Quiz #5
Επιστημονικός Υπολογισμός – Άνοιξη 2009
ΠΡΟΣΕΓΓΙΣΗ ΣΥΝΑΡΤΗΣΕΩΝ ΚΑΙ ΔΕΔΟΜΕΝΩΝ
ΜΕΘΟΔΟΣ ΕΛΑΧΙΣΤΩΝ ΤΕΤΡΑΓΩΝΩΝ και FFT

Όνομα:

1. Πως υπολογίζεται το μέγεθος μιας συνάρτησης; Βρέστε το μέγεθος της παρακάτω συνάρτησης,$f(x) = \sin(x)\exp(\cos x)$
2. Περιγράψετε το πρόβλημα ελαχίστων τετραγώνων για μια ευθεία και λύστετο για τα παρακάτω δεδομένα.
$$,\begin{pmatrix} 1 & 10 & 20 & 30 & 40 \\ 1 & 100 & 400 & 600 & 1200 \end{pmatrix}$$
3. Περιγράψετε το πρόβλημα ΕΤ για ένα πολυώνυμο 2ου βαθμού και λύστετο για τα δεδομένα της άσκησης 2
4. Να διατυπωθεί το πρόβλημα ελαχίστων τετραγώνων για προσεγγιστικές συναρτήσεις τύπου
$p(x) = \sum_{k=0}^{n} c_k B_k(x)$ και να ορισθούν οι εξισώσεις που την ορίζουν σε συνεχή και διακριτή μορφή.
5. Πότε λέμε ότι μια βάση ενός συναρτησιακού χώρου είναι ορθογώνια;
6. Θεωρείστε την προσεγγιστική συνάρτηση
$S_n(x) = a_0/2 + \sum_{k=1}^{n}(a_k\cos(kx) + b_k\sin(kx))$
Ποια είναι η βάση του χώρου αυτών των συναρτήσεων; Είναι ορθογώνια;
Για την βάση αυτή αποδείξτε τις σχέσεις
$$\int_0^{2\pi} \mathrm{B}_\kappa(x)\mathrm{B}_l(x)dx = \begin{cases} 2\pi & k = l = 0 \\ \pi & k = l = 1,2,..n \\ 0 & k\#l \end{cases} ,\sum_{i=0}^{m} \mathrm{B}_\kappa(x_i)\mathrm{B}_l(x_i)dx = \begin{cases} 2\pi & k = l = 0 \\ \pi & k = l = 1,2,..n \\ 0 & k\#l \end{cases}$$
Αν τα σημεία x_i είναι ισαπέχοντα στο διάστημα $[0,2\pi)$. Να βρεθεί η λύση του ΕΤ γαι αυτές τις συναρτήσεις.
7. Τι ονομάζουμε "καλλίτερη" προσέγγιση g^* μιας συνάρτησης ή σύνολου δεδομένων;
8. Να διατυπωθεί το πρόβλημα ΕΤ για την συνάρτηση $f(x,y)$ και το πολυώνυμο $p(x,y) = a + bx + cy$
9. Να διατυπωθεί το πρόβλημα ΕΤ για την συνάρτηση $y = f(x)$ με γραμμικά τμηματικά πολυώνυμα.
10. Έστω μια συνάρτηση $y = f(x)$ με περίοδο T. Να μετασχηματιστεί σε μια συνάρτηση $y = g(x)$ με περίοδο 2π.
11. Να προσεγγισθεί η συνάρτηση $f(x) = 0.2x^2\exp(\sin x^2)$ στο διάστημα $[-\pi,\pi]$ με τριγονωμετρικό πολυώνυμο που παρεμβάλη f σε
8 ισαπέχοντα σημεία $\{(x_i,y_i)\}_{j=0}^{7}$ με $x_j = -\pi + j\frac{\pi}{4}$ και $y_j = f(x_j)$.
12. **FFT (FAST FOURIER TRANSFORM)**

FFT είναι ένας από τους ποιό εύχρηστους αλγορίθμους με εφαρμογές στα επεξεργασία σημάτων, φωνής και ήχου, ραντάρ. Το απλούστερο μαθηματικό πρόβλημα στο οποίο εφαρμόζεται για την λύση του ο "γρήγορος" υπολογισμός ενός πολυωνύμου σε ένα σύνολο σημείων. **Εφαρμόστε τον αλγόριθμο FFT για τον υπολογισμό ενός πολυωνύμου βαθμού** $< m$ $A(x) = a\_0 + a\_1 * x + a\_2 * x\^2 + \ldots + a\_\{m-1\} * x\^\{m-1\}$ **σε** m **σημεία της επιλογή σας έτσι ώστε ο χρόνος υπολογισμού είναι** $\mathrm{O}(m \log m)$**.**

Υποθέστε ότι $m = 2^n$ για ευκολία. Η βασική ιδέα του αλγορίθμου περιγράφεται παρακάτω

ΙΔΕΑ 1: Ορίζουμε τα πολυώνυμα

$$A\_even(x) = a\_0 + a\_2 * x + a\_4 * x\^2 + \ldots + a\_\{m-2\} * x\^\{m/2-1\}$$
$$A\_odd(x) = a\_1 + a\_3 * x + a\_5 * x\^2 + \ldots + a\_\{m-1\} * x\^\{m/2-1\}$$

με βαθμό $< m/2$, και παρατηρούμε ότι: $A(x) = A\_even(x\^2) + x * A\_odd(x\^2)$.

ΙΔΕΑ 2: Επιλέγουμε να υπολογίσουμε το A στα σημεία

$$1, w, w\^2, w\^3, \ldots, w\^\{m-1\}$$

όπου w είναι η κύρια ρίζα της μονάδος δηλαδή της εξίσωσης: $w\^\{m/2\} = -1$

Σύμφωνα με την ΙΔΕΑ 1 θα υπολογίσουμε A_even και A_odd στα σημεία

$$1, w\^2, w\^4, w\^6, \ldots, (w\^\{m/2-1\})\^2$$

Σημειώστε ότι για $w\^m = 1$ επαναλαμβάνουμε τις διαφορετικές ρίζες.

Εφαρμόζοντας την ιδέα 1 αναδρομικά, υπολογίζουμε πολυώνυμα βαθμού $< m/2$ στις δυνάμεις

$w\^2$, που είναι η $m/2$-th ρίζα της μονάδος, και μετά συνδιάζουμε τα αποτέλεσματα για να βρούμε τις τιμές του A σε χρόνο $\mathrm{O}(m)$.

Σημειώστε ότι ο χρόνος για την εκτέλεση του παραπάνω αλγορίθμου είναι $T(m) = 2T(m/2) + O(m)$, όπου $T(m)$ είναι ο χρόνος για τον υπολογισμό πολυψνύμου βαθμού < m στις δυνάμεις της mth ρίζας της μονάδος.

12.1. Να αποδειχθεί ότι $T(m) = \mathrm{O}(m \log m)$

12.2. Να εφαρμοσθεί ο αλγόριθμος στο πολυώνυμο
$A(x) = 2 + 3x + 4x\^2 + 5x\^3 + 6x\^4 + 7x\^5 + 8x\^6 + 9x\^7$.

12.3. Να υλοποιηθεί ο παρακάτω αλγόριθμος σε MATLAB και C η Java και να συγκριθεί με την αντίστοιχη υπορουτίνα της MATLAB (δέστε σημεώσεις).

THE FFT ALGORITHM

Define w to be the "primitive mth root of unity".
–> w = e^{2*pi*i/m}
–> w^m = 1, but w_j != 1 for j < m.
We'll pick:
x_0 = 1, x_1 = w, x_2 = w^2, ..., x_{m-1} = w^{m-1},
Given: array A of length m (representing coefficients of polynomial A(x))
Goal: produce DFT F(A): evaluation of A at 1, w, w^2,...,w^{m-1} where
w is primitive mth root of unity.
1. If m=1, just return (a_0)
2. let A_even = (a_0, a_2, ..., a_{m-2})
A_odd = (a_1, a_3, ..., a_{m-1})
3. Let V_even = F(A_even), V_odd = F(A_odd), w = primitive mth root of unity.
4. For j=0 to m/2-1, let
V[j] = V_even[j] + w^j V_odd[j]
V[m/2+j] = V_even[j] - w^j V_odd[j]
or, in pseudo-C:
Let A be array of length m, w be primitive mth root of unity.
Goal: produce DFT F(A): evaluation of A at 1, w, w^2,...,w^{m-1}.

```
FFT(A, m, w)
{
allocate space for output vector V.
if (m==1) return vector V = (a_0)
else {
A_even = (a_0, a_2, ..., a_{m-2})
A_odd = (a_1, a_3, ..., a_{m-1})
V_even = FFT(A_even, m/2, w^2)
V_odd = FFT(A_odd, m/2, w^2)
for (j=0; j < m/2; ++j) {
V[j] = V_even[j] + w^j*V_odd[j]
V[j+m/2] = V_even[j] - w^j*V_odd[j]
```