

## Περιγραφή συστήματος

Όλα τα πειράματα υλοποιήθηκαν σε φορητό υπολογιστή Toshiba Qosmio X770-10N με τα εξής χαρακτηριστικά.

Επεξεργαστής Intel(R) Core™ i7-2630 QM CPU 2.00/2.90(Turbo boost) GHz

Ο εν λόγω επεξεργαστής διαθέτει 4 πυρήνες(8 εικονικούς) και η συχνότητα λειτουργίας του είναι 2.00 GHz και μπορεί να φτάσει με τεχνολογία TurboBoost μέχρι 2.90 GHz.

Κρυφή μνήμη: Η κρυφή μνήμη, μεγέθους 6MB, αποτελείται από 3 επίπεδα L1,L2,L3

L1 cache: 256KB

L2 cache: 1MB

L3 cache: 6MB

Λειτουργικό σύστημα 64bit Windows 10 Home.

Μνήμη συστήματος 8192MB, τεχνολογία DDR3 RAM(1333MHz).

Το σύστημα δεν υποστηρίζει FMA.

Όλα τα ζητούμενα υλοποιήθηκαν σε MATLAB R2016a.

## ΜΕΡΟΣ Α

(2)

(α)  $A=\text{randn}(n)$   $W=\text{randn}(n,t)$   $H=\text{randn}(n,t)$

n	t	$\text{rank}(WH')$	$\text{rank}((A + WH')^{-1} - A^{-1})$
128	1	1	1
128	2	2	2
128	10	10	10
512	1	1	1
512	2	2	2
512	10	10	10

(β)  $A=\text{tril}(\text{randn}(n))$   $W=\text{randn}(n,t)$   $H=\text{randn}(n,t)$

n	t	$\text{rank}(WH')$	$\text{rank}((A + WH')^{-1} - A^{-1})$
128	1	1	3
128	2	2	3
128	10	10	7
512	1	1	11
512	2	2	11
512	10	10	12

```

>> A=randi(128);
>> W=randn(128,1);
>> H=randn(128,1);
>> rank(W*H')

ans =

     1

>> rank(inv(A+W*H')-inv(A))
Warning: Matrix is close to singular or badly scaled. Results may be inaccurate.
RCOND = 2.041209e-19.
Warning: Matrix is close to singular or badly scaled. Results may be inaccurate.
RCOND = 3.624548e-20.

ans =

     3

>> W=randn(128,2);
>> H=randn(128,2);
>> rank(W*H')

ans =

     2

>> rank(inv(A+W*H')-inv(A))
Warning: Matrix is close to singular or badly scaled. Results may be inaccurate.
RCOND = 3.624548e-20.

ans =

     3

>> W=randn(128,10);
>> H=randn(128,10);

```

```

>> A=randi(512);
>> W=randn(512,1);
>> H=randn(512,1);
>> rank(W*H')

ans =

     1

>> rank(inv(A+W*H')-inv(A))
Warning: Matrix is close to singular or badly scaled. Results may be inaccurate.
RCOND = 3.128194e-21.
Warning: Matrix is close to singular or badly scaled. Results may be inaccurate.
RCOND = 2.079449e-21.

ans =

    11

>> W=randn(512,2);
>> H=randn(512,2);
>> rank(W*H')

ans =

     2

>> rank(inv(A+W*H')-inv(A))
Warning: Matrix is close to singular or badly scaled. Results may be inaccurate.
RCOND = 7.572151e-20.
Warning: Matrix is close to singular or badly scaled. Results may be inaccurate.
RCOND = 2.079449e-21.

ans =

    11

>>

```

(v)

Παρατηρώ ότι για  $n=128$  και  $t=1$  (χάριν παραδειγματος) έχω ότι  $\text{cond}(A) = 3.8770e+17$  και  $\text{cond}(A + WH')^{-1} - A^{-1} = 5.9000e+31$ . Ετσι συμπεραίνω ότι η παραπάνω σχέση δεν ισχύει όταν το μητρώο  $A$  είναι σχεδόν ιδιάζων δηλαδή έχει πολύ μεγάλο (κακό) δείκτη κατάστασης. Σε αυτές τις περιπτώσεις το σφάλμα είναι πολύ μεγάλο και ο τύπος αποτυγχάνει. Επομένως για οποιοδήποτε μητρώο με μεγάλο δείκτη κατάστασης ο τύπος δεν ισχύει.

(δ)

Παρατηρώ ότι χρησιμοποιώντας η τάξη του αντιστρόφου του  $gfpp(10)$  είναι 10 (ανάλογη του μεγέθους του μητρώου). Με τη χρήση του τύπου έχω ότι

$rank((gfpp(10) + WH')^{-1} - gfpp(10)^{-1}) = t$  όπου το  $t$  της επιλογής μας μπορεί να είναι και μονάδα. Έτσι η μέθοδος διορθώνει την αστάθεια της LU καθώς αν είχα μητρώο  $gfpp(100)$  θα επρεπε να διαχειριστώ μητρώο τάξης 100 ενώ με την χρήση της μεθόδου έχω να κάνω με μητρώο τάξης 1.

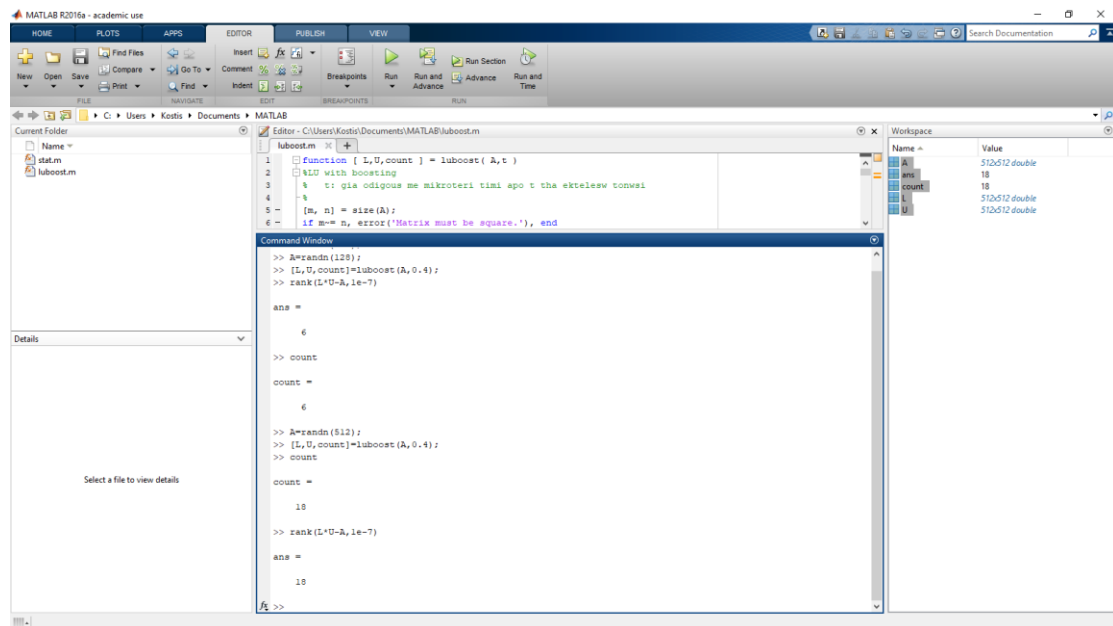
```
>> gfpp(10)
ans =
     1     0     0     0     0     0     0     0     0     1
    -1     1     0     0     0     0     0     0     0     1
    -1    -1     1     0     0     0     0     0     0     1
    -1    -1    -1     1     0     0     0     0     0     1
    -1    -1    -1    -1     1     0     0     0     0     1
    -1    -1    -1    -1    -1     1     0     0     0     1
    -1    -1    -1    -1    -1    -1     1     0     0     1
    -1    -1    -1    -1    -1    -1    -1     1     0     1
    -1    -1    -1    -1    -1    -1    -1    -1     1     1
    -1    -1    -1    -1    -1    -1    -1    -1    -1     1

>> rank(gfpp(10))
ans =
    10

>> rank(inv(gfpp(10)))
ans =
    10

>> W=randn(10,1);
>> H=randn(10,1);
>> rank(inv(gfpp(10)+H*H')-inv(gfpp(10)))
ans =
     1
```

(4) Ικανοποιείται η αριθμητική σχέση  $rank(LU-A)=t$  (όπου  $t$  ο αριθμός των τονώσεων). Χρησιμοποίησα μια κάπως παραλλαγμένη (μόνο για να δείξω τη σχέση που ζητείται) εκδοχή της μεθόδου luboost, εκτός από τα  $L$  και  $U$  η μέθοδος επιστρέφει και ένα μετρητή των τονώσεων για να δείξω ότι η σχέση ικανοποιείται. Στο screenshot που ακολουθεί επέλεξα αυθαίρετα για ελάχιστη τιμή οδηγού την τιμή 0.4 καθώς παρατήρησα ότι δεν προκύπτουν τιμές οδηγών πολύ μικρότερες από αυτή την τιμή. Για το μητρώο  $A=tril(randn(n))$  η σχέση ικανοποιείται αλλά λαμβάνω κάποια προειδοποιητικά μηνύματα όσον αφορά στον δείκτη κατάστασης του μητρώου που χρησιμοποιώ για την τόνωση στην μέθοδο luboost.



(5)

Το μητρώο LU-A θα περιέχει μόνο τις τιμές της τόνωσης στα στοιχεία της διαγωνίου που αντιστοιχούν στις στήλες των οδηγών για τους οποίους χρειάστηκε να εκτελέσω τόνωση. Όλα τα υπόλοιπα στοιχεία του μητρώου θα είναι 0 (ή πάρα πολύ κοντά στο 0).

Έχω ότι  $A=LU$  με  $L = L_1^{-1}L_2^{-1} \dots L_n^{-1}$  και  $U = L^{-1}A$  οπότε  $U = (L_1^{-1}L_2^{-1} \dots L_n^{-1})^{-1}A$ , άρα  $U = L_n \dots L_2L_1A$ , και τελικά...

$$LU - A = (L_1^{-1}L_2^{-1} \dots L_n^{-1})(L_n \dots L_2L_1A) - A$$

(6)

Για την υλοποίηση της μεθόδου geboost(A,B) απλά καλώ την luboost για να δημιουργήσω τους παράγοντες L και U με τόνωση ώστε να μην προκύψει οδηγός μικρότερος του  $1e-02$  και στην συνέχεια λύνω το σύστημα για όλα τα δεύτερα μέλη.

(7)

Κάθε φορά που χρειάζεται τόνωση κάποιο στοιχείο το κόστος δίνεται από τις πράξεις που εκτελούνται μέσα στην συνθήκη if του βρόγχου της luboost. Έχω έναν υπολογισμό αντιστρόφου (τον υπολογίζω μια φορά για κάθε τόνωση και μετά το επαναχρησιμοποιώ όπου χρειάζεται), τον υπολογισμό του παράγοντα a, τον υπολογισμό του μητρώου SMW, τον υπολογισμό του μητρώου T και τον υπολογισμό του A, και το συνολικό κόστος δίνεται από το άθροισμα των παραπάνω πολλαπλασιασμένο με τον αριθμό των τονώσεων. Αν συμβολίσω  $\Omega$  ("υπολογισμού") το κόστος για κάθε υπολογισμό θα έχω ότι το συνολικό κόστος είναι:

$\Omega(\text{αντιστροφή}) = \frac{2}{3}n^3$ ,  $\Omega(\text{υπολογισμός του } a) = n(2n - 1) + 2n - 1 + 1$ ,  $\Omega(\text{υπολογισμός του SMW}) = n(2n - 1) + n(2n - 1) + n^2 + n^2 + n^2$ ,  $\Omega(\text{υπολογισμός του } T) = n^3 + n^2 + n^2$ ,  $\Omega(\text{υπολογισμός του } A) = n^3$ , και συνολικά

$\Omega = t * (\Omega(\text{αντιστροφή}) + \Omega(\text{υπολογισμός του } a) + \Omega(\text{υπολογισμός του SMW}) + \Omega(\text{υπολογισμός του } T) + \Omega(\text{υπολογισμός του } A))$ , όπου  $t$  ο αριθμός των τόνωσεων, άρα

$$\Omega = t * \left( \frac{2}{3}n^3 + 2n^2 - n + 2n - 1 + 1 + 2n^2 - n + 2n^2 - n + 3n^2 + n^3 + 2n^2 + n^3 \right) > \frac{2}{3}n^3 = \Omega(\text{LU}).$$

(Σε όλους τους υπολογισμούς έχω εκτελέσει τις πράξεις με τέτοια σειρά ώστε να ελαχιστοποιείται το κόστος τους.)

(8)

Δεν θα μπορούσαμε να οργανώσουμε όλα τα βήματα της τόνωσης από την αρχή καθώς δεν ξέρουμε σε ποιες θέσεις μπορεί να προκύψουν οδηγοί μικρότεροι της επιτρεπτής τιμής κατά την διάρκεια της άνω τριγωνιοποίησης του  $A$ . Γνωρίζουμε μόνο πια στοιχεία του  $A$  θα χρειαστούν τόνωση στην αρχική μορφή του  $A$  αλλά ένας μικρός οδηγός μπορεί να προκύψει κατά τη διαδικασία.

## ΜΕΡΟΣ Β

(1)

Έχω ονομάσει την μέθοδο `iter_ref` και δέχεται 4 ορίσματα τους πίνακες  $A, W$  και  $H$  και το δεύτερο μέλος  $b$  και επιστρέφει το διάνυσμα της προσέγγισης  $x$ .

Θέλω να βρώ πως σχετίζονται το απόλυτο σφάλμα με το κατάλοιπο, καθώς ο κώδικας πρέπει να τρέχει μέχρι το κατάλοιπο να φράσσεται από συγκεκριμένη τιμή για την οποία το απόλυτο σφάλμα να είναι μικρότερο από  $10^{-6}$ . Παρατηρώ ότι το απόλυτο σφάλμα απορρέει έμμεσα από τον υπολογισμό για το  $z$  στην μέθοδο μου. Πιο συγκεκριμένα,

$$z_k = (A + E)^{-1}r_k = (A + E)^{-1}(b - (A + E)x_k) = (A + E)^{-1}b - (A + E)^{-1}(A + E)x_k = x - x_k$$

και επειδή από τα ζητούμενα της άσκησης θέλω  $\|x - x_k\| < 10^{-6}$  αρκεί να πάρω  $\|z_k\| < 10^{-6}$  αλλά  $z_k = (A + E)^{-1}r_k$  άρα θέλω  $\|(A + E)^{-1}r_k\| < 10^{-6}$  και από τις ιδιότητες των νορμών έχουμε ότι  $\|(A + E)^{-1}r_k\| \leq \|(A + E)^{-1}\| \|r_k\|$  οπότε αρκεί  $\|(A + E)^{-1}\| \|r_k\| < 10^{-6}$  και τελικά παίρνω ότι  $\|r_k\| < \frac{1}{\|(A + E)^{-1}\|} 10^{-6}$ . Η μέθοδος μου θα τρέχει μέχρι το κατάλοιπο να γίνει μικρότερο από αυτήν την τιμή ή αν δεν έχω σύγκλιση μετά από 50(αυθαίρετη τιμή) επαναλήψεις.

(2)

Για να συγκλίνει η μέθοδος πρέπει το μητρώο A να μην έχει πολύ μεγάλο δείκτη κατάστασης. (Για παράδειγμα παρατηρώ ότι για μητρώο vandermonder 10x10 και τυχαίο δεύτερο μέλος b η μέθοδος δεν συγκλίνει μετά από 50 επαναλήψεις.)

(3)

Σε όλα τα ερωτήματα χρησιμοποιήθηκε για δεύτερο μέλος διάνυσμα  $b = \text{randn}(n,1)$  και η μέθοδος τρέχει μέχρι το κατάλοιπο να γίνει μικρότερο από τιμή τέτοια ώστε να έχω απόλυτο σφάλμα μικρότερο του  $10^{-6}$ . Για την μέτρηση του χρόνου χρησιμοποιήθηκε η συνάρτηση `timeit` όπως είχε υποδειχθεί στο μάθημα.

n	h	Επαναλήψεις εκλέπτυνσης	Χρόνος εκτέλεσης( <code>timeit</code> )
10	2	2	3.8641e-04
10	3	2	3.9804e-04
10	4	2	3.9787e-04
100	2	2	0.0119
100	3	2	0.0103
100	4	2	0.0099
1000	2	3	6.4284
1000	3	2	6.0391
1000	4	2	6.1582

Τα παραπάνω αποτελέσματα είναι αναμενόμενα καθώς γνωρίζω από την θεωρία ότι τα μητρώα της μορφής `toeplitz` έχουν  $2n-1$  βαθμούς ελευθερίας αντί για  $n^2$  συνεπώς αναμένουμε καλή συμπεριφορά όσον αφορά στην λύση συστημάτων  $Ax=b$ . Παρατηρώ ότι ο δείκτης κατάστασης του μητρώου  $(A+E)$  κυμαίνεται μεταξύ των τιμών 2.5-3.5 ανεξαρτήτως του  $n$ , έτσι ικανοποιείται η συνθήκη που εκφράστηκε παραπάνω όσον αφορά τον δείκτη κατάστασης και την σύγκλιση της μεθόδου. Τελικά είναι αναμενόμενο, όπως και συμβαίνει η μέθοδος να έχει καλή συμπεριφορά.

**Παρατήρηση:** Όπως είναι αναμενόμενο ο υπολογισμός παίρνει περισσότερο χρόνο για  $n=1000$ .

(4)

(α) Χρησιμοποίησα παραλλαγή του κώδικα του προηγούμενου ερωτήματος, όπου αντι για παραγοντοποίηση LU του μητρώου  $(A+WH')$  εκτελώ την αντιστροφή όπου χρειάζεται με τον τύπο Sherman –Morinson-Woodbury(SMW). Και επειδή τα μητρώα  $W$  και  $H$  είναι τάξης 1, ο τύπος θα είναι  $(A + WH')^{-1} = A^{-1} - \frac{1}{a} A^{-1} WH' A^{-1}$ , για  $a = 1 + H' A^{-1} W \neq 0$

n	h	Επαναλήψεις εκλέπτυνσης με SMW	Χρόνος εκτέλεσης(timeit)
10	2	2	5.1624e-04
10	3	2	5.0016e-04
10	4	2	5.0085e-04
100	2	2	0.0111
100	3	2	0.0125
100	4	2	0.0107
1000	2	3	3.0143
1000	3	2	3.3141
1000	4	2	3.2187

Παρατηρώ ότι με την χρήση του τύπου SMW η αλλαγμένη μέθοδος εκτελεί τον ίδιο αριθμό επαναλήψεων αλλά έχει καλύτερους χρόνους εκτέλεσης(συγκλίνει πιο γρήγορα) από την μέθοδο του προηγούμενου ερωτήματος.

(β)

n	h	Επαναλήψεις εκλέπτυνσης με SMW	Χρόνος εκτέλεσης(timeit)
10	2	Αστοχία(>50 επαναλήψεις)	-
10	3	2	7.5858e-04
10	4	2	6.0310e-04
100	2	Αστοχία(>50 επαναλήψεις)	-
100	3	3	0.0154
100	4	3	0.0163
1000	2	Αστοχία(>50 επαναλήψεις)	-
1000	3	Αστοχία(>50 επαναλήψεις)	-
1000	4	5	4.4083

Παρατηρώ ότι η μέθοδος έχει μικρότερο χρόνο εκτέλεσης τώρα αλλά υπάρχουν περιπτώσεις για τις οποίες δεν συγκλίνει.

## ΜΕΡΟΣ Γ

(2)

Το μητρώο A που προκύπτει με την εκτέλεση των εντολών(η τιμή μου είναι myhash=5161)

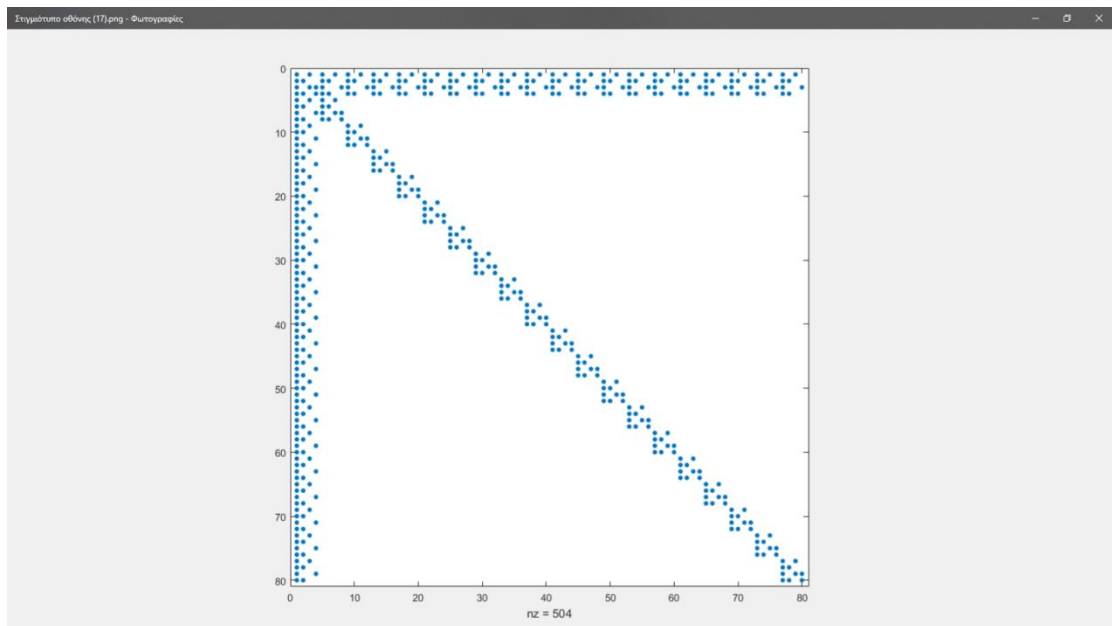
```
rng(5161);,
```

```
T=full(sprand(4,4,0.6);
```

```
A=arrowNW(T,20);
```

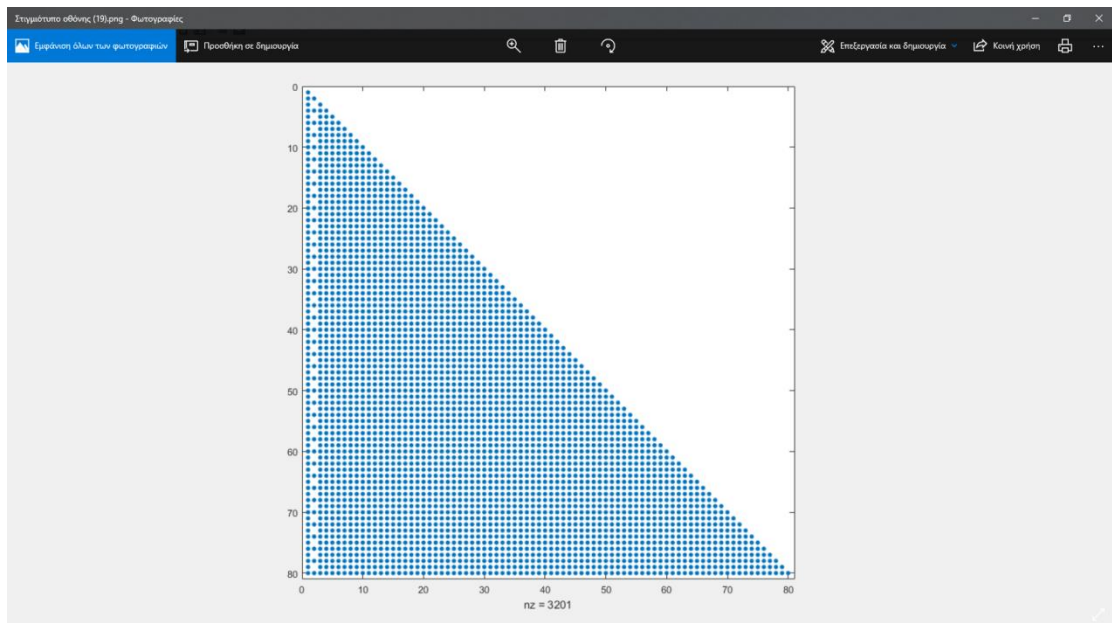
```
spy(A);
```

είναι το ακόλουθο,



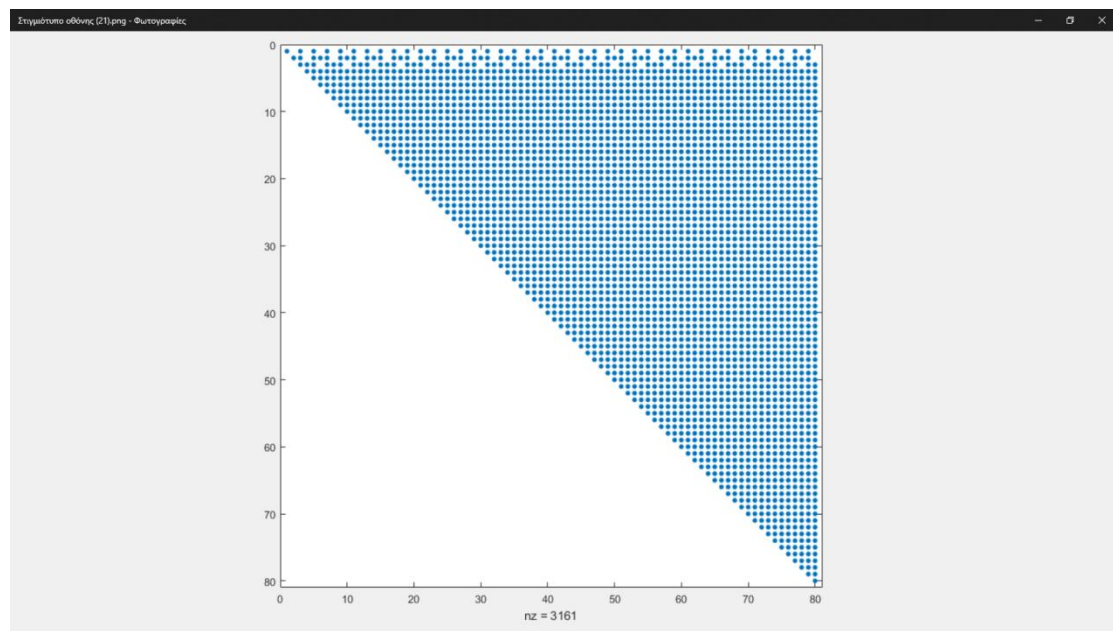
(3)

Εκτελώ την εντολή  $[L,U]=lu(A)$ ; και το κάτω τριγωνικό L είναι:





Και το άνω τριγωνικό U είναι:



(4)

```
>> rng(5161);
>> T=full(sprand(4,4,0.6));
>> A=arrowNW(T,20);
>> spy(A)
>> [L,U]=lu(A);
>> spy(L)
>> spy(U)
>> (nnz(L)+nnz(U)-m*n)/nnz(A)
Undefined function or variable 'm'.

>> (nnz(L)+nnz(U)-4*20)/nnz(A)

ans =

    12.4643

fx >> |
```

Βλέπω ότι το γέμισμα που δημιουργείται είναι 12.4643. Αυτό σημαίνει ότι με την παραγοντοποίηση τα L και U μαζί έχουν 12 φορές περισσότερα μη-μηδενικά στοιχεία από ότι το A. Επομένως δεν μας συμφέρει να εκτελέσουμε την παραγοντοποίηση lu(A) όταν το μητρώο είναι σε αυτήν την μορφή καθώς "χαλάει" η αραιότητα του A.

Πράγμα που φαίνεται και από τις αναπαραστάσεις των L και U. Χονδρικά έχουμε ότι τα L και U έχουν περισσότερα από 3000 μη-μηδενικά στοιχεία το καθένα ενώ το A έχει 500 μη-μηδενικά στοιχεία. Έτσι πολύ πρόχειρα θα μπορούσαμε να δούμε ότι  $(3000+3000)/500=12$ .

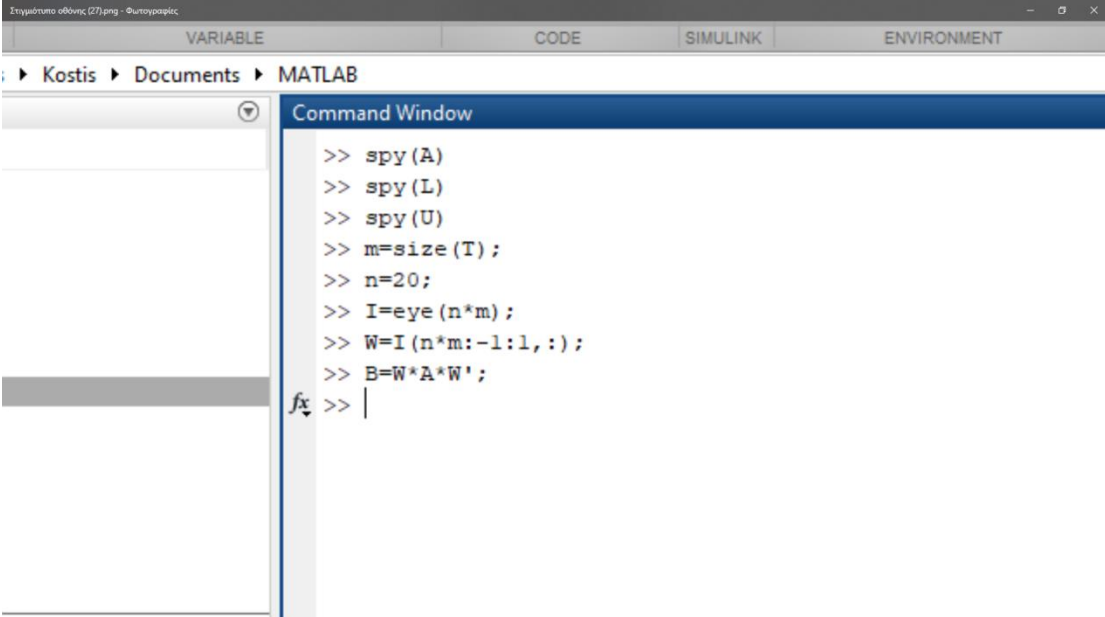
(5)

Θέλω να μετατρέψω το μητρώο από μορφή βέλους(block) με ΒΔ φορά σε μορφή βέλους(block) με ΝΑ φορά που γνωρίζω από το μάθημα ότι διατηρεί την δομή του μητρώου κατά την παραγοντοποίηση LU.

Παρατηρώ ότι οι μεταθέσεις που πρέπει να γίνουν στο A είναι

$A(n*m:-1:1, n*m:-1:1)$  (διαφάνειες μαθήματος). Το μητρώο W θα είναι της μορφής  $W = \begin{pmatrix} 0 & \dots & I \\ \vdots & \ddots & \vdots \\ I & \dots & 0 \end{pmatrix}$  όπου I ταυτοτικοί πίνακες  $m \times m$ .

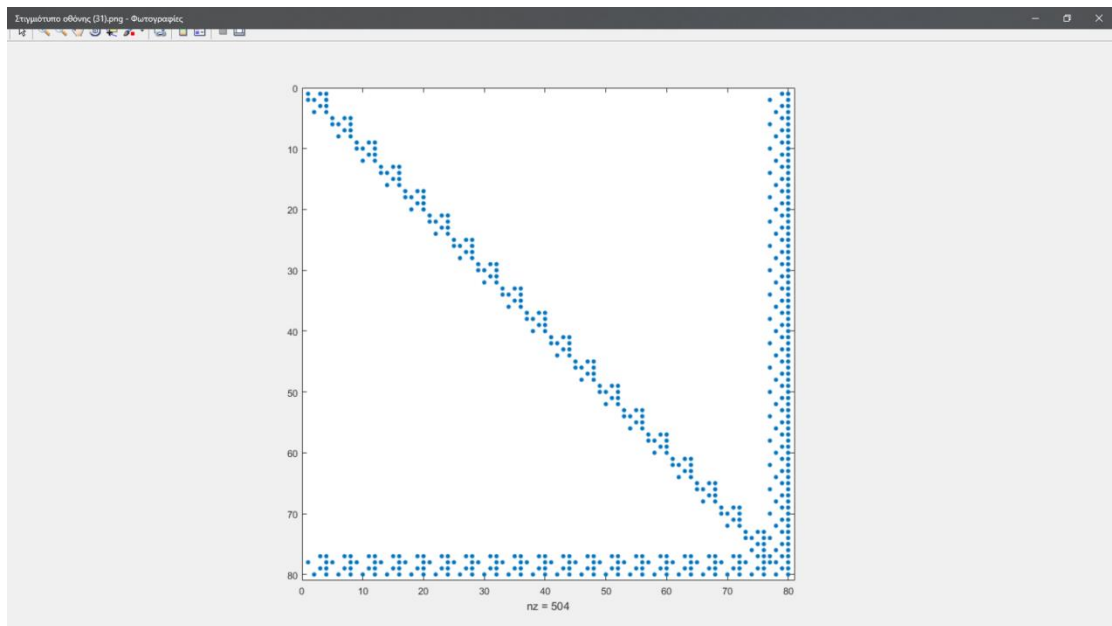
Φτιάχνω το μητρώο W όπως φαίνεται στο screenshot.



The screenshot shows the MATLAB Command Window with the following code:

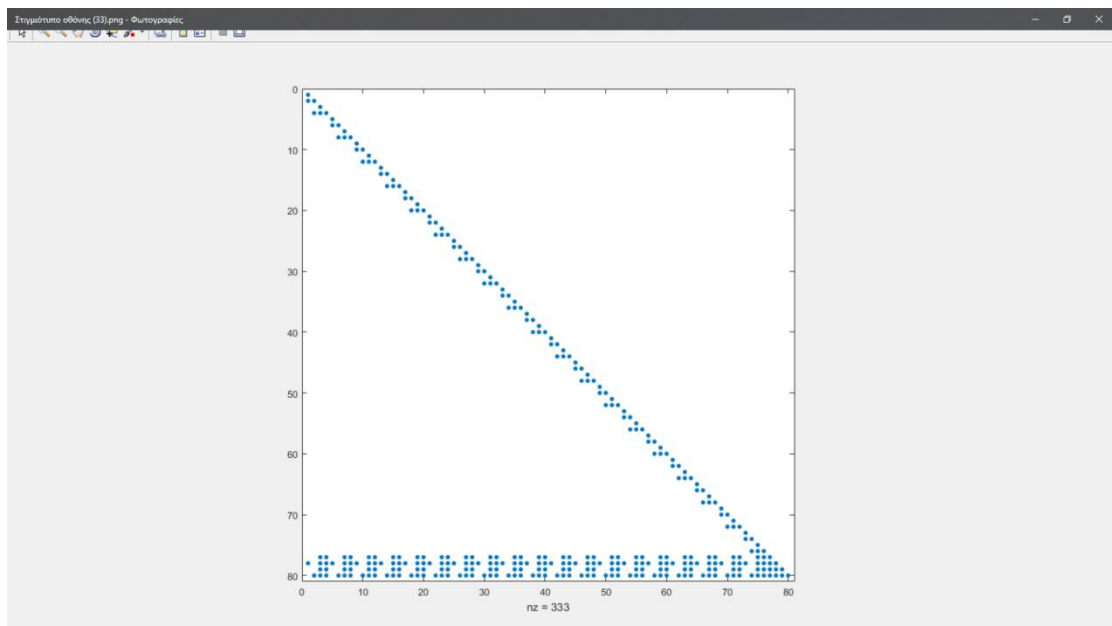
```
>> spy(A)
>> spy(L)
>> spy(U)
>> m=size(T);
>> n=20;
>> I=eye(n*m);
>> W=I(n*m:-1:1,:);
>> B=W*A*W';
fx >> |
```

Το μητρώο B:

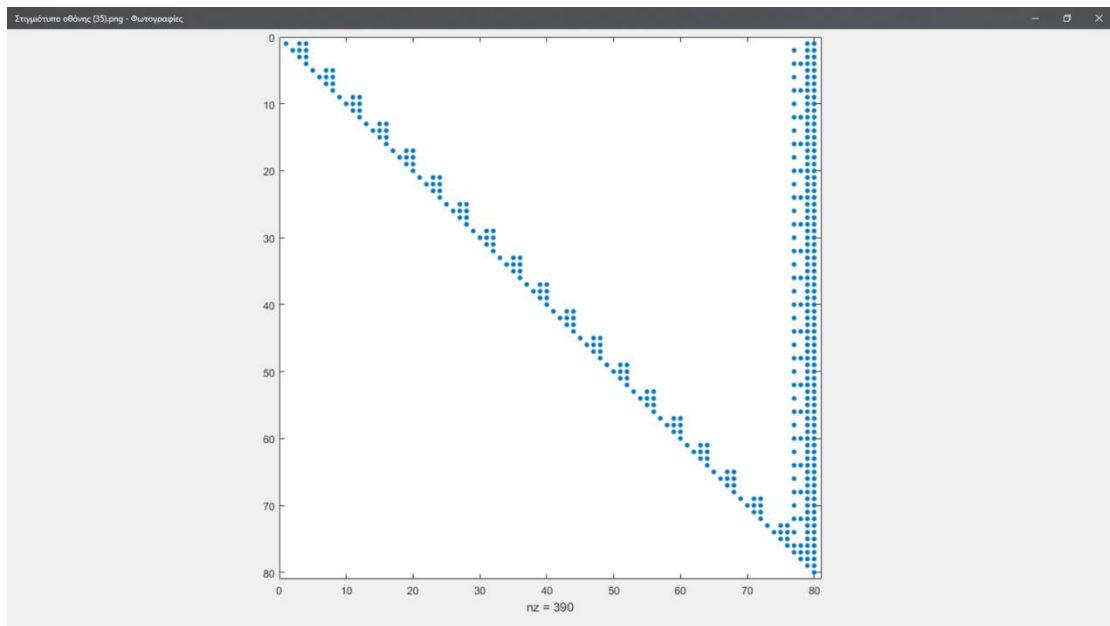


Εκτελώ την παραγοντοποίηση LU του B ( $[L,U]=lu(B)$ ) και οι παράγοντες που προκύπτουν είναι,

Το κάτω τριγωνικό μητρώο L



Το άνω τριγωνικό μητρώο U



Μετράω και πάλι το γέμισμα και έχω

```

>> I=eye(n*m);
>> W=I(n*m:-1:1,:);
>> B=W*A*W';
>> spy(A)
>> spy(B)
>> spy(B)
>> [L,U]=lu(B);
>> spy(L)
>> spy(U)
>> spy(U)
>> (nnz(L)+nnz(U)-4*20)/nnz(B)

ans =

    1.2758

fx >> |

```

Ο παράγοντας γεμίσματος τώρα είναι 1.2758. Παρατηρώ ότι είναι πολύ μικρότερος από την τιμή 12.4643 που ήταν πριν. Βλέπω ότι μετά την μετατροπή του A σε μορφή ΝΔ βέλους(block) πίνακα B οι παράγοντες L και U διατηρούν την ειδική μορφή του μητρώου και έχουν σχεδόν τον ίδιο αριθμό μη-μηδενικών στοιχείων με το B. Επομένως όταν το

μητρώο είναι σε μορφή NA βέλους η παραγοντοποίηση LU διατηρεί την αραιότητα του μητρώου.

(6)

Για τις ανάγκες του ερωτήματος υλοποίησα συνάρτηση με όνομα `arrow_solve` η οποία δέχεται ένα μητρώο  $A$  και ένα δεύτερο μέλος  $b$  και το διάνυσμα  $e$  σαν ορίσματα λύνει το σύστημα  $Ax=b$  με παραγοντοποίηση LU και επιστρέφει την λύση  $x$  και το ζητούμενο σφάλμα.

Για την επίλυση του  $Ax=b$  έχω ( $A$  ΒΔ βέλους ,αποθήκευση σε πυκνή μορφή, πυκνό κάτω τριγωνικό  $L$ ,πυκνό άνω τριγωνικό  $U$ )

m	n	time	$\ e - x\ _{Inf}/\ e\ _{Inf}$
4	250	0.0356	3.7066e-12
8	125	0.0392	8.3373e-12
8	250	0.1950	5.1767e-11
16	125	0.2055	2.3514e-11

Για την επίλυση του  $WAW'x=Wb$  έχω ( $B$  NA βέλους ,αποθήκευση σε πυκνή μορφή, αραιό κάτω τριγωνικό  $L$ ,αραιό άνω τριγωνικό  $U$ )

m	n	time	$\ e - x\ _{Inf}/\ e\ _{Inf}$
4	250	0.0367	5.8176e-13
8	125	0.0385	7.9081e-13
8	250	0.2176	1.5343e-12
16	125	0.2195	1.2528e-12

Παρατηρώ ότι όσον αφορά στους χρόνους έχω παρόμοια συμπεριφορά. Το σφάλμα είναι μια τάξη μεγέθους μικρότερο για την επίλυση του  $WAW'x=Wb$  σε σχέση με την  $Ax=b$ . Αυτό οφείλεται στο ότι το μητρώο  $WAW'$  θα έχει  $N\Delta$  μορφή βέλους και καλύτερη συμπεριφορά όπως εξηγήσαμε παραπάνω.

**Παρατήρηση:** Λαμβάνω χρόνους παρόμοιους σε σύγκριση με την πρώτη περίπτωση, και αυτό γιατί δεν αξιοποιείται η αραιότητα του μητρώου  $B$  (είναι αποθηκευμένο σε πυκνή μορφή) κατά την παραγοντοποίηση, ενώ γνωρίζουμε ότι οι παράγοντες  $L$  και  $U$  θα είναι και αυτοί πολύ αραιά μητρώα. Βλέπω σημαντική βελτίωση παρακάτω από αποθηκεύουμε το  $B$  σε αραιή μορφή(`sparse(B)`).

(7)

Θα χρησιμοποιήσω και πάλι την μέθοδο `arrow_solve` αλλά αυτή την φορά τα μητρώα είναι σε αραιή αναπαράσταση `sparse(A)` και `sparse(B)`.

Για την επίλυση του  $Ax=b$  με  $A=\text{sparse}(A)$  έχω (A ΒΔ βέλους ,αποθήκευση σε αραιή μορφή, πυκνό κάτω τριγωνικό L,πυκνό άνω τριγωνικό U)

m	n	time	$\ e - x\ _{Inf}/\ e\ _{Inf}$
4	250	0.8906	1.8252e-13
8	125	0.9483	8.2045e-13
8	250	7.5667	1.2630e-12
16	125	7.5247	2.8044e-13

Για την επίλυση του  $WAW'x=Wb$  με  $B=\text{sparse}(B)$  έχω (B NA βέλους ,αποθήκευση σε αραιή μορφή, αραιό κάτω τριγωνικό L,αραιό άνω τριγωνικό U)

m	n	time	$\ e - x\ _{Inf}/\ e\ _{Inf}$
4	250	0.0016	5.7954e-13
8	125	0.0025	7.9203e-13
8	250	0.0049	1.5434e-12
16	125	0.0097	1.2443e-12

Για την μεταξύ τους σύγκριση, παρατηρώ ότι όταν μετατρέπω το μητρώο σε NA μορφή βέλους έχω καλύτερους χρόνους σε σχέση με την ΒΔ μορφή. Αυτό οφείλεται στο γεγονός ότι παρόλο που το μητρώο A είναι αποθηκευμένο σε αραιή μορφή από την MATLAB οι παράγοντες L και U θα είναι πυκνά κάτω και άνω τριγωνικά μητρώα αντίστοιχα, γ'αυτό και παρατηρείται τόσο κακή συμπεριφορά από άποψη χρόνου. Σε σύγκριση με πριν βλέπω ότι όταν έχω ΒΔ μορφή βέλους έχω πολύ χειρότερους χρόνους με αναπαράσταση σε αραιή μορφή από ότι αν είχα το A σε πυκνή αναπαράσταση. Αυτό γιατί εκτελώ παραγοντοποίηση LU για αραιό μητρώο και προκύπτουν πυκνοί παράγοντες. Όταν όμως έχω μορφή NA βέλους βλέπω ότι με συμφέρει να αποθηκεύσω το μητρώο σε αραιή μορφή καθώς πετυχαίνω πολύ καλύτερους χρόνους. Η περίπτωση με την καλύτερη συμπεριφορά είναι η τελευταία όπου έχω NA μορφή βέλους και αραιή αναπαράσταση. Αυτό γιατί εκτελώντας την LU για αυτό το μητρώο θα προκύψουν εγγυημένα (όπως είδαμε πιο πάνω) αραιοί παράγοντες L και U.

**Σημείωση:** Αρχικά είχα υλοποιήσει την μέθοδο `arrow_solve` ώστε να λύνει το σύστημα, χωρίς παραγοντοποίηση LU, με την εντολή  $x=A \backslash b$ ; και παρατήρησα καλύτερη συμπεριφορά στην περίπτωση  $Ax=b$  με  $\text{sparse}(A)$  αποθήκευση. Τελικά επέλεξα να λύσω με παραγοντοποίηση LU καθώς γίνονται πιο ξεκάθαρες οι διαφοροποιήσεις όταν το A είναι αραιό αλλά οι παράγοντες L και U όχι.

## ΜΕΡΟΣ Δ

(1)

Για τις ανάγκες του ερωτήματος υλοποίησα 2 συναρτήσεις ,την T\_X που υπολογίζει το πολυώνυμο για την εντολή  $x=qr(A)$  και την T\_QR που υπολογίζει το πολυώνυμο για την εντολή  $[Q,R]=qr(A)$ ; Και οι 2 δέχονται για είσοδο το διάνυσμα  $n$  και επιστρέφουν τους συντελεστές του πολυωνύμου καθώς και το διάνυσμα  $y$  που περιέχει της μετρήσεις αντίστοιχων χρόνων για κάθε  $n$ .

Εκτελώντας τις συναρτήσεις βλέπω ότι για την εντολή  $x=qr(A)$  το διάνυσμα  $p$  που επιστρέφεται είναι το  $p=(0.0193,0.0310,0.196,0.0169)$ . Άρα το αντίστοιχο πολυώνυμο που προκύπτει είναι  $p = 0.0193n^3 + 0.0310n^2 + 0.0196n + 0.0169$ . Το διάνυσμα με τους χρόνους που επιστρέφεται είναι  $y=(0.0007 \ 0.0037 \ 0.0128 \ 0.0213 \ 0.0400 \ 0.0665 \ 0.1596)$ .

Και για την εντολή  $[Q,R]=qr(A)$  το διάνυσμα  $p$  που επιστρέφεται είναι το

$p=(0.0036 \ 0.0340 \ 0.0758 \ 0.0514)$ . Άρα το αντίστοιχο πολυώνυμο που προκύπτει είναι  $p = 0.0036n^3 + 0.0340n^2 + 0.0758n + 0.0514$ . Το διάνυσμα με τους χρόνους που επιστρέφονται είναι  $y=(0.0015 \ 0.0091 \ 0.0215 \ 0.0520 \ 0.0937 \ 0.1543 \ 0.2318)$ .

**Σημείωση:** Αρχικά λάμβανα τις μετρήσεις για το χρόνο όπως φαίνεται στα σχόλια των συναρτήσεων αλλά διαπίστωσα ότι είχα παρόμοια συμπεριφορά(και καλύτερους χρόνους) ακόμη και με μια κλήση της `timeit` για κάθε  $n$ .

(2)

Για την εντολή  $x=qr(A)$ :

από το πολυώνυμο και χρησιμοποιώντας την εντολή `polyval` της MATLAB παίρνω το διάνυσμα με τις αντίστοιχες τιμές για  $n=200:200:1400$  πού είναι

`polyval(p,y)=(0.0170 0.0170 0.0172 0.0174 0.0178 0.0180 ,0.0209)`

και για τις τιμές  $n=250:200:1750$  είναι

`polyval(p,y)=(0.0325 0.0327 0.0330 0.0336 0.0347 0.0361 0.0387 0.0436)`

Για την εντολή  $[Q,R]=qr(A)$ :

Όμοια με πριν από το πολυώνυμο και χρησιμοποιώντας την εντολή `polyval` της MATLAB παίρνω το διάνυσμα με τις αντίστοιχες τιμές για  $n=200:200:1400$  πού είναι

`polyval(p,y)=( 0.0515 0.0521 0.0530 0.0554 0.0588 0.0639 0.0709)`

και για τις τιμές  $n=250:200:1750$  είναι

`polyval(p,y)=( 0.0817 0.0830 0.0845 0.0887 0.0942 0.1025 0.1138 0.1327)`

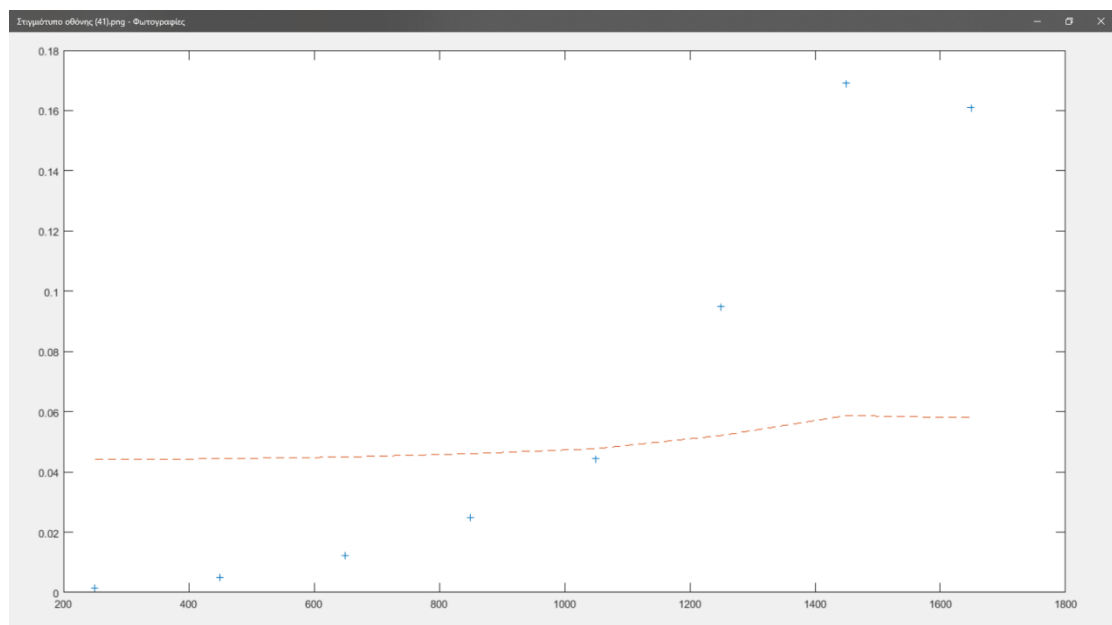
(3)

Για την εντολή  $x=qr(A)$ :

Για την ακρίβεια των προβλέψεων θα παραθέσω το διάνυσμα των απόλυτων διαφορών των μετρήσεων(διάνυσμα  $y$ ) με τις εκτιμήσεις(διάνυσμα  $\text{polyval}(p,y)$ ). Έτσι έχω ότι

$\text{abs}(y-\text{polyval}(p,y)) = (0.0163 \quad 0.0133 \quad 0.0044 \quad 0.0039 \quad 0.0222 \quad 0.0485 \quad 0.1387)$ .

Χρησιμοποίησα την εντολή  $\text{plot}(n,y,'+',n,\text{polyval}(p,y),'--')$  για να φτιάξω την γραφική παράσταση που ζητείται. Με + αναπαρίστανται οι μετρήσεις για το χρόνο και η γραμμή είναι το πολυώνυμο που προέκυψε με την  $\text{polyfit}$ .



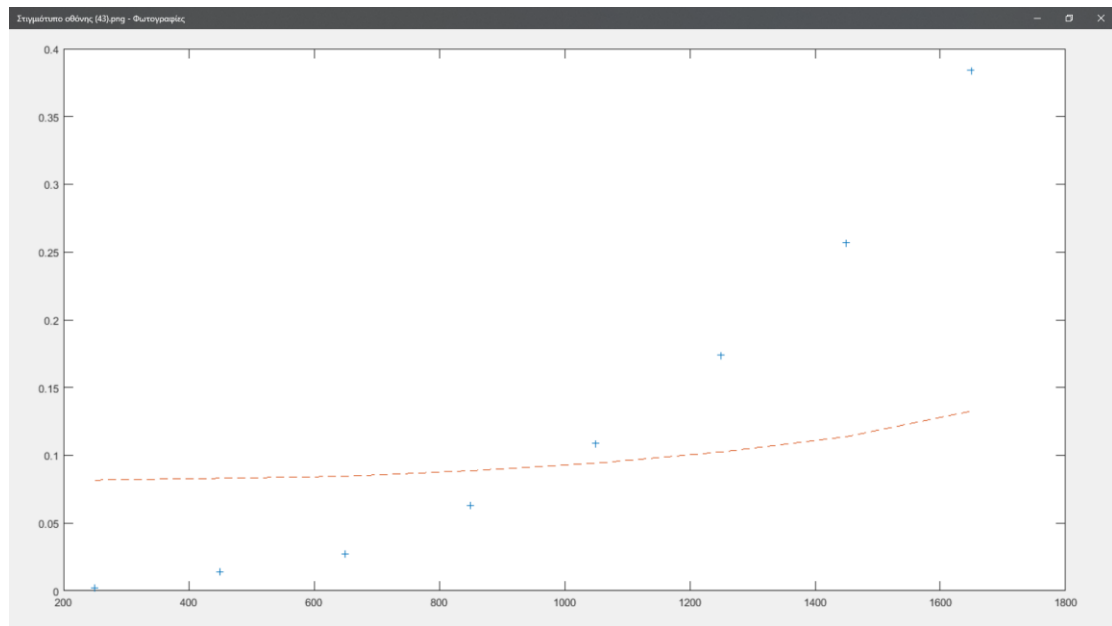
Για την εντολή  $[Q,R]=qr(A)$ :

Όμοια με πριν για την ακρίβεια των προβλέψεων θα παραθέσω το διάνυσμα των απόλυτων διαφορών των μετρήσεων(διάνυσμα  $y$ ) με τις εκτιμήσεις(διάνυσμα  $\text{polyval}(p,y)$ ) που είναι

$\text{abs}(y-\text{polyval}(p,y)) = (0.0500 \quad 0.0430 \quad 0.0315 \quad 0.0034 \quad 0.0349 \quad 0.0904 \quad 0.1610)$

Χρησιμοποίησα και πάλι την εντολή  $\text{plot}(n,y,'+',n,\text{polyval}(p,y),'--')$  για να φτιάξω την γραφική παράσταση που ζητείται. Με + αναπαρίστανται οι μετρήσεις για το χρόνο και η γραμμή είναι το πολυώνυμο που προέκυψε με την  $\text{polyfit}$ .





(4)

Και στις 2 περιπτώσεις αναμένουμε καλύτερη προσέγγιση με πολυώνυμο 4<sup>ου</sup> βαθμού καθώς όσο αυξάνεται ο βαθμός θα αυξάνεται και ακρίβεια με την οποία προσεγγίζουμε τα ζητούμενα σημεία. Για τον ίδιο λόγο θα έχουμε μεγαλύτερη απόκλιση από τα σημεία με πολυώνυμο 2<sup>ου</sup> βαθμού άρα και μεγαλύτερο σφάλμα στις προσεγγίσεις των πολυωνύμων.

Για την εντολή  $x=qr(A)$ :

Για πολυώνυμο 2<sup>ου</sup> βαθμού παίρνω διάνυσμα συντελεστών  $p=(0.0152 \ 0.0351 \ 0.0218)$  δηλαδή έχω την εξίσωση  $p = 0.0152n^2 + 0.0351n + 0.0218$ . Οι χρόνοι είναι  $\gamma=(0.0006 \ 0.0047 \ 0.0097 \ 0.0200 \ 0.0413 \ 0.0666 \ 0.1006)$  και η προσέγγιση του πολυωνύμου 2<sup>ου</sup> βαθμού για αυτούς είναι  $\text{polyval}(p,\gamma)=(0.0218 \ 0.0219 \ 0.0221 \ 0.0225 \ 0.0233 \ 0.0242 \ 0.0255)$ . Το σφάλμα είναι  $\text{abs}(\gamma-\text{polyval}(p,\gamma))=(0.0212 \ 0.0173 \ 0.0124 \ 0.0025 \ 0.0180 \ 0.0424 \ 0.0751)$ . Όπως αναμένεται το σφάλμα είναι μεγαλύτερο σε σχέση με την προσέγγιση με πολυώνυμο 3<sup>ου</sup> βαθμού.

Αντίστοιχα με πολυώνυμο 4<sup>ου</sup> βαθμού παίρνω διάνυσμα συντελεστών  $p=(-0.0588 \ -0.0255 \ 0.1362 \ 0.0886 \ 0.0072)$  δηλαδή έχω την εξίσωση  $p = -0.0588n^4 - 0.0255n^3 + 0.1362n^2 + 0.0886n + 0.0072$ . Οι χρόνοι που λαμβάνω είναι  $\gamma=(0.0008 \ 0.0035 \ 0.0098 \ 0.0200 \ 0.0385 \ 0.1658 \ 0.1006)$  και η προσέγγιση του πολυωνύμου 4<sup>ου</sup> βαθμού για αυτούς είναι  $\text{polyval}(p,\gamma)=(0.0073 \ 0.0075 \ 0.0081 \ 0.0091 \ 0.0108 \ 0.0255 \ 0.0175)$ . Το σφάλμα τώρα είναι  $\text{abs}(\gamma-\text{polyval}(p,\gamma))=(0.0065 \ 0.0040 \ 0.0017 \ 0.0110 \ 0.0276 \ 0.1403 \ 0.0831)$  και βλέπω ότι είναι αρκετά μικρότερο από το αντίστοιχο σφάλμα με πολυώνυμο 3<sup>ου</sup> βαθμού. Δηλαδή όπως αναμέναμε το πολυώνυμο 4<sup>ου</sup> βαθμού προσεγγίζει με μεγαλύτερη ακρίβεια τις μετρήσεις.

Για την εντολή [Q,R]=qr(A):

Για πολυώνυμο 2<sup>ου</sup> βαθμού παίρνω διάνυσμα συντελεστών  $p = (0.0325 \ 0.0874 \ 0.0589)$  δηλαδή έχω την εξίσωση  $p = 0.0325n^2 + 0.0874n + 0.0589$ . Οι χρόνοι είναι  $\gamma = (0.0014 \ 0.0076 \ 0.0212 \ 0.0578 \ 0.1032 \ 0.1800 \ 0.2366)$  και η προσέγγιση του πολυωνύμου 2<sup>ου</sup> βαθμού για αυτούς είναι  $\text{polyval}(p,\gamma) = (0.0591 \ 0.0596 \ 0.0608 \ 0.0641 \ 0.0683 \ 0.0757 \ 0.0814)$ . Το σφάλμα τώρα είναι  $\text{abs}(\gamma - \text{polyval}(p,\gamma)) = (0.0577 \ 0.0520 \ 0.0396 \ 0.0063 \ 0.0349 \ 0.1043 \ 0.1552)$  και παρατηρώ ότι είναι λίγο μεγαλύτερο, όπως και αναμενόταν, από όταν είχα πολυώνυμο 3<sup>ου</sup> βαθμού.

Αντίστοιχα με πολυώνυμο 4<sup>ου</sup> βαθμού παίρνω διάνυσμα συντελεστών  $p = (-0.0062 \ 0.0024 \ 0.0477 \ 0.0784 \ 0.0483)$  δηλαδή έχω την εξίσωση  $p = -0.0062n^4 + 0.0024n^3 + 0.0477n^2 + 0.0784n + 0.0483$ . Οι χρόνοι είναι  $\gamma = (0.0015 \ 0.0106 \ 0.0215 \ 0.0468 \ 0.0971 \ 0.1576 \ 0.2327)$  και η προσέγγιση του πολυωνύμου 4<sup>ου</sup> βαθμού για αυτούς είναι  $\text{polyval}(p,\gamma) = (0.0484 \ 0.0491 \ 0.0500 \ 0.0520 \ 0.0563 \ 0.0618 \ 0.0691)$ . Το σφάλμα τώρα είναι  $\text{abs}(\gamma - \text{polyval}(p,\gamma)) = (0.0469 \ 0.0385 \ 0.0285 \ 0.0052 \ 0.0408 \ 0.0958 \ 0.1636)$  και παρατηρώ ότι είναι μικρότερο από όταν είχα πολυώνυμο 3<sup>ου</sup> ή 2<sup>ου</sup> βαθμού, πράγμα που το περιμέναμε καθώς όσο αυξάνεται ο βαθμός του πολυωνύμου τόσο μεγαλύτερη ακρίβεια θα έχουμε.