**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*)

Ορίζουμε στο [0,2π]

Άρα μέγεθος: max |f(x)| με 0 ≤ x ≤ 2π

**2.** Περιγράψετε το πρόβλημα ελαχίστων τετραγώνων για μια ευθεία και λύστετο για τα

παρακάτω δεδομένα.

,

1 10 20 30 40

1 100 400 600 1200

g=a+bx

min Σ4Κ=0 (yi -(a+bxi))2 = φ

a,b

*=0*

*=0*

**3.** Περιγράψετε το πρόβλημα ΕΤ για ένα πολυώνυμο 2ου βαθμού και λύστετο για τα

δεδομένα της άσκησης 2

g=a0+a1x+a2x2

*=0*

*=0*

*=0*

(yk-( a0+a1x+a2x2))2

a0,a1,a2

**4.** Να διατυπωθεί το πρόβλημα ελαχίστων τετραγώνων για προσεγγιστικές συναρτήσεις

τύπου

*p*(*x*) = Σ*k*=0

*n ckBk*(*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

*Sn*(*x*) = *a*0/2 + Σ*k*=1

*n* (*ak* cos(*kx*)  *bk* sin(*kx*))

Ποια είναι η βάση του χώρου αυτών των συναρτήσεων; Είναι ορθογώνια;

Για την βάση αυτή αποδείξτε τις σχέσεις

0

2**Β*κ**x*Β*l**x**dx*  

2* k*  *l*  0

* k*  *l*  1, 2, . .*n*

0 *k*#*l*

,Σ*i*0

*m* Β*κ**xi*Β*l**xi**dx*  

2* k*  *l*  0

* k*  *l*  1, 2, . .*n*

0 *k*#*l*

Αν τα σημεία *xi* είναι ισαπέχοντα στο διάστημα 0,2**. 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 2*π*.

**11.** Να προσεγγισθεί η συνάρτηση *f*(*x*) = 0.2*x*2 exp(sin *x*2) στο διάστημα [−*π*, *π*] με

τριγονωμετρικό πολυώνυμο που παρεμβάλη *f* σε

8 ισαπέχοντα σημεία {(*xi*, *yi*)}*j*=0

7 με *xj* –*π,* *j π/4*

και *yj*  *f**xj*.

**12. FFT (FAST FOURIER TRANSFORM)**

FFT είναι ένας από τους ποιό εύχρηστους αλγορίθμους με εφαρμογές στα επεξεργασία

σημάτων, φωνής και ήχου, ραντάρ. Το απλούστερο μαθη\_\_\_\_\_\_\_\_\_\_ματικό 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 . . . *a*\_*m* − 1 ∗ *x*^*m* − 1 **σε** *m* **σημεία της**

**επιλογή σας έτσι ώστε ο χρόνος υπολογισμού είναι** Ο*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 . . . *a*\_*m* − 2 ∗ *x*^*m*/2 − 1

*A*\_*odd**x*  *a*\_1  *a*\_3 ∗ *x*  *a*\_5 ∗ *x*^2 . . . *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, . . . ,*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, . . . , *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 Α σε χρόνο Ο*m*.

Σημειώστε ότι ο χρόνος για την εκτέλεση του παραπάνω αλγορίθμου είναι

*T**m*  2*T**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*  Ο*m*log*m*

**12.2.** Να εφαρμοσθεί ο αλγόριθμος στο πολυώνυμο

*A**x*  2  3*x*  4*x*^2  5*x*^3  6*x*^4  7*x*^5  8*x*^6  9*x*^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]