ΑΠΑΝΤΗΣΕΙΣ - ΣΧΟΛΙΑΣΜΟΣ ΠΡΟΟΔΟΥ ΕΠΙΣΤΗΜΟΝΙΚΟΥ ΥΠΟΛΟΓΙΣΜΟΥ Ι *11/2005*

I.

1. Στην τάξη περιγράψαμε και χρησιμοποιήσαμε εκτενώς ένα υπολογιστικό μοντέλο που προβλέπει το χρόνο λύσης με βάση το πλήθος των πράξεων αριθμητικής κινητής υποδιαστολής (α.κ.υ.) Ω και μεταφορών Φ. Να περιγράψετε συνοπτικά 4 σημαντικές απλοποιήσεις που έχουν γίνει σ' αυτό το υπολογιστικό μοντέλο σε σχέση με την πραγματικότητα.

   Απάντηση. Μερικές από τις απλοποιήσεις που έγιναν στο μοντέλο: α) οι πράξεις α.κ.υ. εκτελούνται με ίδιο κόστος ενώ δεν λαμβάνεται υπόψη το κόστος των υπόλοιπων πράξεων· β) Το μοντέλο χρησιμοποιεί ένα μόνον επίπεδο κρυφής μνήμης· γ) θεωρείται ότι το κόστος μεταφοράς στην περίπτωση `load` είναι ίδιο με την περίπτωση `store`· δ) Δεν λαμβάνεται υπόψη το γεγονός ότι μπορεί να επικαλύπτονται πράξεις μεταφοράς και αριθμητικές πράξεις· ε) Ο προγραμματιστής δεν ελέγχει ποια στοιχεία πηγαίνουν και διατηρούνται στην κρυφή μνήμη· στ) στην περίπτωση miss, δεν μεταφέρεται μόνον ένα στοιχείο στην κρυφή μνήμη (αλλά ολόκληρη «γραμμή κρυφής μνήμης»). □

2. Να υπολογίσετε τις τιμές $\Omega, \Phi_{\min}$ για τις πράξεις $y \leftarrow y + Ax$ και $A \leftarrow A + xy^\top$ (όπου $A \in \mathbb{R}^{n\times n}$ και $x, y, z \in \mathbb{R}^n$) και να εξηγήσετε γιατί ανήκουν στην κατηγορία BLAS-2.

   Απάντηση. Για το `MV` $y \leftarrow y + Ax$: $\Omega = n(2n-1), \Phi_{\min} = n^2 + 3n$. Για το `GER` $A \leftarrow A + xy^\top$: $\Omega = 2n^2, \Phi_{\min} = 2n^2 + 2n$. Και στις δυο περιπτώσεις, οι πράξεις και οι ελάχιστες μεταφορές είναι τετραγωνικές ως προς το n. Επίσης, μπορούν να γραφτούν και οι δυο ως $A \leftarrow A + BC$ με μόνο μια διάσταση από τις 3 που συνυπάρχουν στη γενική αυτή μορφή είναι 1. Γι' αυτό κατηγοριοποιήθηκαν στις BLAS-2. □

II.

1. Έστω ότι χρησιμοποιούμε αριθμητική IEEE διπλής ακρίβειας (δηλ. $1.*\cdots*$ όπου υπάρχουν 52 bits μετά το δεκαδικό «.» ενώ το πρώτο ψηφίο, που είναι 1 λόγω κανονικοποίησης, είναι κρυμμένο) στην οποία ισχύει η συνθήκη ακριβούς στρογγύλευσης και ότι επιλέγουμε να χρησιμοποιήσουμε «αποκοπή» και όχι «στρογγύλευση προς τον πλησιέστερο» α.κ.υ. Εξηγήστε ποια θα είναι η μονάδα στρογγύλευσης $\mathbf{u}$;

   Απάντηση. Η μονάδα στογγύλευσης προκύπτει από το μέγιστο σχετικό σφάλμα κατά τη στρογγύλευση. Στην περίπτωση της αποκοπής, αν το πραγματικό στοιχείο είναι $x = 1.*\cdots**\star\cdots$, κρατούμε τις πρώτες $t = 53$ θέσεις μετά την αποκοπή ενώ απορρίπτουμε τις υπόλοιπες (που συμβολίζουμε με το $\star$). Επομένως η μονάδα στρογγύλευσης θα είναι

$$\begin{aligned}\mathbf{u} &= \max_x \frac{|x - \mathrm{fl}(x)|}{|x|} = \frac{|0.0\cdots0\star\cdots|}{1.0} \\ &= 2^{-t} + 2^{-t-1} + \cdots = 2^{-t}(1 + 2^{-1} + \cdots) = 2^{-t}\frac{1}{(1-1/2)} = 2^{-t+1} = 2^{-52}.\end{aligned}$$

   (Σημ. Για ορθότερο μαθηματικό συμβολισμό θα μπορούσαμε, αντί το μέγιστο «max», να χρησιμοποιήσουμε το ελάχιστο άνω φράγμα «sup».) □

2. Έστω ότι οι μεταβλητές `realmax, realmin` περιέχουν αντίστοιχα το μέγιστο και ελάχιστο κανονικοποιημένο α.κ.υ. Τότε να περιγράψετε το αποτέλεσμα των παρακάτω πράξεων:

   i) `realmax+ realmin`. *ii*) `realmin/0`. *iii*) `realmin/2 == 0`. *iv*) `realmax+realmax/2`.

   Απάντηση. *i*) `realmax+ realmin = realmax` λόγω κανονικοποίησης των εκθετών. Σημειώστε επίσης ότι αν γράψουμε `realmax+ realmin = realmax (1+realmin/realmax)` το αποτέλεσμα ακολουθεί άμεσα αφού παρατηρήσουμε ότι `realmin/realmax < eps` όπου eps είναι το έψιλον της μηχανής. Μια ακόμα δυνατή απάντηση, σε περίπτωση που ερμηνεύσετε ότι ο realmin είναι - σε αντίθεση με την πρακτική στη MATLAB - ο αλγεβρικά ελάχιστος αναπαραστήσιμος αριθμός, είναι 0 (δηλ. `realmin = -realmax`. *ii*) `realmin/0`. Από τον ορισμό των ειδικών αριθμών, το αποτέλεσμα θα είναι το `Inf`. Αν όμως, στην προηγούμενη ερώτηση,

απαντήσατε 0, η απάντησή σας εδώ πρέπει να είναι `-Inf`. *iii*) `realmin/2 == 0`. Εφόσον χρησιμοποιούμε όλα τα στοιχεία της α.κ.υ., που συμπεριλαμβάνει τη βαθμιαία υποχείλιση, το αποτέλεσμα της πράξης `realmin/2` δεν θα είναι 0, επομένως η απάντηση θα είναι λογικό 0 «λάθος». *iv*) `realmax+realmax/2`. Η πράξη οδηγεί σε αριθμό που υπερβαίνει το μέγιστο αναπαραστώμενο α.κ.υ. και επιστρέφει `Inf`. Προσέξτε ότι αυτή η απάντηση διαφέρει από το αποτέλεσμα της (*i*) γιατί εδώ η κανονικοποίηση των εκθετών δεν μηδενίζει τον παράγοντα `realmax/2`. □

3. Να υπολογίσετε καλό φράγμα για το εμπρός απόλυτο σφάλμα στο τελικό αποτέλεσμα του αλγορίθμου

```
s=0; for j = 1:4, s = s+ x(j)y(j); end;
```

για τον υπολογισμό του `DOT` δυο διανυσμάτων α.κ.υ. υπό τον όρο ότι η υλοποίηση χρησιμοποιεί εντολές `FMA`. (Υπόδειξη: Στην `FMA` εμπλέκεται το πολύ ένα σφάλμα στρογγύλευσης ανά εντολή).

Απάντηση. Στη συνέχεια θεωρούμε ότι s και $\mathrm{fl}(s)$ είναι το ακριβές και το υπολογισμένο αποτέλεσμα, αντίστοιχα. Έστω ότι η εντολή `FMA` καλείται ως εξής: `s = FMA(t,a,b)` και επιστρέφει `s = t+a*b`. Τότε η πράξη μπορεί να αναλυθεί ως εξής:
`s=0; for j = 1:4, s = FMA(s,x(j),y(j)); end;` Τα σφάλματα θα είναι ως εξής:

$$\begin{aligned}\mathrm{fl}(s) &= ((((s+x(1)y(1))(1+\delta_1)+x(2)y(2))(1+\delta_2)+x(3)y(3))(1+\delta_3)+x(4)y(4))(1+\delta_4)\\ &= x(1)y(1)(1+\theta_4)+x(2)y(2)(1+\theta_3)+x(3)y(3)(1+\theta_2)+x(4)y(4)(1+\theta_1)\end{aligned}$$

όπου, ως συνήθως, $|\delta_j|\le \mathbf{u}$ και $|\theta_j|\le \gamma_j = j\mathbf{u}/(1-j\mathbf{u})$. Επομένως το απόλυτο σφάλμα φράσσεται ως εξής:

$$\begin{aligned}|\mathrm{fl}(s)-s| &= |x(1)y(1)\theta_4+x(2)y(2)\theta_3+x(3)y(3)\theta_2+x(4)y(4)\theta_1|\\ &\le \gamma_4\sum_{j=1}^{4}|x(j)y(j)|.\end{aligned}$$

Προσέξτε ότι <u>δεν είναι σωστή</u> η ανισότητα

$$|\mathrm{fl}(s)-s| \le |s|\gamma_4.$$

□

III. Θέλουμε να χρησιμοποιήσουμε *πίσω ανάλυση σφάλματος* για να υπολογίσουμε φράγμα για το εμπρός σχετικό σφάλμα στον υπολογισμό του τετραγώνου της ευκλείδειας νόρμας με τον παρακάτω αλγόριθμο:

```
s=x(1)*x(1); for j = 2:n, s = s+ x(j)*x(j); end;
```

όπου κάθε στοιχείο `x(j)` είναι αριθμός κινητής υποδιαστολής και ότι $x(j)*x(j) < \mathtt{Inf}$.

1. Να υπολογίσετε το δείκτη κατάστασης του προβλήματος και μια καλή εκτίμηση για το πίσω σφάλμα του αλγόριθμου.

   Απάντηση. Από την υπόθεση, δεν υπάρχει υπερχείλιση. Το αποτέλεσμα s είναι μια συνάρτηση του διανύσματος x: $f(x)=\sum_{j=1}^{n}\xi_j^2$, όπου για συντομία γράφουμε ξ_j για το $x(j)$. Επομένως, $f:\mathbb{R}^n\to\mathbb{R}^+$, όπου $\mathbb{R}^+$ συμβολίζει τους μη αρνητικούς αριθμούς. Επομένως, η Ιακωβιανή της f θα είναι μητρώο (διάνυσμα) $J\mathbb{R}^{1\times n}$:

$$\begin{aligned}J &= [\frac{\partial f}{\partial\xi_1},\ldots,\frac{\partial f}{\partial\xi_n}]\\ &= [2\xi_1,\ldots,2\xi_n]=2x.\end{aligned}$$

   Ο δείκτης κατάστασης του προβλήματος σε ένα σημείο υπολογίζεται ως εξής:

$$\mathrm{cond}(f;x)=\|J\|\frac{\|x\|}{\|s\|}=\|2x\|\frac{\|x\|}{|s|}=2\frac{\|x\|^2}{s}$$

αφού το s είναι μη αρνητικός βαθμωτός. Επομένως, αν επιλέξουμε την ευκλείδεια νόρμα, τα παραπάνω απλοποιούνται καθώς $s = \|x\|_2^2$ και ο δείκτης κατάστασης γίνεται $\mathrm{cond}(f;x) = 2$.

Για το πίσω σφάλμα, από γνωστή ανάλυση έχουμε:

$$\mathrm{fl}(s) = \xi_1^2 < n > +\xi_2^2 < n > +\xi_3^2 < n-1 > +\cdots \xi_n^2 < 2 >$$

όπου εδώ $< j >= \prod_{k=1}^{j}(1+\delta_k)$. Επομένως το αποτέλεσμα μπορεί να θεωρηθεί ως το ακριβές άθροισμα τετραγώνων των στοιχείων του διανύσματος

$$\tilde{x} := [\xi_1(1+\theta_n)^{1/2}, \xi_2(1+\theta_n)^{1/2}, \xi_3(1+\theta_{n-1})^{1/2}, ..., \xi_n(1+\theta_2)^{1/2}]$$

Τα στοιχεία του $\tilde{x}$ είναι κοντά στα αντίστοιχα του x. Ειδικότερα,

$$\begin{aligned} \|\tilde{x}-x\| &= \|[\xi_1(1+\theta_n)^{1/2}-\xi_1, \xi_2(1+\theta_n)^{1/2}-\xi_2, \xi_3(1+\theta_{n-1})^{1/2}-\xi_3, ..., \xi_n(1+\theta_2)^{1/2}-\xi_n]\| \\ &\leq \gamma_n\|x\| \end{aligned}$$

□

2. Να υπολογίσετε καλό φράγμα για το εμπρός σχετικό σφάλμα στο υπολογισμένο αποτέλεσμα s.

   Απάντηση. Από τη θεωρία, θα ισχύει ότι

$$\frac{|\mathrm{fl}(s)-s|}{|s|} \leq 2\gamma_n.$$

□

IV.

1. Να τροποποιήσετε τον παρακάτω βρόχο ώστε ο υπολογισμός να χρησιμοποιεί «ξετύλιγμα βρόχου» με βάθος 5. Ο νέος κώδικας πρέπει να λειτουργεί ορθά ανεξάρτητα από το (μη αρνητικό) μέγεθος του n.

```
s = 0.0; for j=1:n, s = s+x(j)*y(j); end
```

   Απάντηση. Θα θεωρήσουμε, όπως στη MATLAB , ότι η συνάρτηση $\mathrm{rem}(x,y)$ επιστρέφει το υπόλοιπο της διαίρεσης δυο αριθμών x,y. Ο βρόχος θα είναι ως εξής:

```
m = rem(n,5); s = 0.0;
for j=1:m, s = s+x(j)*y(j); end
for j=m+1:5:n
  s = s+x(j)*y(j)+x(j+1)*y(j+1)+x(j+2)*y(j+2)+x(j+3)*y(j+3)+x(j+4)*y(j+4);
end
```

□

2. Θέλουμε να υπολογίσουμε το $y \leftarrow y + A^k x$, όπου $k > 1$ είναι θετικός ακέραιος, $A \in \mathbb{R}^{n\times n}$ και $x, y \in \mathbb{R}^n$. Να γράψετε απλή υλοποίηση και να υπολογίσετε τα $\Omega, \Phi_{\min}$ της διαδικασίας (προσοχή, η υλοποίηση θα πρέπει να έχει πλήθος αριθμητικών πράξεων πολύ μικρότερο του kn^3).

   Απάντηση. Ένας τρόπος να γραφτεί η διαδικασία είναι:

LOAD A, x, y
for $kstep = 1 : k$
  for $i = 1 : n$
   $z(i) = 0$
   for $j = 1 : n$
    $z(i) = z(i) + A(i,j)x(j)$
   end
  end
  $x = z$
end
$y = y + x$;
STORE y

Το συνολικό κόστος είναι $\Omega = 2kn^2 + n$ και $\Phi_{\min} = n^2 + 3n$. Προσέξτε ότι υπολογισμός γίνεται ως εξής: $y = y + A(A(\cdots A(Ax)\cdots)$ αξιοποιώντας το γεγονός ότι γενικά, είναι πολύ πιο φθηνό να υπολογίζουμε μητρώο επί διάνυσμα ($\Omega = 2n^2$) αντί μητρώο επί μητρώο ($\Omega = 2n^3$). Δηλαδή, το παραπάνω αναμένεται να είναι πολύ πιο φθηνό από μια λογαριθμικού δένδρου οργάνωση των υπολογισμών, που με κατάλληλη επιλογή των παρενθέσεων, θα είχε μορφή: $y = y + (\cdots((A^2)(A^2))\cdots(A^2)(A^2))\cdots)$ και θα κόστιζε $O(n^3 \log_2 k)$. Σημειώνουμε επίσης ότι μπορούμε να εξοικονομήσουμε πράξεις με το εναλλακτικό σχήμα:

```
LOAD A,x,y
for kstep = 1 : k
  for i = 1 : n
   z(i) = A(i,1) * x(1)
   for j = 2 : n
    z(i) = z(i) + A(i,j)x(j)
   end
  end
  x = z
end
y = y + x;
STORE y
```

οπότε $\Omega = kn(2n-1) + n$. □

3. Σε συνέχεια της προηγούμενης ερώτησης, να υπολογίσετε το Φ για υλοποίηση σε σύστημα με κρυφή μνήμη μεγέθους $K = O(n)$.

   Απάντηση. Στόχος είναι να χρησιμοποιήσουμε το ελάχιστο Φ (κάτω φράγμα είναι το $\Phi_{\min}$). Δυστυχώς, δεν είναι δυνατόν να επιτύχουμε $\Phi = \Phi_{\min}$ με κρυφή μνήμη $O(n)$, καθώς δεν είναι δυνατή η διατήρηση του A μεταξύ των επαναλήψεων του εξωτερικού βρόχου.

```
LOAD x
for kstep = 1 : k
  for i = 1 : n
   z(i) = 0
   for j = 1 : n
    LOAD A(i,j)
    z(i) = z(i) + A(i,j)x(j)
   end
  end
  x = z
end
for i = 1 : n
  y(i) = y(i) + x(i);
end
STORE y
```

Το παραπάνω, υλοποιείται εύκολα με κρυφή μνήμη μεγέθους $3n$ (για τα x, y, z). Με την παραπάνω πολιτική, $\Phi = 3n + kn^2$. □