ultimo esempio di possibile instabilità della sottrazione ci occuperemo ora della formula risolutiva per le equazioni di secondo grado, formula che siamo abituati ad usare senza farci problemi.
Scopriremo invece che in certe situazioni questa formula in aritmetica floating-point può perdere moltissima precisione, fino a far perdere del tutto di significato al risultato.
Vedremo però anche che la formula può essere convenientemente "stabilizzata" eliminando la sottrazione potenzialmente instabile.

Esempio 4

Consideriamo un'equazione di grado effettivo 2 $az^2+bz+c=0, a \neq 0$ nel caso di discriminante $\Delta = b^2-4ac > 0$, le cui soluzioni scriviamo abitualmente come $z_{\pm} = \dfrac{-b \pm \sqrt{\Delta}}{2a}$.
Ora, una di queste due soluzioni richiede una sottrazione, ovvero z_+ per $b>0$ e z_- per $b<0$: *prendiamo per semplicità* $b>0$, essendo l'analisi dell'altro caso del tutto analoga.

Osserviamo che in aritmetica floating-point $\sqrt{\Delta}$ sarà quasi sempre arrotondato, d'altra parte la funzione $\sqrt{\cdot}$ viene calcolata alla precisione di macchina in tutti i linguaggi di calcolo (con un metodo che vedremo più avanti chiamato *metodo delle tangenti* o *metodo di Newton*).
Quindi nella sottrazione $\sqrt{\Delta}-b$ il primo dato è sicuramente affetto da un errore al massimo dell'ordine della precisione di macchina.

Quando ci si aspetta che questa sottrazione possa far perdere precisione? Quando $\sqrt{\Delta}$ è vicina a b in termini relativi, cioè per $b^2 \gg |4ac| \implies \sqrt{\Delta} \approx b$.
Infatti posto $x=\sqrt{\Delta}, y=-b$, i pesi sono:

$$w_1 = \frac{|x|}{|x+y|} = \frac{\sqrt{\Delta}}{|\sqrt{\Delta}-b|} = \frac{\sqrt{\Delta}(\sqrt{\Delta}+b)}{|\sqrt{\Delta}-b|(\sqrt{\Delta}+b)} = \frac{\sqrt{\Delta}(\sqrt{\Delta}+b)}{\Delta-b^2} = \frac{\sqrt{\Delta}(\sqrt{\Delta}+b)}{|\cancel{b^2}-(\cancel{b^2}-4ac)|}$$

ma $\sqrt{\Delta} \approx b$ quindi

$$w_1 = \frac{\sqrt{\Delta}(\sqrt{\Delta} + b)}{|4ac|} \approx \frac{2b^2}{|4ac|} = \frac{b^2}{2|ac|}$$

e analogamente $w_2 \approx \frac{b^2}{2|ac|}$.
Siccome il rapporto $\frac{b^2}{2|ac|}$ può diventare arbitrariamente grande, perchè dipende dagli ordini di grandezza dei coefficienti a, b, c, la soluzione z_+ può subire una fortissima perdita di precisione in aritmetica floating-point, fino a renderla priva di significato. Facciamo a questo proposito un paio di esempi.

Esempio 4.1

Consideriamo l'equazione $z^2 + 100z - 1 = 0$ cioè $a = 1, b = 100, c = -1$ con $\Delta = 100^2 + 4 = 10004$.
Nel sistema floating-point vistuale $\mathbb{F}(10, 4, L, U)$ con L, U opportuni (qui il parametro chiave è il numero di cifre di mantissa = 4).
Con 4 cifre di mantissa Δ viene arrotondato a 10000:
$fl^4(\Delta) = fl^4(0.10004) \cdot 10^5 = (0.1000) \cdot 10^5$.
Quindi viene calcolato $\sqrt{\Delta} = 100$. Quindi in questo sistema viene calcolata $\tilde{z}_+ = 0$ con un errore relativo del 100%. Infatti

$$\frac{|z_+ - \tilde{z}_+|}{|z_+|} = \frac{|z_+|}{|z_+|} = 1$$

In un certo senso questo è un caso limite, per la scarsità di cifre di mantissa si è costretti ad approssimare con 0 una quantità che non è nulla, facendo direttamente un errore relativo del 100%.
Per vedere l'effetto dei coefficienti di amplificazione

$$w_1, w_2 \approx \frac{b^2}{2|ac|} \approx \frac{10^4}{2} = 5000$$

consideriamo $\mathbb{F}(10, 8, L, U)$ quindi con $\epsilon_M = 10^{-7}/2$. In questo caso si può verificare che $\tilde{z_+} = 10^{-2}$ mentre il valore "esatto" è $z_+ = 0.0099990002$, per cui abbiamo un errore relativo

$$\frac{|z_+ - \tilde{z}_+|}{|z_+|} \approx 10^{-4}$$

Abbiamo quindi perso 3 ordini di precisione rispetto alla precisione di macchina, il che è ben spiegato dai pesi w_1, w_2 che sono dell'ordine di 10^3.

> Osserviamo che l'errore commesso è $\approx 10^{-4}$, cioè circa dello 0.01%.

D'altra parte tale errore, che potrebbe essere comunque accettabile in molti contesti applicativi, è 2000 volte più grande della precisione di macchina. Se avessimo lavorato con 16 cifre di mantissa, ci saremmo aspettati un'amplificazione degli errori di

arrotondamento fino a $5000 \cdot \epsilon_M = 5000 \cdot \dfrac{10^{15}}{2} \approx 10^{-12}$ e quindi avremmo ottenuto una precisione $< \epsilon_M$ ma comunque molto elevata.

Non è difficile convincersi che ci vuole poco per mettere in crisi qualsiasi sistema floatin-point, basta che cambino i rapporti tra gli ordini di grandezza dei coefficienti.

Esempio 4.2

Consideriamo infatti $10^{-2}z^2 + 10^4 z + 10^{-2} = 0$ in $\mathbb{F}(10, 16, L, U)$, sostanzialmente la precisione di Matlab.
Si osservi che qui con 8 cifre di mantissa saremmo stati di nuovo nella situazione di approssimazione perchè Δ ha 12 cifre significative.

Con 16 cifre di mantissa si calcola

$$\tilde{z}_+ = \frac{-10^4 + \sqrt{10^8 - 4 \cdot 10^{-4}}}{2 \cdot 10^{-2}} = -(0.9999894\ldots46) \cdot 10^{-6}$$

mentre la soluzione "esatta", arrotondata a 16 cifre è $z_+ = -(0.1000000000001000) \cdot 10^{-5}$ con un errore relativo $\approx 1.1 \cdot 10^{-5}$ e una perdita di precisione di 10 ordini di grandezza rispetto a $\epsilon_M = \dfrac{10^{-15}}{2}$, che è spiegabile con i fattori di amplificazione

$$w_1, w_2 \approx \frac{b^2}{2|ac|} = \frac{10^8}{2 \cdot 10^{-4}} = \frac{1}{2} \cdot 10^{12}$$

Questi esempi mostrano che la formula risolutiva classica delle equazioni di secondo grado è molto instabile quando $b^2 \gg 4|ac|$, ovver quando $\sqrt{\Delta} \approx |b|$, a causa di una sottrazione intrinseca nel modo in cui è scritta.

In qusto caso però il problema si può risolvere riscrivendo la formula con un "trucchetto" algebrico: consideriamo nuovamente il caso $b > 0$

$$z_+ = \frac{\sqrt{\Delta} - b}{2a} = \frac{(\sqrt{\Delta} - b)(\sqrt{\Delta} + b)}{2a(\sqrt{\Delta} + b)} = \frac{\Delta - b^2}{2a(\sqrt{\Delta} + b)} = -\frac{4ac}{2a(\sqrt{\Delta} + b)} = -\frac{2c}{\sqrt{\Delta} + b}$$

in $\mathbb{R}$ le 2 formule sono equivalenti; in $\mathbb{F}$ invece, la formula $z_+ = -\dfrac{2c}{\sqrt{\Delta} + b}$ diventa stabile perchè è stata eliminata la sottrazione. D'altra parte $z_- = -\dfrac{b + \sqrt{\Delta}}{2a}$ è stabile perchè non contiene sottrazioni.
Possiamo a questo punto scrivere una formula risolutiva *stabilizzata* che tiene conto del segno di b ovvero

$$\begin{cases} z_1 = -\operatorname{sgn}(b) \cdot \dfrac{2c}{|b| + \sqrt{\Delta}} \\ z_2 = -\operatorname{sgn}(b) \cdot \dfrac{|b| + \sqrt{\Delta}}{2a} \end{cases}$$

Tornando all'esempio *4.2*, si ha che $\tilde{z}_+$ fa un errore relativo $\approx 10^{-5}$ su z_+ come abbiamo visto, mentre, chiamato $\tilde{z}_1$ il valore di z_1 colocato in $\mathbb{F}$, si ha $\tilde{z}_1 = fl^{16}(z_+)$, cioè il calcolo di z_+ è stato completamente stabilizzato.

Qualcuno potrebbe oservare che anche il calcolo di Δ può contenere una sottrazione, precisamente quando $\mathrm{sgn}(a) = \mathrm{sgn}(c)$ e che questa può diventare instabile quando $b^2 \approx 4ac$, o meglio quando $b^2, 4ac \gg |b^2 - 4ac|$. Questa situazione è molto più difficile da trattare e non ce ne occuperemo.
Diciamo solo che qui la perdita di precisione non è completamente eliminabile, ma esiste un algoritmo grazie al quale l'errore relativo sulle due soluzioni per $\Delta > 0$ con $b^2 \approx 4ac$ è dell'ordine di $\epsilon_M^{1/3}$ cioè con perdita di precisione consistente ma controllata.