ΑΡΙΣΤΟΤΕΛΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΟΝΙΚΗΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Ψηφιακά Φίλτρα

Εργασία 2
Προσαρμογή στο Πεδίο της Συχνότητας

ΧΑΤΖΗΘΩΜΑ ΑΝΤΡΕΑΣ

ΑΕΜ: 8026

antreasc@ece.auth.gr

Υπεύθυνος καθηγητής: κ. Πιτσιάνης Νίκος
Ημερομηνία Παράδοσης: 08/05/2016

Ερώτημα 1

Στο 1ο ερώτημα συμπληρώθηκε ο κώδικας στο αρχείο $fftproof.m$ χρησιμοποιώντας τον ορισμό του διακριτού μετασχηματισμού Fourier (DFT) ο οποίος δίνεται στην εκφώνηση. Τρέχοντας λοιπόν τον αλγόριθμο, εμφανίζονται στην κονσόλα τα παρακάτω:

```
>> fftproof
DFT : 6.575202e-15
split output top bottom : 6.575202e-15
split input even odd : 6.623297e-15
using DFT definition: 2.882307e-15
done : 2.310224e-15
```

Εύκολα παρατηρεί κανείς πως το σφάλμα είναι πάρα πολύ μικρό και σχετικά κοντά στις υπόλοιπες υλοποιήσεις. Εξαιρουμένου της τελευταίας υλοποίησης, το σφάλμα είναι ελάχιστα μικρότερο απ' όλες τις υπόλοιπες.

Ερώτημα 2

(a) Στο 2ο ερώτημα, αρχικά γράφτηκε μια αναδρομική συνάρτηση στο MATLAB που υπολογίζει τον FFT για μια οποιαδήποτε είσοδο της οποίας το μήκος είναι δύναμη του δύο. Η αναδρομική αυτή σχέση γράφτηκε με τη βοήθεια 2 sites των οποίων τα URL τους υπάρχουν υπό τη μορφή σχολίων στον κώδικα του προγράμματος. Ο κώδικας βρίσκεται στο αρχείο $recursiveFFT.m$.

Ο έλεγχος ορθότητας έγινε με τη χρήση της συνάρτησης $fft(x)$ του MATLAB. Χρησιμοποιώντας ένα τυχαίο σήμα βλέπουμε ότι το σφάλμα της υλοποίησης $recursiveFFT$ είναι αρκετά μικρό. Αυτό φαίνεται και παρακάτω στο screenshot που πάρθηκε από την κονσόλα του MATLAB.

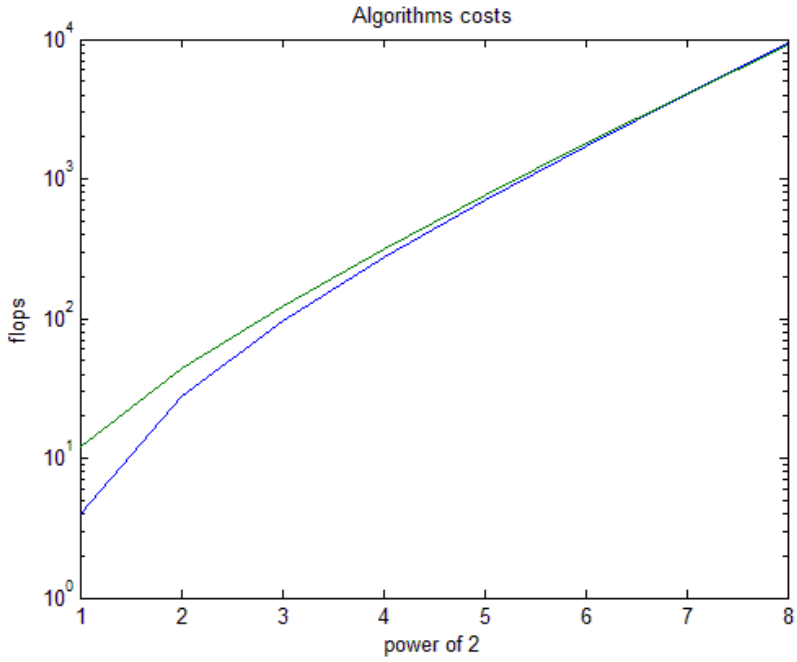
```
>> signal = (cos(4*pi*(1:8)/8+sin(2*pi*(1:8)/8)));
>> y=recursiveFFT(signal);
>> ys=fft(signal);
>> norm(y-ys)

ans =

   9.6737e-16
```

(b) Ακολούθως, γράφτηκε ο αλγόριθμος υπολογισμού του κόστους σε flops λαμβάνοντας υπόψη ότι η πρόσθεση μιγαδικών αντιστοιχεί σε 2 flops και ο πολλαπλασιασμός μιγαδικών αντιστοιχεί σε 6 flops. Ο υπολογισμός έγινε βάσει της αναδρομικής σχέσης που δόθηκε στην εκφώνηση. Ο κώδικας βρίσκεται στο αρχείο $T.m$. Η ορθότητα του υπολογισμού του κόστους ελέγχθηκε χρησιμοποιώντας τον αλγόριθμο recursiveFFT ο οποίος αναφέρθηκε πιο πάνω.

$\Longrightarrow$ $To\ υπολογιστικό\ κόστος\ και\ στις\ 2\ περιπτώσεις\ είναι\ περίπου\ το\ ίδιο.$

Αυτό φαίνεται και από το διάγραμμα που ακολουθεί αλλά και από το screenshot με τα αποτελέσματα από την εκτέλεση των 2 αλγορίθμων (recursive και θεωρητικός) δίνοντας διαδοχικά διάφορα τυχαία σήματα δύναμης του 2. Ο έλεγχος του υπολογιστικού κόστους αλλά και η δημιουργία του γραφήματος, δημιουργούνται καλώντας το script που βρίσκεται στο αρχείο $calc\_costs.m$.

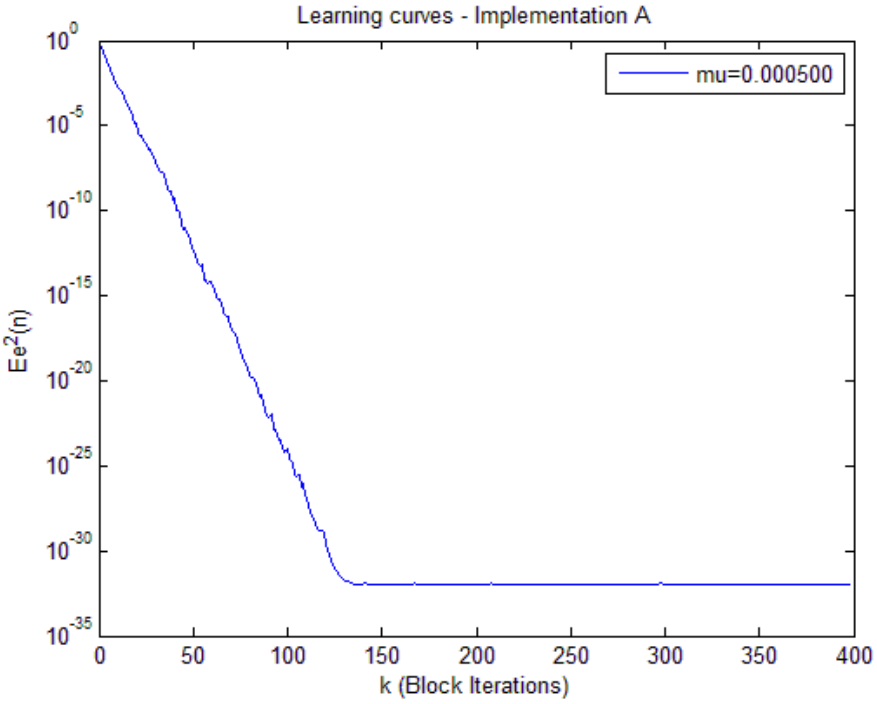
```
T_cost =

           4
          28
          96
         272
         704
        1728
        4096
        9472


recursive_cost =

          12
          44
         124
         316
         764
        1788
        4092
        9212
```



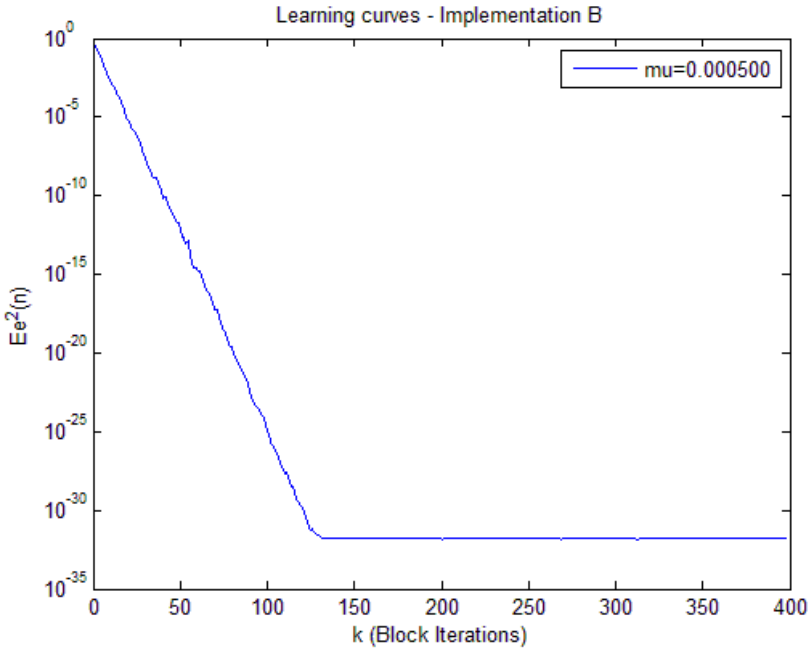
Ερώτημα 3

Στο 3ο ερώτημα έγινε χρήση ενός προσαρμοσμένου φίλτρου 3000 συντελεστών για την μοντελοποίηση ενός άγνωστου, γραμμικού συστήματος. Το σύστημα βρίσκεται στο αρχείο $plant.p$. Η προσαρμογή του φίλτρου υλοποιήθηκε σε 4 διαφορετικές υλοποιήσεις. Η κάθε υλοποίηση αλγόριθμου αποθηκεύτηκε σε ξεχωριστό αρχείο με όνομα $BlockLMS\_k.m$, όπου το $k = a, b, c, d$ ανάλογα με την υλοποίηση που αντιπροσωπεύει το συγκεκριμένο αρχείο. Η παράμετρος μ για τις 2 πρώτες υλοποιήσεις, επιλέχθηκε πιλοτικά με $\mu = 0.0005$ ενώ για τις 2 τελευταίες υλοποιήσεις η παράμετρος επιλέγεται δυναμικά εντός του αλγόριθμου. Παρακάτω παρουσιάζονται οι 4 διαφορετικές υλοποιήσεις καθώς και τα διαγράμματα που απεικονίζουν τις καμπύλες εκμάθησης (learning curves).

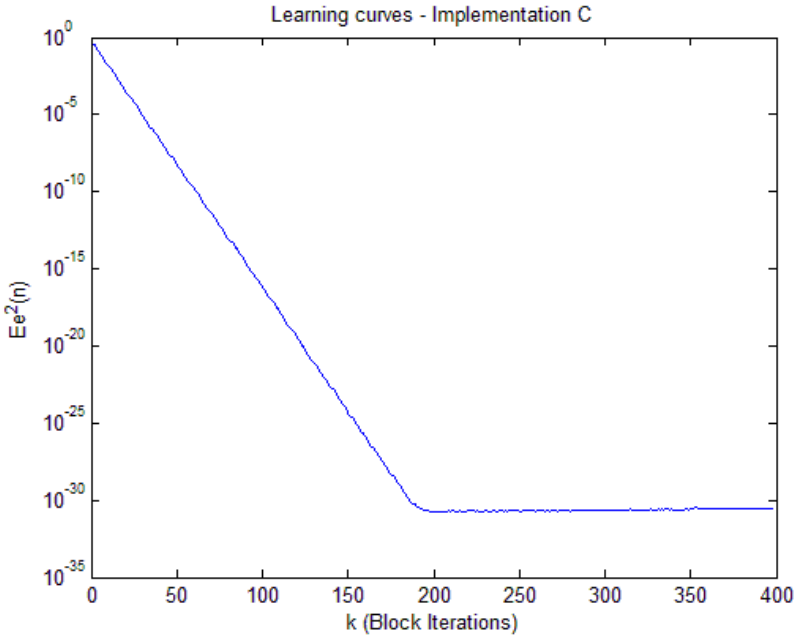
(α) Με δύο εμφωλευμένους βρόχους (nested loops)



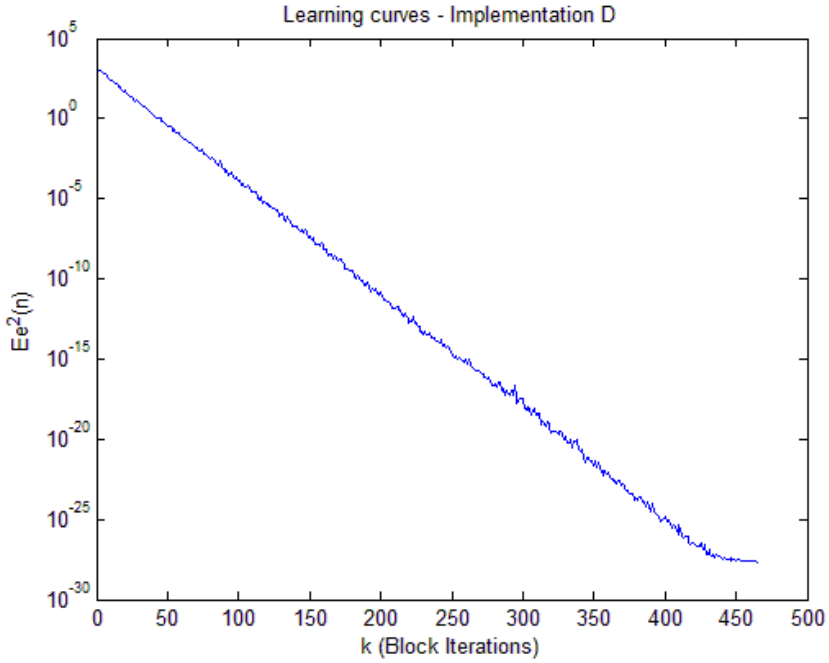
(β) Με ένα βρόχο και πράξεις πινάκων



(γ) Με προσαρμογή στο πεδίο της συχνότητας κάνοντας χρήση του FFT



(δ) Με μη περιορισμένη (unconstrained) προσαρμογή στο πεδίο της συχνότητας



Στο παρακάτω screenshot παρατίθενται κάποιοι ενδεικτικοί χρόνοι εκτέλεσης για την κάθε υλοποίηση ξεχωριστά. Οι χρόνοι πάρθηκαν χρησιμοποιώντας της εντολές $tic - toc$ του MATLAB.

```
>> BlockLMS_a
Elapsed time is 112.431085 seconds.
>> BlockLMS_b
Elapsed time is 142.039077 seconds.
>> BlockLMS_c
Elapsed time is 42.354324 seconds.
>> BlockLMS_d
Elapsed time is 50.596523 seconds.
```

Από τα παραπάνω διαγράμματα αλλά και τους χρόνους εκτέλεσης καταλήγουμε σε κάποια βασικά συμπεράσματα όσον αφορά την απόδοση και το υπολογιστικό κόστος της κάθε υλοποίησης. Καταρχάς, και οι 4 υλοποιήσεις συγκλίνουν στο ελάχιστο σφάλμα. Συγκεκριμένα, οι 2 πρώτες υλοποιήσεις (α, β) φτάνουν στο σφάλμα μετά από σχετικά μικρό αριθμό βημάτων συγκριτικά με τις υπόλοιπες 2 υλοποιήσεις (γ, δ). Παρόλα αυτά, οι υλοποιήσεις α και β απαιτούν περισσότερο χρόνο εκτέλεσης από τις υλοποιήσεις γ και δ.

Επίσης, οι 3 πρώτες υλοποιήσεις (α, β, γ) συγκλίνουν περίπου στο $10^{-31} \sim 10^{-32}$ ενώ η υλοποίηση δ συγκλίνει περίπου στο 10^{-26}. Οι υλοποιήσεις α και β απαιτούν περίπου 125 βήματα, η υλοποίηση γ περίπου 200 και η τελευταία υλοποίηση περίπου 440 βήματα.