

Επιστημονικός Υπολογισμός

ΟΝΟΜΑ: ΡΟΤΧΩΤΑ ΕΙΡΗΝΗ

A.M: 1059654

ΕΤΟΣ: 4^ο

Περιεχόμενα

Εισαγωγικά

Αραιές αναπαραστάσεις πέραν των CSR, CSC

Τανυστές και διαδρομές

Στατιστικά μητρώων

Επαναληπτικές μέθοδοι

Εισαγωγικά

Τα χαρακτηριστικά του υπολογιστικού μου συστήματος για τα πειράματα της εργασίας συνοψίζονται στον παρακάτω πίνακα .

Χαρακτηριστικά	Απάντηση
Έναρξη/λήξη εργασίας	12/01/2021 ώρα 10:00π.μ. 23/2/2021 ώρα 17:00
Υπολογιστικό Σύστημα	Επιτραπέζιος Υπολογιστής(PC)
O/S	Windows 10 Home
processor name	Intel Pentium G3220 ¹
processor speed	3.000 GHZ (base) ²
number of processors	1
total cores	2
total threads	2
L1 D-Cache	32Kbytes(X2)(8-way set associative,64-byte line size) ³
L1 I-Cache	32Kbytes(X2)(8-way set associative,64-byte line size)
L2 D-Cache	256Kbytes(X2)(8-way set associative,64-byte line size)
L3 D-Cache	3Mbytes(12-way set associative,64-byte line size)
Gflops/s	36.72 ⁴
Memory	12GB ⁵
Memory Speed	1333 MHZ ⁶
Matlab Version	9.9.0.1467703 (R2020b) August 26, 2020 ⁷
BLAS	'Intel(R) Math Kernel Library Version 2019.0.3' ⁸ Product Build 20190125 for Intel(R) 64 architecture applications, CNR branch SSE4-2'
LAPACK	'Intel(R) Math Kernel Library Version 2019.0.3' ⁹ Product Build 20190125 for Intel(R) 64 architecture applications, CNR branch SSE4-2, supporting Linear Algebra PACKAge Version 3.7.0

Πίνακας 1: Στοιχεία για τα Πειράματα

¹CPU-Z Processor

²Ο Υπολογιστή μου →Ιδιότητες

³CPU-Z Cache

⁴Intel Pentium G3220 Specs

⁵CPU-Z Memory

⁶Διαχείριση Εργασιών →Επιδόσεις

⁷Εκτελώντας στη Matlab [v d] = version

⁸Εκτελώντας στη Matlab version -blas

⁹Εκτελώντας στη Matlab version -lapack

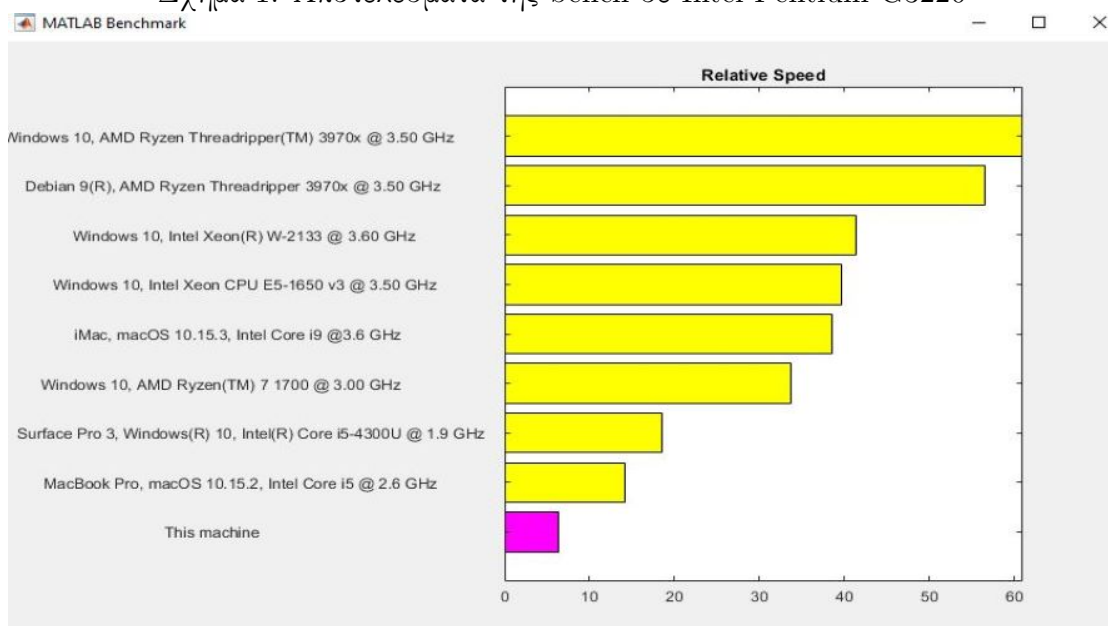
Εκτελώντας την εντολή `bench` στη Matlab παίρνουμε τις παρακάτω εικόνες.

MATLAB Benchmark (times in seconds)

Computer Type	LU	FFT	ODE	Sparse	2-D	3-D
Windows 10, AMD Ryzen Threadripper(TM) 3970x @ 3.50 GHz	0.1930	0.1892	0.3545	0.4085	0.1999	0.2188
Debian 9(R), AMD Ryzen Threadripper 3970x @ 3.50 GHz	0.2612	0.1259	0.3393	0.4216	0.3005	0.2809
Windows 10, Intel Xeon(R) W-2133 @ 3.60 GHz	0.4010	0.3255	0.4494	0.5081	0.3484	0.3166
Windows 10, Intel Xeon CPU E5-1650 v3 @ 3.50 GHz	0.4571	0.3189	0.4957	0.4492	0.3445	0.3922
iMac, macOS 10.15.3, Intel Core i9 @3.6 GHz	0.3286	0.2994	0.3307	0.2971	0.8115	0.5337
Windows 10, AMD Ryzen(TM) 7 1700 @ 3.00 GHz	0.7786	0.5169	0.5180	0.5948	0.3184	0.2160
Surface Pro 3, Windows(R) 10, Intel(R) Core i5-4300U @ 1.9 GHz	1.7749	0.9768	0.7254	0.6882	0.6982	0.6290
MacBook Pro, macOS 10.15.2, Intel Core i5 @ 2.6 GHz	1.5406	0.9646	0.6172	0.6083	2.0698	1.4805
This machine	10.5573	2.1010	1.2166	1.6073	1.1096	1.1074

Place the cursor near a computer name for system and version details. Before using this data to compare different versions of MATLAB, or to download an updated timing data file, see the help for the bench function by typing "help bench" at the MATLAB prompt.

Σχήμα 1: Αποτελέσματα της bench σε Intel Pentium G3220



Σχήμα 2: Αποτελέσματα της bench σε Intel Pentium G3220

Αραιή Αναπαράσταση BCRS

1. Η μελέτη για τη κατασκευή της αραιής αναπαράστασης βασίστηκε στην κατανόηση της CSR.

Για την συνάρτηση *sp_mx2bcrs* έχουμε:

- A: τετραγωνικός αραιό μητρώο
- nb: χωρισμός του A σε nb μπλοκς. Προφανώς για nb=1 εκτελείται η CSR.

Παράδειγμα

```
>> load('A.mat')
>> A

A =

    4.6000    9.3000         0         0         0         0    2.4000    5.6000
    8.6000    8.2000         0         0         0         0    5.3000    1.6000
         0         0         0         0    1.9000    7.9000         0         0
         0         0         0         0    7.1000         0         0         0
         0         0    8.6000    1.7000    2.4000    7.6000         0         0
         0         0    3.9000    2.2000    3.0000    3.3000         0         0
         0         0         0         0    1.8000         0    7.9000    1.2000
         0         0         0         0         0    7.8000    1.0000    5.3000

>> nb=4;
>> [val,col_idx,row_blk]= sp_mx2bcrs(A,nb)

val(:, :, 1) =

    4.6000    9.3000         0         0
    8.6000    8.2000         0         0
         0         0         0         0
         0         0         0         0
```

Σχήμα 3: Εκτέλεση της *sp_mx2bcrs* για το μητρώο A

```

Command Window

val(:, :, 2) =

    0         0    2.4000    5.6000
    0         0    5.3000    1.6000
    1.9000    7.9000         0         0
    7.1000         0         0         0

val(:, :, 3) =

    0         0    8.6000    1.7000
    0         0    3.9000    2.2000
    0         0         0         0
    0         0         0         0

val(:, :, 4) =

    2.4000    7.6000         0         0
    3.0000    3.3000         0         0
    1.8000         0    7.9000    1.2000
         0    7.8000    1.0000    5.3000

col_idx =

    1     5     1     5

row_blk =

    1     3     5

```

Σχήμα 4: Συνέχεια εκτέλεσης της *sp_mx2bcrs*

2. Για το ερώτημα αυτό στόχος είναι να εκμεταλευτούμε την παραπάνω αποθήκευση του αραιού μητρώου για να υπολογίσουμε τη πράξη $y \leftarrow y + A * x$

Για την *spmv_bcrs* έχουμε:

- *y*: κατάλληλο διάνυσμα που χρησιμοποιείται ως είσοδος και έξοδος
- *x*: κατάλληλο διάνυσμα για είσοδο
- *val*: array 3 διαστάσεων που περιέχει το *i*-οστό(κατά γραμμές) *nb x nb* μη μηδενικό μπλοκ
- *col_idx*, *row_blk* : δεδομένα από την κλήση της συνάρτησης *sp_mx2bcrs*

3. Η ορθότητα των παραπάνω συναρτήσεων επιβεβαιώνεται και με τη χρήση μεγαλύτερων μητρώων από τη συλλογή SuiteSparse. Όλα τα μητρώα που έχουν αξιοποιηθεί για την εργασία βρίσκονται στο παραδοτέο αρχείο.

Παράδειγμα

1. Παίρνουμε το μητρώο *Plants_10NN*
2. $x=y=ones(1600,1)$
3. Εκτελούμε *sp_mx2bcrs* για *nb=100*
4. Εκτελούμε *spmv_bcrs*
5. Το αποτέλεσμα αποθηκεύεται στο *y*

Έλεγχος: Για $k=w=ones(1600,1)$ εκτελώ τη πράξη $k=k+A*w$. Στη συνέχεια παίρνουμε τη νόρμα της διαφοράς μεταξύ του *y* και *k* και παρατηρούμε όπως φαίνεται και στην εικόνα παρακάτω ότι η διαφορά τους είναι αμελητέα τάξης του $e-15$ για το μητρώο *Plants_10NN*.

```
Command Window

>> load('Plants_10NN.mat')
>> A=Problem.A;
>> nb=100;
>> [val,col_idx,row_blk]= sp_mx2bcrs(A,nb);
>> y=ones(1600,1);
>> x=y;
>> [y] = spmv_bcrs(y,val,col_idx,row_blk,x);
>> k=ones(1600,1);
>> w=k;
>> k=k+A*w;
>> norm(y-k)

ans =

9.7775e-15
```

Σχήμα 5α: Επαλήθευση της πράξης $y \leftarrow y + A * x$

```

>> load('G23.mat')
>> A=Problem.A;
>> nb=100;
>> [val,col_idx,row_blk]= sp_mx2bcrs(A,nb);
>> y=ones(2000,1);
>> x=y;
>> [y] = spmv_bcrs(y,val,col_idx,row_blk,x);
>> k=ones(2000,1);
>> w=k;
>> k=k+A*w;
>> norm(y-k)

ans =

    0

```

Σχήμα 5β: Επαλήθευση της πράξης $y \leftarrow y + A * x$

Συμπέρασμα:

Για το μητρώο G23 παρατηρούμε ότι η νόρμα της διαφοράς είναι μηδέν. Αυτή η διαφοροποίηση που υπάρχει στα δύο μητρώα σχετίζεται άμεσα με το περιεχόμενο των μητρώων. Αν το μητρώο περιέχει δεκαδικές τιμές, τότε επειδή δεν υπάρχει άπειρη ακρίβεια στο σύστημα μας, παρατηρείται σφάλμα, αφού θα έχουμε απώλεια δεκαδικών ψηφίων. Σε αντίθεση με το μητρώο G23.mat που περιέχει ακέραιες τιμές και το σφάλμα βγαίνει μηδέν. Επειδή και στις δύο περιπτώσεις τα σφάλματα είναι αμελητέα, θεωρούμε ότι επαληθεύεται η ορθή λειτουργία των συναρτήσεων.

Τανυστές και Διαδρομές

Ένας γράφος $G=(V,E)$ με n κορυφές μπορεί να αναπαρασταθεί ως ένας $n \times n$ πίνακας που περιέχει τις τιμές μηδέν και ένα. Αν (i,j) είναι μία ακμή τότε $A[(i,j)]=1$, διαφορετικά $A[(i,j)]=0$.

Γνωρίζουμε από το θεώρημα υπολογισμού διαδρομών ότι το στοιχείο (i,j) του μητρώου A^k (ο πίνακας γειτνίασης υψωμένος στη k δύναμη) δείχνει πόσες διαδρομές μήκους k υπάρχουν από τη κορυφή v_i στη κορυφή v_j .

Πόρισμα του παραπάνω είναι ότι το στοιχείο (i,j) του μητρώου

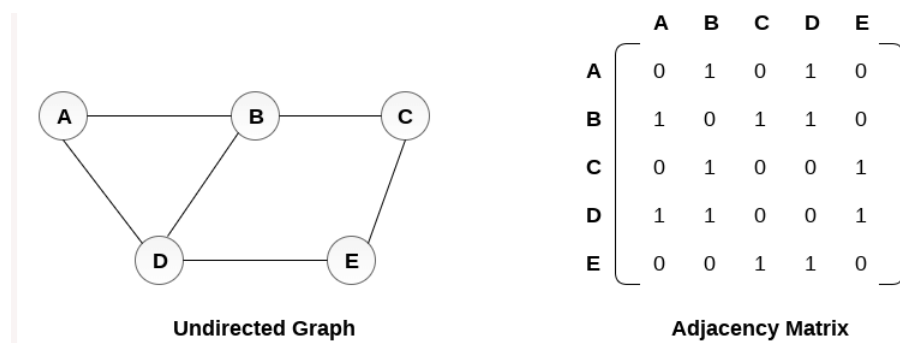
$$A^1 + A^2 + A^3 + \dots + A^k$$

δίνει πόσες διαδρομές μήκους το πολύ k υπάρχουν από τη κορυφή v_i στη κορυφή v_j .

Η συνάρτηση που έχει υλοποιηθεί για τα ζητούμενα της άσκησης είναι η *adjacency.m*

1. Για την κατασκευή του 3-mode τανυστή εκμεταλλευόμαστε τους προηγούμενους $n \times n \times (k-1)$ τανυστές. Για παράδειγμα, ο τανυστής $G(:,:,2)$ προκύπτει από τον $G(:,:,1) * A$. Με αυτό τον τρόπο έχουμε μόνο έναν πολλαπλασιασμό μεταξύ μητρώων (MM), αντί για $(k-1)$ πολλαπλασιασμούς μητρώο με μητρώο.
2. Ουσιαστικά ζητείται η εφαρμογή του πορίσματος για ένα ζευγάρι. Θέλοντας να αξιοποιήσουμε τις συναρτήσεις που δίνονται από το MMT παίρνουμε το ξεδίπλωμα τρόπου-3. Προκύπτει ένα μητρώο με διαστάσεις $k \times (n \times n)$, που σε κάθε στήλη του περιέχει το στοιχείο των αντίστοιχων θέσεων για κάθε διάσταση του G .
3. Αποτελεί γενίκευση του δεύτερου ερωτήματος. Η μόνη διαφορά είναι ότι υπολογίζουμε το sum του μητρώου που προκύπτει από το ξεδίπλωμα τρόπου-3 του τανυστή G . Τώρα, έχουμε όλη την πληροφορία που χρειαζόμαστε για τον υπολογισμό διαδρομών έως k μεταξύ δύο κόμβων.

Για τις ανάγκες της άσκησης χρησιμοποιώ το παρακάτω γράφο.



Σχήμα 6: Γράφος και Πίνακας Γειτνίασης

```

Command Window
>> [G] = adjacency(A,k)
Do you want to search for a specific pair? YES or NO:'YES'
Which row?:1
Which column?:4
The max length 3 for pair (1,4) is 6
G is a tensor of size 5 x 5 x 3
G(:,:,1) =
    0     1     0     1     0
    1     0     1     1     0
    0     1     0     0     1
    1     1     0     0     1
    0     0     1     1     0
G(:,:,2) =
    2     1     1     1     1
    1     3     0     1     2
    1     0     2     2     0
    1     1     2     3     0
    1     2     0     0     2
G(:,:,3) =
    2     4     2     4     2
    4     2     5     6     1
    2     5     0     1     4
    4     6     1     2     5
    2     1     4     5     0

```

Σχήμα 7: Εκτέλεση του ερωτήματος 3.2


```
[G] = adjacency(A,k)
Do you want to search for a specific pair? YES or NO: 'NO'
The max length 3 for pair (1,1) is 4
The max length 3 for pair (1,2) is 6
The max length 3 for pair (1,3) is 3
The max length 3 for pair (1,4) is 6
The max length 3 for pair (1,5) is 3
The max length 3 for pair (2,1) is 6
The max length 3 for pair (2,2) is 5
The max length 3 for pair (2,3) is 6
The max length 3 for pair (2,4) is 8
The max length 3 for pair (2,5) is 3
The max length 3 for pair (3,1) is 3
The max length 3 for pair (3,2) is 6
The max length 3 for pair (3,3) is 2
The max length 3 for pair (3,4) is 3
The max length 3 for pair (3,5) is 5
The max length 3 for pair (4,1) is 6
The max length 3 for pair (4,2) is 8
The max length 3 for pair (4,3) is 3
The max length 3 for pair (4,4) is 5
The max length 3 for pair (4,5) is 6
The max length 3 for pair (5,1) is 3
The max length 3 for pair (5,2) is 3
The max length 3 for pair (5,3) is 5
The max length 3 for pair (5,4) is 6
The max length 3 for pair (5,5) is 2
```

Σχήμα 8: Εκτέλεση του ερωτήματος 3.3

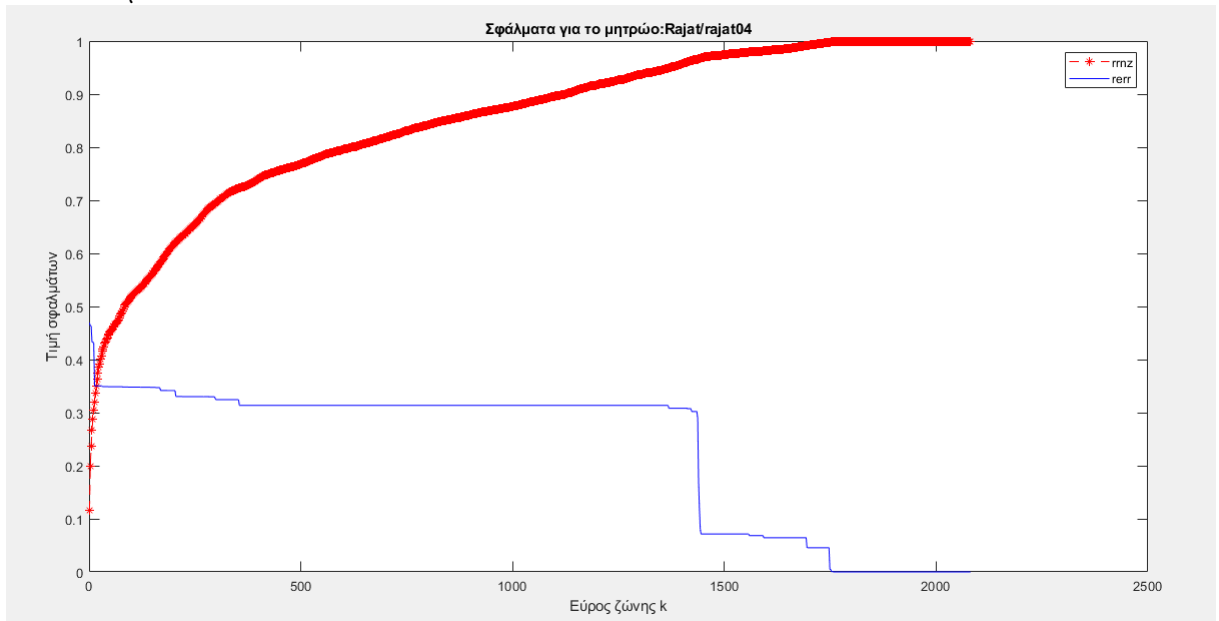
Στατιστικά Αραιών Μητρώων

Ο βαθμός ζωνικότητας ενός μητρώου υπολογίζεται μέσω της συνάρτησης *band_stats*. Πιο συγκεκριμένα:

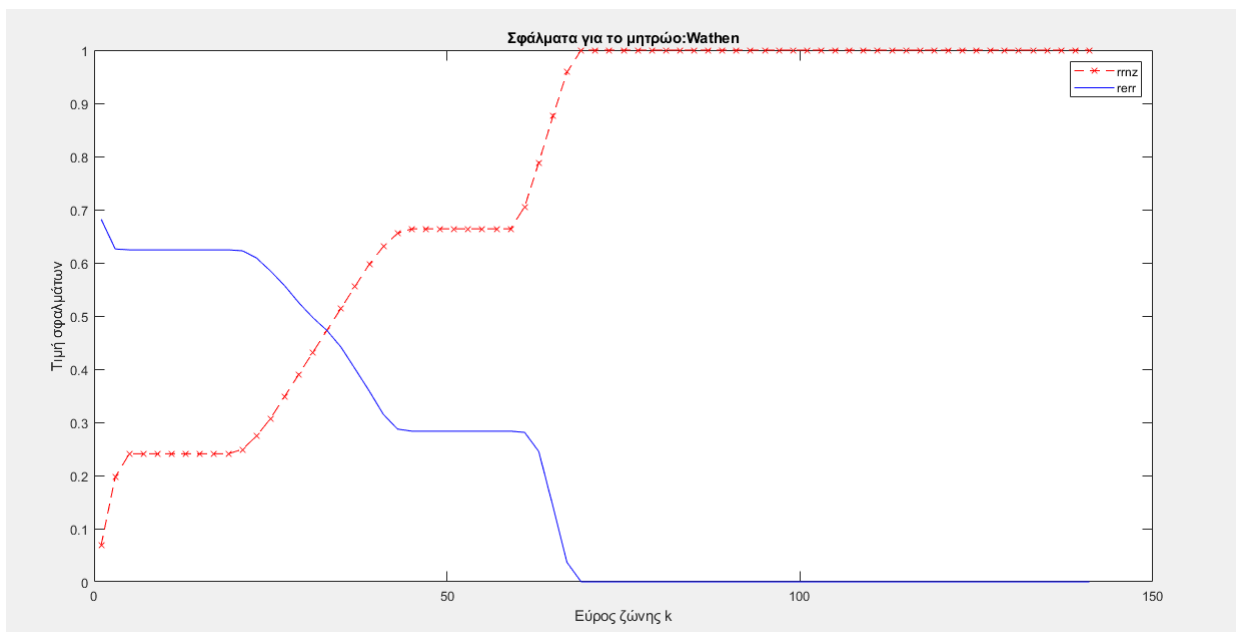
- *mxid*:¹⁰ Η τιμή του *mxid* μπορεί να πάρει τις παρακάτω τιμές:
 1. Μορφή string και επιλέγεται μητρώο με όνομα το string αυτό από τη συλλογή SuiteSparse όπως το *'Rajat/rajat04'*
 2. Το string *'AM'*. Αν επιλεγεί ζητείται από το χρήστη να πληκτρολογήσει τα τελευταία τέσσερα ψηφία του AM του. Στη συνέχεια επιλέξετε το μητρώο με $mxid = 1 + \text{mod}(\text{τελευταία } 4 \text{ ψηφία του AM}, 4892)$ από τη συλλογή SuiteSparse. Γίνεται ειδική προσαρμογή έτσι ώστε το μητρώο που επιστραφεί να είναι τετραγωνικό και μέχρι 1000x1000.
 3. Μητρώο όπως το *gallery=('wathen',10,20)*
- *p*: Εκφράζει το πλήθος των υπό και υπερ-διαγωνίων για το σχηματισμό μητρώων ζώνης. Για $p=0$ επιλέγεται μόνο η κύρια διαγώνιος. Για $p=1$ επιλέγεται η κύρια διαγώνιος, μία υποδιαγώνιος και μία υπερδιαγώνιος. Η μέγιστη τιμή που μπορεί να πάρει το *p* είναι έως $n-1$, όπου n η διάσταση του μητρώου *A*.
- *P*: Η επιστρεφόμενη τιμή της συνάρτησης είναι ένα struct 1x1 με πεδία τα σφάλματα *rnnz* και *rerr*. Αυτά εκφράζουν για κάθε μητρώο το πηλίκο των αριθμό μη μηδενικών στοιχείων του μητρώου ζώνης προς το συνολικό αριθμό μη μηδενικών στοιχείων του αρχικού μητρώου καθώς και τη νόρμα Frobenius της διαφοράς του αρχικού μητρώου και μητρώου ζώνης προς τη νόρμα Frobenius του αρχικού μητρώου ζώνης αντίστοιχα.

¹⁰Για τη λήψη των μητρώων απαιτείται το *ssget.rar* από εδώ

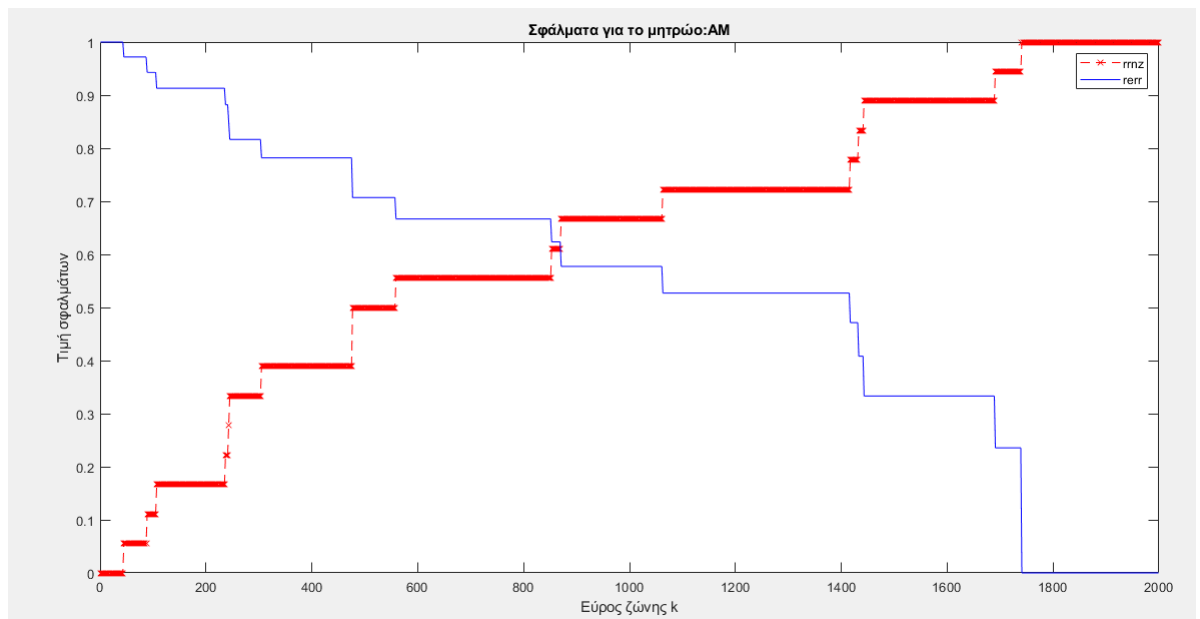
Για τα ζητούμενα τρία μητρώα της άσκησης παίρνουμε τα παρακάτω αποτελέσματα:



Σχήμα 9: Στατιστικά για το μητρώο 'Rajat/rajat04'

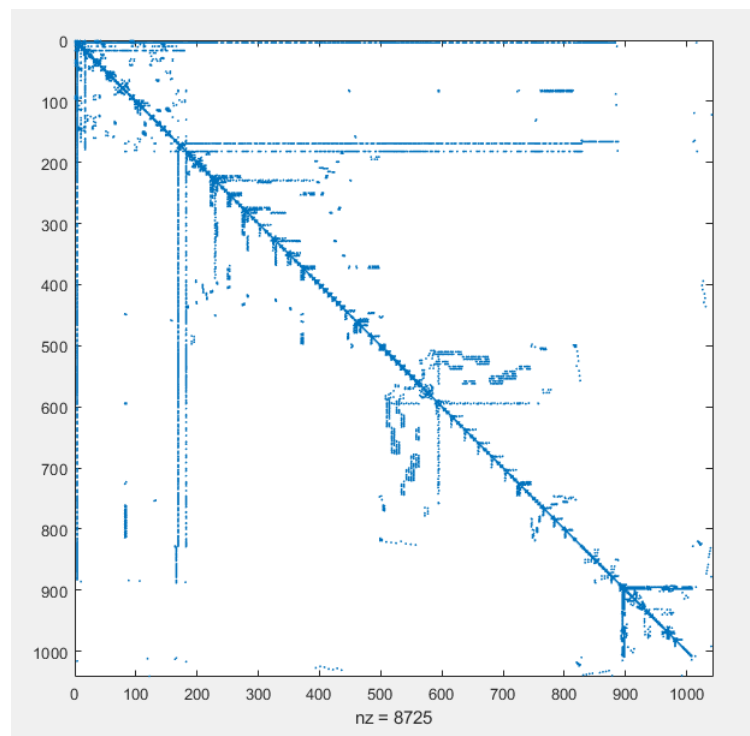


Σχήμα 10: Στατιστικά για το μητρώο *gallery('wathen',10,20)*

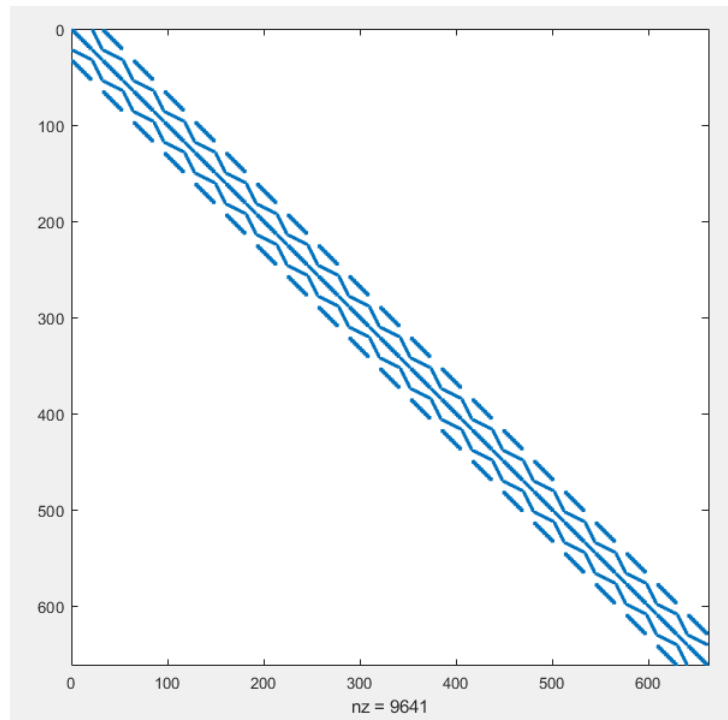


Σχήμα 11: Στατιστικά για το μητρώο με AM:1059654

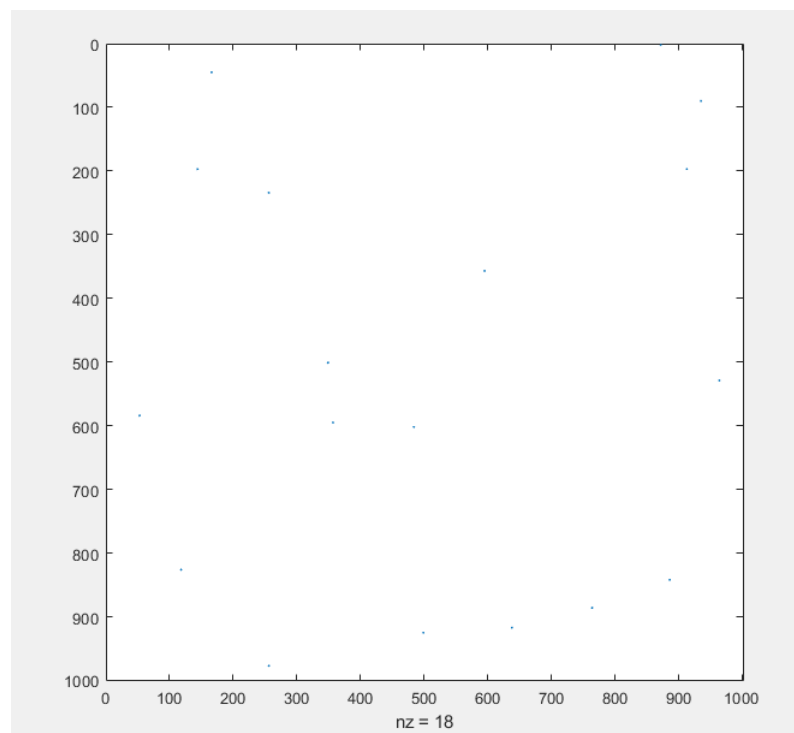
Παρατηρήσεις Εκτελώντας τη εντολή spy για το κάθε μητρώο έχουμε:



Σχήμα 12: Απεικόνιση αραιότητας μητρώου 'Rajat/rajat04'



Σχήμα 13: Απεικόνιση αραιότητας μητρώου $gallery('wathen',10,20)$



Σχήμα 14: Απεικόνιση αραιότητας μητρώου AM

Συμπέρασμα:

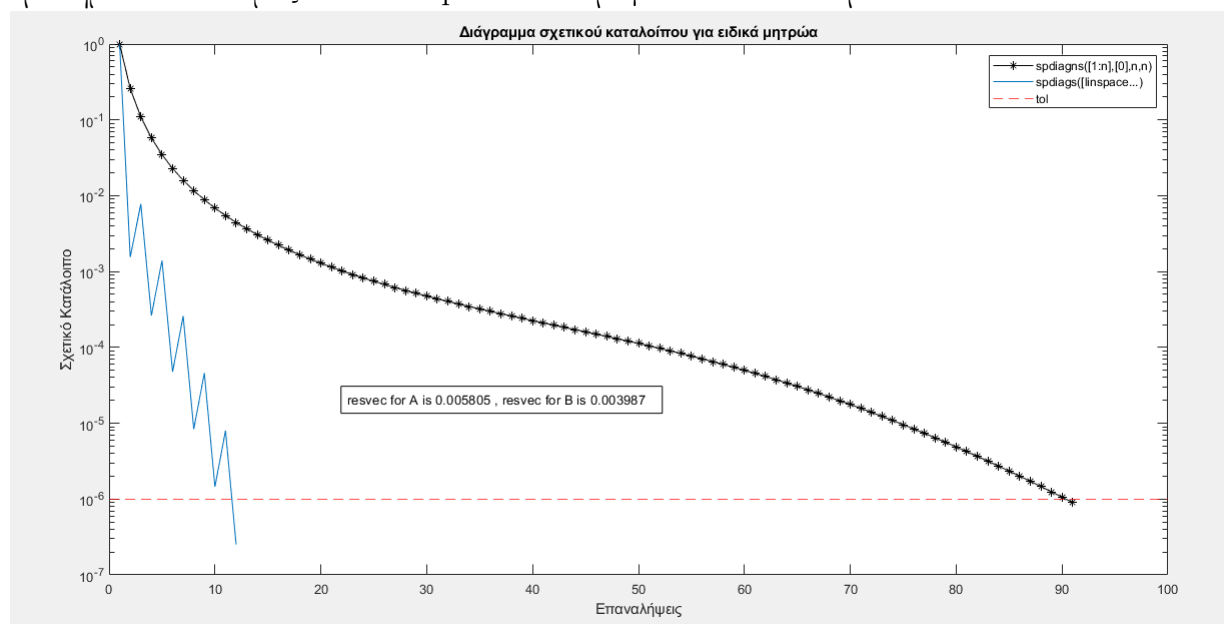
Η σωστή επιλογή του p και του k αντίστοιχα πρέπει να λαμβάνει υπόψη τη δομή του μητρώου. Για παράδειγμα, η δομή του μητρώου Wathen φαίνεται να είναι

μπλοκ τριδιαγώνιο Σχήμα 13. Οπότε δε προσφέρει καμία πληροφορία η τιμή του εύρους ζώνης k να ξεπερνά τη τιμή 70 όπως φαίνεται και στο Σχήμα 10. Για $k=70$ η τιμή του σφάλματος `reft` έχει μηδενιστεί και η τιμή του `rrnz` έχει σταθεροποιηθεί. Αυτό σημαίνει έχουμε σχηματίσει ήδη το αρχικό μητρώο A .

Επαναληπτικές μέθοδοι

Ειδικά Μητρώα

Τα απαιτούμενα της άσκησης εκτελούνται μέσω του αρχείου `ask51.m`. Κάνουμε τη σύμβαση το μητρώο στο ερώτημα 5.1.1 ονομάζεται A και το μητρώο στο ερώτημα 5.1.2 ονομάζεται B . Το plot που παράγεται είναι το παρακάτω:









Σχήμα 15: Διάγραμμα καταλοίπου για ειδικά μητρώα

Απαιτούμενες μετρήσεις:







- **Συνολικό πλήθος πολλαπλασιασμών MV:** Αξιοποιήθηκε η εντολή `profile` της Matlab, με την οποία μπορούμε να μετρήσουμε τον αριθμό επαναλήψεων για τις πιο κοστοβόρες εντολές μέσα σε μια συνάρτηση

```
profile on;
μέθοδος που μας ενδιαφέρει;
profile viewer;
```

- **Ταχύτητα σύγκλισης:** Εντολές χρονόμετρησης `tic`; `toc`;
- **Ακρίβεια αποτελεσμάτων:** Υπολογίζεται η νόρμα της διαφοράς της τελευταίας προσέγγισης της μεθόδου με το `xsol`. Όσο μικρότερη τιμή, τόσο πιο ακριβές το αποτέλεσμα.
- **Δείκτης κατάστασης:** Απααραίτητη μέτρηση για τη σύγκριση των δύων μητρώων.

Line Number	Code	Calls	Total Time	% Time	Time Plot
224	<code>q = iterapp('mtimes',afun,atyp...</code>	90	0.006 s	21.0%	
64	<code>[atype,afun,afcnstr] = iterchk...</code>	1	0.005 s	16.0%	
161	<code>r = b - iterapp('mtimes',afun,...</code>	1	0.003 s	9.6%	
283	<code>end ...</code>	89	0.003 s	9.6%	
67	<code>[m,n] = size(A);</code>	1	0.002 s	5.6%	
All other lines			0.012 s	38.3%	
Totals			0.031 s	100%	

Σχήμα 16: Πλήθος πολλαπλασιασμών MV για το A μητρώο. Γραμμή 224 - 90 MVs

Line Number	Code	Calls	Total Time	% Time	Time Plot
64	<code>[atype,afun,afcnstr] = iterchk...</code>	1	0.006 s	16.9%	
224	<code>q = iterapp('mtimes',afun,atyp...</code>	11	0.006 s	16.3%	
283	<code>end ...</code>	10	0.004 s	9.9%	
161	<code>r = b - iterapp('mtimes',afun,...</code>	1	0.003 s	9.1%	
67	<code>[m,n] = size(A);</code>	1	0.002 s	4.8%	
All other lines			0.016 s	42.9%	
Totals			0.036 s	100%	

Σχήμα 17: Πλήθος πολλαπλασιασμών MV για το B μητρώο. Γραμμή 224 - 11 MVs

```

Command Window

>> ask51
Time for A is: 2.835250e-02
Time for B is: 9.281000e-04
Norm A-xsol is: 3.295953e-04
Norm B-ysol is: 2.866130e-03
Cond for A is: 500
Cond for b is: 1001

```

Σχήμα 18: Μετρήσεις για κάθε μητρώο

Περίεργο: Αυτό που φαίνεται περίεργο λαμβάνοντας υπόψιν και τις παραπάνω πληροφορίες είναι ότι ενώ το A έχει μικρότερο δείκτη κατάστασης από το B, απαιτεί πολλές περισσότερες επαναλήψεις για να συγκλίνει. Επίσης όπως φαίνεται και στο Σχήμα 15 το B έχει μία μη μονότονη συμπεριφορά.

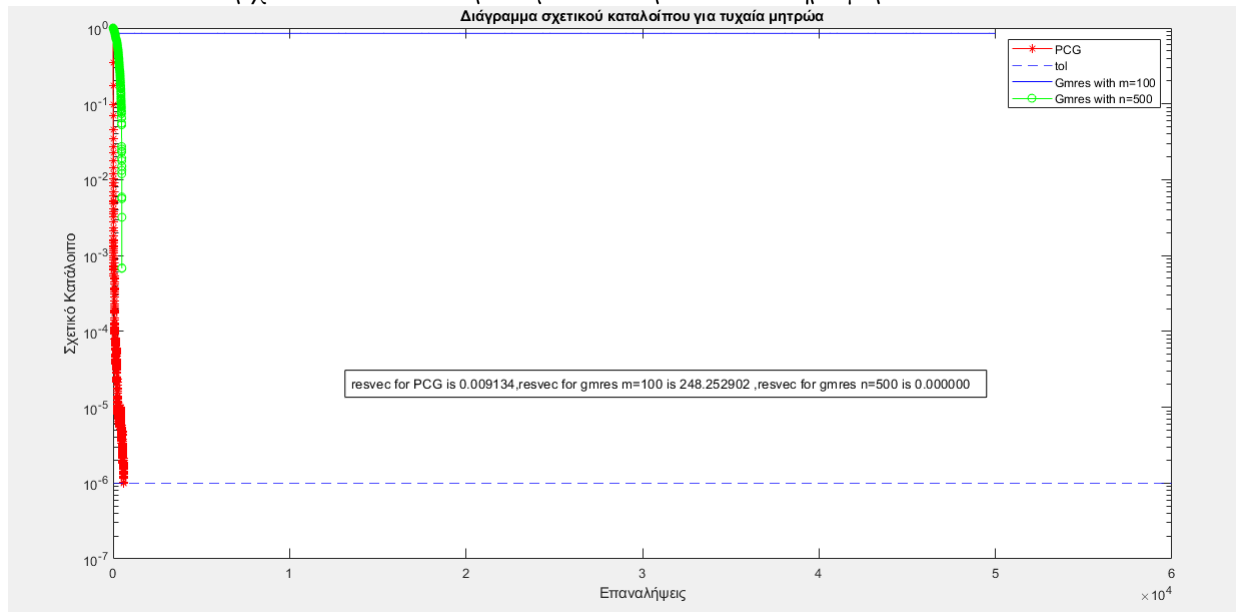
Παρατηρήσεις: Γνωρίζουμε ότι η ταχύτητα σύγκλισης εξαρτάται από την κατανομή των ιδιοτιμών του μητρώου. Ένας άλλος σημαντικός παράγοντας είναι ο δείκτης κατάστασης του μητρώου. Για τη μέθοδο CG, ο ρυθμός σύγκλισης συμπεριφέρεται περίπου όπως η ρίζα του δείκτη κατάστασης.

Μία πιθανή εξήγηση για το περίεργο φαινόμενο που επισημάνθηκε παραπάνω είναι ότι στο μητρώο A, η σύγκλιση αργεί, καθώς η κατανομή των ιδιοτιμών είναι ομοιόμορφα κατανεμημένη, αφού οι διαγώνιοι παίρνουν τις τιμές 1,2,...n. Αντίθετα στο μητρώο B συγκεντρώνονται σε δύο clusters. Τέλος, αριθμός των MVs ισούται με το πλήθος των iterations.







Τυχαία Μητρώα

Ακολουθήθηκε η ίδια λογική με τα ειδικά μητρώα. Επειδή, όμως, η άσκηση αναφέρει να εφαρμόσουμε τη pcg επί των κανονικών εξισώσεων, διαφοροποιείται σε σχέση με τα προηγούμενα ερωτήματα. Πιο συγκεκριμένα αντί να λύνουμε το σύστημα $Ax=b$, επιλύουμε το ισονύαμο σύστημα $A^T * A * b = A^T * b$;

Εκτελώντας το αρχείο ask52.m παίρνουμε τις παρακάτω πληροφορίες:









Σχήμα 19: Διάγραμμα για τυχαία μητρώα







Line Number	Code	Calls	Total Time	% Time	Time Plot
224	<code>q = iterapp('mtimes',afun,atyp...</code>	696	0.080 s	71.0%	
64	<code>[atype,afun,afcnstr] = iterchk...</code>	1	0.005 s	4.0%	
161	<code>r = b - iterapp('mtimes',afun,...</code>	1	0.004 s	3.5%	
283	<code>end ...</code>	695	0.003 s	2.7%	
67	<code>[m,n] = size(A);</code>	1	0.002 s	1.5%	
All other lines			0.019 s	17.2%	
Totals			0.113 s	100%	

Σχήμα 20: Profile για τη μέθοδο pcg

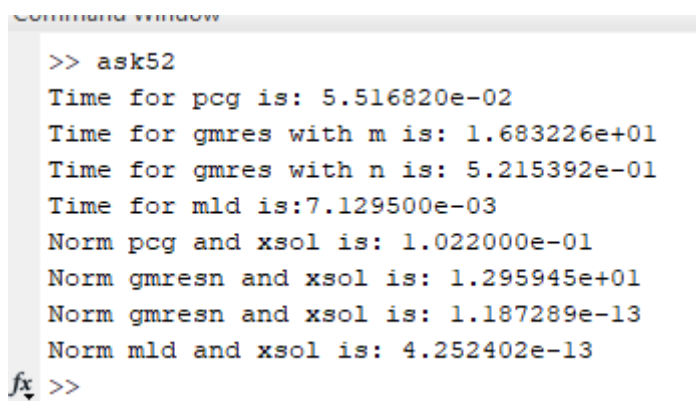
Lines where the most time was spent

Line Number	Code	Calls	Total Time	% Time	Time Plot
318	<code>v = iterapp('mtimes',afun,atyp...</code>	50000	7.012 s	37.9%	
338	<code>v = v - Utemp*(2*(Utemp'*v));</code>	2525000	2.217 s	12.0%	
312	<code>v = v - Utemp*(2*(Utemp'*v));</code>	2475000	2.142 s	11.6%	
337	<code>Utemp = U(:,k);</code>	2525000	1.818 s	9.8%	
311	<code>Utemp = U(:,k);</code>	2475000	1.777 s	9.6%	
All other lines			3.520 s	19.0%	
Totals			18.486 s	100%	

Σχήμα 21: Profile για τη μέθοδο gmres(m)

Line Number	Code	Calls	Total Time	% Time	Time Plot
224	<code>q = iterapp('mtimes',afun,atyp...</code>	696	0.080 s	71.0%	
64	<code>[atype,afun,afcnstr] = iterchk...</code>	1	0.005 s	4.0%	
161	<code>r = b - iterapp('mtimes',afun,...</code>	1	0.004 s	3.5%	
283	<code>end ...</code>	695	0.003 s	2.7%	
67	<code>[m,n] = size(A);</code>	1	0.002 s	1.5%	
All other lines			0.019 s	17.2%	
Totals			0.113 s	100%	

Σχήμα 22: Profile για τη μέθοδο gmres(n)



```
>> ask52
Time for pcg is: 5.516820e-02
Time for gmres with m is: 1.683226e+01
Time for gmres with n is: 5.215392e-01
Time for mld is: 7.129500e-03
Norm pcg and xsol is: 1.022000e-01
Norm gmresn and xsol is: 1.295945e+01
Norm gmresn and xsol is: 1.187289e-13
Norm mld and xsol is: 4.252402e-13
fx >>
```

Σχήμα 23: Εκτέλεση του script ask52.m

Επιβεβαίωση: Επαληθεύεται ότι η επίλυση του συστήματος με την ανάποδη κάθετο επιστρέφει αποτελέσματα πολύ ταχύτατα και με μεγαλύτερη ακρίβεια σε σχέση με την άλλες μεθόδους όπως φάνηκε και στο παραπάνω σχήμα.

Ο χρόνος εκτέλεσης της MLD είναι 0,0012 και η ακρίβεια $1.535e-12$. Αντίθετα αποτελέσματα όσον αφορά την ταχύτητα και την ακρίβεια έχει η gmres για $m=100$.

Το παραπάνω δικαιολογείται, καθώς η mldivide αξιολογεί τη δομή του μητρώου και εκτελεί την καλύτερη μη επαναληπτική μέθοδο. Στη gmres γενικότερα χάνεται η ορθογωνιότητα και δεν αξιοποιείται η δομή του μητρώου. Την επιλέγουμε όταν δε γνωρίζουμε αρκετές πληροφορίες για τη μορφή του μητρώου. Αν από την άλλη, γνωρίζουμε ότι είναι σθο επιλέγουμε την επαναληπτική μέθοδο των συζυγών κλήσεων.