3
Γραμμική Άλγεβρα με το MATLAB

Δημήτριος Χριστόπουλος[1,2]

[1]Εθνικό Καποδιστριακό Πανεπιστήμιο Αθηνών, Τμήμα Οικονομικών Επιστημών
[2]dchristop@econ.uoa.gr

Άνοιξη 2011

Σημειώσεις Εργαστηρίου Γραμμικών Μαθηματικών[3].

[3]Οι ηλεκτρονικές σημειώσεις που ακολουθούν περιέχουν υπερσυνδέσεις, με ένα απλό κλικ, εσωτερικά ή εξωτερικά του κειμένου.

Περιεχόμενα

1 Γραμμική Άλγεβρα με το MATLAB

1.1 Διανύσματα και Πίνακες

Μπορούμε να εισάγουμε ένα διάνυσμα γραμμή με τις εντολές:

```
>> r=[10,20,30,40]

r =

    10    20    30    40

>> r=[10 20 30 40]

r =

    10    20    30    40
```

Επίσης μπορούμε να εισάγουμε ένα διάνυσμα στήλη:

```
>> c=[15;25;35;45]

c =

    15
    25
    35
    45

>> c=[15
25
35
45]
```

Η αναστροφή (**T**ranspose) γίνεται με μία απόστροφο δίπλα στο διάνυσμα ή στον πίνακα:

```
>> r'

ans =

    10
    20
    30
    40
```

```
>> c'

ans =

    15     25     35     45
```

Ισχύουν όλα τα γνωστά από τον πολλαπλασιασμό πινάκων, λ.χ. μπορούμε να κάνουμε την πράξη:

```
>> r*c

ans =

        3500
```

αφού η πράξη $(1 \times 4) \cdot (4 \times 1)$ γίνεται, αλλά δεν μπορούμε να κάνουμε την πράξη:

```
>> r*r

??? Error using ==>mtimes
Inner matrix dimensions must agree.
```

διότι δεν ταιριάζουν οι διαστάσεις των πινάκων ($(1 \times 4) \cdot (1 \times 4)$ δεν μπορεί να γίνει).
Ομοίως μπορούμε να εισάγουμε έναν πίνακα γράφοντας τις γραμμές:

```
>> A=[1,2,3;4,5,6;7,8,9]

A =

     1     2     3
     4     5     6
     7     8     9

>> A=[1 2 3;4 5 6;7 8 9]

A =

     1     2     3
     4     5     6
     7     8     9
```

είτε γράφοντας τις στήλες:

```
>> A=[[1;4;7],[2;5;8],[3;8;9]]

A =
```

```
     1     2     3
     4     5     8
     7     8     9
```

Μάλλον είναι προτιμότερη η γραφή ' κατά γραμμές ' διότι χρειάζονται λιγότερες πληκτρολογήσεις, αλλά καλό είναι να γνωρίζουμε και τον άλλο τρόπο εισαγωγής πίνακα.

Γενικός κανόνας σχηματισμού πινάκων στο MATLAB:
Η χρήση του ερωτηματικού ή του enter στο MATLAB *σημαίνει άλλαξε γραμμή.*
Η χρήση του κόμματος ή του κενού στο MATLAB *σημαίνει άλλαξε στήλη.*

Εάν λοιπόν έχουμε στην επιφάνεια εργασίας μας τους 2×2 πίνακες A και B, τότε για να κατασκευάσουμε τους πίνακες C,D πρέπει να δώσουμε τις αντίστοιχες εντολές:

$$C = \begin{bmatrix} A & B \end{bmatrix}$$

```
>> C=[A,B]
>> C=[A B]
```

$$D = \begin{bmatrix} A \\ B \end{bmatrix}$$

```
>> D=[A;B]
>> D=[A
B]
```

Ένα αριθμητικό παράδειγμα:

```
>> A=[1,2;3,4]

A =

       1              2
       3              4

>> B=[-1,7;4,8]

B =

      -1              7
       4              8

>> C=[A,B]
```

```
C =

       1              2             -1              7
       3              4              4              8

>> C=[A B]

C =

       1              2             -1              7
       3              4              4              8

>> D=[A;B]

D =

       1              2
       3              4
      -1              7
       4              8

>> D=[A
B]

D =

       1              2
       3              4
      -1              7
       4              8
```

Οι στοιχειώδεις έτοιμοι πίνακες του MATLAB βρίσκονται στον Πίνακα 1. Επίσης αν αντί του ορίσματος n βάλουμε το γενικότερο m,n, τότε εμφανίζεται ο αντίστοιχος μη τετραγωνικός πίνακας. Παραδείγματα:

```
>> eye(3)

ans =

     1     0     0
     0     1     0
     0     0     1
```

eye(n)	Μοναδιαίος Πίνακας $n \times n$
ones(n)	Πίνακας $n \times n$ με στοιχεία μονάδες
zeros(n)	Μηδενικός Πίνακας $n \times n$
rand(n)	Τυχαίος Πίνακας $n \times n$ - ομοιόμορφη κατανομή $U(0,1)$
rand(n)	Τυχαίος Πίνακας $n \times n$ - κανονική κατανομή $N(0,1)$

Πίνακας 1: Στοιχειώδεις Έτοιμοι Πίνακες MATLAB,

```
>> eye(3,2)

ans =

     1     0
     0     1
     0     0

>> ones(3)

ans =

     1     1     1
     1     1     1
     1     1     1
>> ones(2,3)

ans =

     1     1     1
     1     1     1

>> zeros(2)

ans =

     0     0
     0     0

>> zeros(2,4)
```

```
ans =

     0     0     0     0
     0     0     0     0
>> rand(5)

ans =

    0.3500    0.3517    0.2858    0.0759    0.1299
    0.1966    0.8308    0.7572    0.0540    0.5688
    0.2511    0.5853    0.7537    0.5308    0.4694
    0.6160    0.5497    0.3804    0.7792    0.0119
    0.4733    0.9172    0.5678    0.9340    0.3371

>> randn(5)

ans =

   -1.7947    0.3035   -0.1941    0.9610   -1.2078
    0.8404   -0.6003   -2.1384    0.1240    2.9080
   -0.8880    0.4900   -0.8396    1.4367    0.8252
    0.1001    0.7394    1.3546   -1.9609    1.3790
   -0.5445    1.7119   -1.0722   -0.1977   -1.0582
```

Δημιουργούμε τον ομοιόμορφα κατανεμημένο τυχαίο πίνακα $1000\,x\,1000$ φροντίζοντας να βάλουμε ένα ερωτηματικό στο τέλος για να μην εμφανιστεί στην οθόνη:

```
>> A=rand(1000);
```

Τώρα με την ακόλουθη εντολή βρίσκουμε την μέση τιμή για τις μέσες τιμές των 1000 στηλών του A, η οποία θεωρητικά πρέπει να είναι $\frac{1}{2} = 0.5$. Πράγματι:

```
>> mean(mean(A))

ans =

   0.500306113473453
```

Εάν κάνουμε το ίδιο γαι έναν κανονικά κατανεμημένο τυχαίο πίνακα 1000×1000 θα πρέπει να βρούμε μέση τιμή περίπου μηδέν, όπως και βρίσκουμε:

```
>> B=randn(1000);
```

```
>> mean(mean(B))

ans =

   0.001775366619321
```

Φυσικά κάθε φορά που εκτελείτε τις άνω εντολές θα έχετε διαφορετικά αποτελέσματα, αφού οι πίνακες είναι σχεδόν τυχαίοι (‘ ψευδο - τυχαίοι ’ για την ακρίβεια). Η ύπαρξη εντολών για δημιουργία τυχαίων πινάκων είναι εξαιρετικά χρήσιμη στον σχεδιασμό στατιστικών πειραμάτων και στον έλεγχο οικονομικών υποδειγμάτων. Η γενικότερη μορφή της εντολής δημιουργίας τυχαίου πίνακα είναι rand(m,n,k,...) ή randn(m,n,k,...), όπου $m, n, k, \ldots$ οι διαστάσεις του πίνακα και η κατάληξη rand**n**(m,n,k,...) σημαίνει κανονική κατανομή των τυχαίων αριθμών. Π.χ. για ένα τυχαίο διάνυσμα στήλη 10×1 γράφουμε:

```
>> rand(10,1)

ans =

    0.1493
    0.2575
    0.8407
    0.2543
    0.8143
    0.2435
    0.9293
    0.3500
    0.1966
    0.2511

>> randn(10,1)

ans =

    0.4882
   -0.1774
   -0.1961
    1.4193
    0.2916
    0.1978
    1.5877
   -0.8045
    0.6966
```

```
    0.8351
```

ανάλογα αν θέλουμε ομοιόμορφη ή κανονική κατανομή.
Μπορούμε να ορίσουμε διανύσματα ή πίνακες με συγκεκριμένο 'βήμα', δηλ. τα στοιχεία τους να ισαπέχουν. Αυτό γίνεται ως εξής:

```
>> v=[1:2:10]

v =

     1     3     5     7     9

>> A=[1:2:10;20:3:32]

A =

     1     3     5     7     9
    20    23    26    29    32
```

Μπορούμε να επιλέξουμε τα πρώτα 3 στοιχεία του διανύσματος v και τα τελευταία τρία στοιχεία της δεύτερης γραμμής του πίνακα A:

```
>> v(1:3)

ans =

     1     3     5

>> A(2,5:-1:3)

ans =

    32    29    26
```

Εδώ το -1 σημαίνει αρνητικό βήμα 1, δηλ. ξεκίνα από το τέλος και εμφάνισε τα στοιχεία ανά 1 μέχρι το 3ο στοιχείο. Υπάρχει και η δυνατότητα χρήσης του βοηθήματος end που υποδεικνύει την τελευταία γραμμή ενός πίνακα. Ας ορίσουμε τον πίνακα A κι ας χρησιμοποιήσουμε το βοήθημα αυτό:

```
>> A=[1,2,3,4,5;11,22,33,44,55;111,222,333,444,555]

A =
```

```
     1      2      3      4      5
    11     22     33     44     55
   111    222    333    444    555

>> A(end)

ans =

   555

>> A(end,:)

ans =

   111    222    333    444    555

>> A(end-1,:)

ans =

    11     22     33     44     55
```

Γενικότερα ο συμβολισμός:

$$A(i,:)$$

σημαίνει *εμφάνισε όλα τα στοιχεία της i-γραμμής*. Ο συμβολισμός:

$$A(:,j)$$

σημαίνει *εμφάνισε όλα τα στοιχεία της j-στήλης*. Όταν αντί για το : βάλουμε μία εντολή $k:l:m$ αυτό σημαίνει εμφάνισε τα στοιχεία από το k με βήμα l έως το m, π.χ. με την εντολή:

```
>> A(2,1:2:end)

ans =

    11     33     55
```

εμφανίσαμε από τη 2η γραμμή του A όλα τα στοιχεία με βήμα 2 έως το τέλος της 2ης γραμμής. Εάν θέλουμε να κρατήσουμε συγκεκριμένες μόνο γραμμές, έστω τις 1,5,8 και στήλες, έστω τις 3,6,9, από ένα πίνακα, δεν έχουμε παρά να το ζητήσουμε με την εντολή A([1 5 8],[3 6 9]).

1.2 Πράξεις με Πίνακες

Εκτός από τις 4 πράξεις της αριθμητικής, οι οποίες γίνονται όταν το επιτρέπουν οι διαστάσεις των πινάκων, π.χ.:

```
>> A=ones(3,2)

A =

     1     1
     1     1
     1     1

>> B=eye(2,4)

B =

     1     0     0     0
     0     1     0     0

>> A*B

ans =

     1     1     0     0
     1     1     0     0
     1     1     0     0

>> B*A
```

??? Error using ==>mtimes
Inner matrix dimensions must agree.
(μήνυμα λάθους διότι δεν ταιριάζουν οι διαστάσεις των πινάκων), στο MATLAB έχουμε και άλλες δυνατότητες.
Έστω λ ένας αριθμός και A ένας πίνακας. Τότε οι παρακάτω πράξεις στο MATLAB:

$$A + \lambda,\ A - \lambda, A * \lambda,\ A/\lambda$$

όλες σημαίνουν να γίνει η αντίστοιχη πράξη σε όλα τα στοιχεία a_{ij} του A. Όμως η πράξη:

$$A^k = A * A * \cdots * A$$

σημαίνει να υψώσουμε τον πίνακα στην k δύναμη. Μπορούμε όμως να χρησιμοποιήσουμε την εντολή:

$$\texttt{A.\^{}}k \rightarrow \begin{pmatrix} a_{1,1}^k & a_{1,2}^k & \dots & a_{1,n}^k \\ a_{2,1}^k & a_{2,2}^k & \dots & a_{2,n}^k \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1}^k & a_{n,2}^k & \dots & a_{n,n}^k \end{pmatrix}$$

με την οποία μπορούμε να κάνουμε την ύψωση σε δύναμη για κάθε στοιχείο του πίνακα, π.χ. για τον πίνακα C ακολούθως έχουμε:

```
>> C=[2:2:12;3:3:18]

C =

     2     4     6     8    10    12
     3     6     9    12    15    18

>> C.^2

ans =

     4    16    36    64   100   144
     9    36    81   144   225   324

>> C.^0.5

ans =

    1.4142    2.0000    2.4495    2.8284    3.1623    3.4641
    1.7321    2.4495    3.0000    3.4641    3.8730    4.2426

>> D=[1:1:5;2:4:20;3:3:15;4:4:20]

D =

     1     2     3     4     5
     2     6    10    14    18
     3     6     9    12    15
     4     8    12    16    20

>> D.^3

ans =
```

```
     1     8    27    64   125
     8   216  1000  2744  5832
    27   216   729  1728  3375
    64   512  1728  4096  8000
```

```
>> D.^(-1)

ans =

    1.0000    0.5000    0.3333    0.2500    0.2000
    0.5000    0.1667    0.1000    0.0714    0.0556
    0.3333    0.1667    0.1111    0.0833    0.0667
    0.2500    0.1250    0.0833    0.0625    0.0500
```

Μπορούμε να υπολογίσουμε την ορίζουσα ενός πίνακα A με την εντολή det(A) , παράδειγμα ο λεγόμενος τετραγωνικός πίνακας Hilbert[1], π.χ. για $n = 10$ έχουμε:

$$H = \begin{pmatrix} 1 & 1/2 & 1/3 & 1/4 & 1/5 & 1/6 & 1/7 & 1/8 & 1/9 & 1/10 \\ 1/2 & 1/3 & 1/4 & 1/5 & 1/6 & 1/7 & 1/8 & 1/9 & 1/10 & 1/11 \\ 1/3 & 1/4 & 1/5 & 1/6 & 1/7 & 1/8 & 1/9 & 1/10 & 1/11 & 1/12 \\ 1/4 & 1/5 & 1/6 & 1/7 & 1/8 & 1/9 & 1/10 & 1/11 & 1/12 & 1/13 \\ 1/5 & 1/6 & 1/7 & 1/8 & 1/9 & 1/10 & 1/11 & 1/12 & 1/13 & 1/14 \\ 1/6 & 1/7 & 1/8 & 1/9 & 1/10 & 1/11 & 1/12 & 1/13 & 1/14 & 1/15 \\ 1/7 & 1/8 & 1/9 & 1/10 & 1/11 & 1/12 & 1/13 & 1/14 & 1/15 & 1/16 \\ 1/8 & 1/9 & 1/10 & 1/11 & 1/12 & 1/13 & 1/14 & 1/15 & 1/16 & 1/17 \\ 1/9 & 1/10 & 1/11 & 1/12 & 1/13 & 1/14 & 1/15 & 1/16 & 1/17 & 1/18 \\ 1/10 & 1/11 & 1/12 & 1/13 & 1/14 & 1/15 & 1/16 & 1/17 & 1/18 & 1/19 \end{pmatrix}$$

```
>> A=hilb(10);
>> det(A)
ans =
       1/46202068362411145000000000000000000000000000000000000
>> format long e
```

[1] Ορίζεται σαν ο $n \times n$ πίνακας H με στοιχεία που δίνονται από τη σχέση $H_{i,j} = \frac{1}{i+j-1}$

```
>> det(A)
ans =
    2.164405264621389e-053
```

Η ακριβής τιμή της ορίζουσας με τη χρήση του wxMaxima είναι:

$$\frac{1}{46206893947914691316295628839036278726983680000000000} \quad = \quad 2.16417922643149191\,10^{-53}$$

Μπορείτε να βρείτε και άλλους πίνακες, παρόμοιους με τον πίνακα Hilbert; Εάν ναι, τότε επικοινωνήστε με τον συγγραφέα[2] των σημειώσεων αυτών.

1.2.1 Αντίστροφος και Ψευδοαντίστροφος ενός Πίνακα

Επίλυση τετραγωνικού συστήματος $n \times n$ με το MATLAB

Μία πολύ σημαντική πράξη στα μαθηματικά είναι η αντιστροφή ενός πίνακα $n \times n$, όταν αυτός μπορεί να αντιστραφεί. Ένα απλό κριτήριο αντιστροφής είναι η ορίζουσα του πίνακα να μην είναι μηδέν. Θα ορίσουμε τον 3×3 τετραγωνικό πίνακα A, θα υπολογίσουμε την ορίζουσά του και τον αντίστροφο αυτού με διάφορους τρόπους και θα κάνουμε επαλήθευση ότι πράγματι είναι αντίστροφος του A:

```
>> format rat
>> A=[1,-1,2;-1,1,0;2,0,-1]

A =

       1            -1             2
      -1             1             0
       2             0            -1

>> det(A)

ans =

      -4

>> Ainv1=A^(-1)

Ainv1 =

       1/4           1/4           1/2
       1/4           5/4           1/2
```

[2] Δημήτριος Θ. Χριστόπουλος, dchristop@econ.uoa.gr

```
       1/2            1/2            0

>> A*Ainv1

ans =

       1              0              0
       0              1              0
       0              0              1

>> Ainv2=inv(A)

Ainv2 =

       1/4            1/4            1/2
       1/4            5/4            1/2
       1/2            1/2            0

>> A*Ainv2

ans =

       1              0              0
       0              1              0
       0              0              1
```

Ένα γραμμικό σύστημα $n \times n$, δηλ. n εξισώσεων με n αγνώστους της μορφής $Ax = b$ ή πιο αναλυτικά:

$$\underbrace{\begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix}}_{A} \cdot \underbrace{\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}}_{x} = \underbrace{\begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}}_{b}$$

έχει μοναδική λύση όταν ο A είναι αντιστρέψιμος και αυτή προκύπτει ως εξής:

$$Ax = b \Rightarrow x = A^{-1}b$$

Αν λοιπόν θεωρήσουμε ένα οποιοδήποτε διάνυσμα b μπορούμε να έχουμε την λύση του συστήματος ως εξής στο MATLAB:

```
>> format compact
```

```
>> b=[12;4;6]
b =
      12
       4
       6
>> sol=A^(-1)*b
sol =
       7
      11
       8
>> sol=inv(A)*b
sol =
       7
      11
       8
>> sol=A\b
sol =
       7
      11
       8
```

Προσέξτε την εμφάνιση του format compact με την οποία αποφεύγουμε να εμφανίζονται κενές γραμμές ανάμεσα στην εκτέλεση διάφορων εντολών. Αν θέλουμε να εμαφανίζονται κενές γραμμές πληκτρολογούμε format loose. Η διαφοροποίηση του MATLAB από τα άλλα προγράμματα στις εντολές επίλυσης ενός τετραγωνικού γραμμικού συστήματος έγκειται απλά στην ύπαρξη της 'αριστερής διαίρεσης ', $A\backslash b$ ανωτέρω.

Επίλυση μη τετραγωνικού συστήματος $m \times n$ με το MATLAB

Ας θεωρήσουμε τώρα το γενικότερο σύστημα m εξισώσεων με n αγνώστους της μορφής $Ax = b$ ή πιο αναλυτικά:

$$\underbrace{\begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{pmatrix}}_{A} \cdot \underbrace{\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}}_{x} = \underbrace{\begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix}}_{b}$$

Αυτό το σύστημα μπορεί να λυθεί με την χρήση του γενικευμένου αντιστρόφου ή ψευδοαντιστρόφου ή αντιστρόφου Moore - Penrose του πίνακα A. Ο ψευδοαντίστροφος ενός πραγματικού πίνακα A ορίζεται σαν εκείνος ο πίνακας $A^{\dagger}$ ο οποίος

ικανοποιεί τις σχέσεις:

$$\begin{aligned} AA^{\dagger}A &= A \\ A^{\dagger}AA^{\dagger} &= A^{\dagger} \\ \left(AA^{\dagger}\right)^{T} &= AA^{\dagger} \\ \left(A^{\dagger}A\right)^{T} &= AA^{\dagger} \end{aligned}$$

Περισσότερα για τον ψευδοαντίστροφο πίνακα μπορείτε να βρείτε εδώ. Η λύση του συστήματος είναι τώρα:

$$Ax = b \Rightarrow x = A^{\dagger}b$$

Η λύση αυτή μπορεί να δειχθεί ότι είναι η λύση με το μικρότερο ευκλείδιο μήκος (ακριβέστερα νόρμα ή στάθμη).

Παράδειγμα 1.1. *Έστω ότι έχουμε να λύσουμε το σύστημα:*

$$\begin{aligned} x - y + 2z &= 12 \\ 2x + 3y - 2z &= 4 \end{aligned}$$

Υπολογίζουμε με την εντολή pinv(A) του MATLAB τον ψευδοαντίστροφο, έπειτα την λύση και στο τέλος κάνουμε επαλήθευση:

```
>> A,b
A =
       1             -1              2
       2              3             -2
b =
      12
       4
>> Pinv=pinv(A)
Pinv =
      27/77          17/77
      -2/77          13/77
      24/77          -2/77
>> sol=Pinv*b
sol =
      56/11
       4/11
      40/11
>> A*sol
ans =
```

```
12
 4
```

Θα μελετήσουμε περισσότερο αναλυτικά τα Γραμμικά Συστήματα στο Κεφάλαιο 5, όπου και θα αναπτύξουμε μόνοι μας αλγορίθμους επίλυσης αυτών των συστημάτων.

1.3 Ανάλυση Πινάκων (Matrix Decomposition)

Πολλές φορές έχουμε να λύσουμε κάποιο πρόβλημα όπου παρουσιάζεται ένας πίνακας ο οποίος δεν είναι τόσο εύκολος στον χειρισμό του. Μπορούμε όμως πάντοτε να βρούμε έναν πιο απλό *όμοιο πίνακα*[3] έτσι ώστε ο αρχικός πίνακας να αναλύεται σε γινόμενο άλλων στοιχειωδών πινάκων. Κατόπιν λύνουμε το πρόβλημά μας για τον απλούστερο όμοιο πίνακα και στο τέλος επιστρέφουμε στο αρχικό πρόβλημα με την βοήθεια του πίνακα P. Επίσης πολλές φορές μας ενδιαφέρει απλά να παραγοντοποιήσουμε τον πίνακά μας σε γινόμενο δύο απλούστερων πινάκων, ώστε να λύσουμε το πρόβλημά μας σε δύο στάδια. Εδώ θα παρουσιάσουμε συνοπτικά τις βασικές αναλύσεις (decompositions) που μπορούμε να κάνουμε σε έναν πίνακα με τη χρήση του MATLAB.

1. **Διαγωνοποίηση** τετραγωνικού πίνακα (ομοιότητα με διαγώνιο πίνακα)

$$AP = PD \Leftrightarrow A = PDP^{-1}, \text{ D διαγώνιος}$$

Ο διαγώνιος πίνακας έχει για στοιχεία τις *ιδιοτιμές(eigenvalues)* του A:

$$D = \begin{pmatrix} \lambda_1 & 0 & 0 & \cdots & 0 \\ 0 & \lambda_2 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_{n-1} & 0 \\ 0 & 0 & \cdots & 0 & \lambda_n \end{pmatrix}$$

ενώ ο διαγωνοποιών πίνακας P έχει, σε αντιστοίχιση προς τις ιδιοτιμές, για στήλες τα αντίστοιχα *ιδιοδιανύσματα(eigenvectors)* του A. Είναι χρησιμότατη ανάλυση για λύση πολλών προβλημάτων, π.χ λύση συστήματος διαφορικών εξισώσεων. Στο MATLAB γίνεται με είτε την εντολή eig(A):

```
>> A=[1 0 0; 0 1 0; -4 8 -1]
A =
       1              0              0
       0              1              0
      -4              8             -1
>> [P,D]=eig(A)
P =
       0           1292/2889         0
       0              0           528/2177
       1          -2584/2889      2112/2177
```

[3] Ο πίνακας A είναι όμοιος με τον B όταν υπάρχει αντιστρέψιμος πίνακας P τέτοιος ώστε $AP = PB \Leftrightarrow A = PBP^{-1}$

```
D =
      -1             0             0
       0             1             0
       0             0             1
>> P*D*inv(P)-A
ans =
       0             0             0
       0             0             0
       0             0             0
```

είτε με την εντολή eig(A,'nobalance'):

```
>> [P,D]=eig(A,'nobalance')
P =
       0          -1/2             0
       0             0          1/4
       1             1             1
D =
      -1             0             0
       0             1             0
       0             0             1
>> P*D*inv(P)
ans =
       1             0             0
       0             1             0
      -4             8            -1
```

Η πρώτη εντολή κάνει κάποια επεξεργασία στα στοιχεία του A και για αυτό όταν υπάρχουν μικροί αριθμοί που προέρχονται από τα σφάλματα στρογγύλευσης τους μεγεθύνει με αποτέλεσμα να λύνει άλλο πρόβλημα τελικά. Προτείνεται η δεύτερη εντολή, η οποία όμως δεν υποστηρίζεται από το Octave.

2. **Τριγωνοποίηση** τετραγωνικού πίνακα (ομοιότητα με άνω τριγωνικό πίνακα)

$$AP = PT \Leftrightarrow A = PTP^{-1}, \text{ T άνω τριγωνικός πίνακας}$$

Στο MATLAB γίνεται την εντολή schur(A):

```
>> A=[1 0 0; 0 1 0; -4 8 -1]
A =
       1             0             0
       0             1             0
      -4             8            -1
```

```
>> [P,T]=schur(A)
P =
       0            -1             0
       0             0             1
       1             0             0
T =
      -1             4             8
       0             1             0
       0             0             1
>> P*T*inv(P)-A
ans =
       0             0             0
       0             0             0
       0             0             0
```

Η τριγωνοποίηση είναι πολλές φορές το μόνο που μπορούμε να κάνουμε με κάποιον τετραγωνικό πίνακα, όταν αυτός δεν έχει απλή δομή[4]. Σε αυτήν την περίπτωση μία μορφή που μπορεί να πάρει ο τριγωνικός πίνακας είναι η λεγόμενη κανονική μορφή *Jordan*, δηλ.:

$$AP = PJ \Leftrightarrow A = PJP^{-1}$$

$$J = \begin{pmatrix} \lambda_1 & 1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_{n-1} & 1 \\ 0 & 0 & \cdots & 0 & \lambda_n \end{pmatrix}$$

Δυστυχώς το βασικό πακέτο του MATLAB δεν διαθέτει εντολή για εύρεση κανονικής μορφής Jordan ενός τετραγωνικού πίνακα, αλλά είναι απαραίτητη η χρήση του SymbolicsToolbox για το σκοπό αυτό.

3. **LU** παραγοντοποίηση οποιουδήποτε πίνακα

   $A = LU$, L κάτω τριγωνικός, U άνω τριγωνικός

   Στο MATLAB γίνεται με την εντολή lu(A):

```
>> A=[1,3,3;-2,-5,1;0,1,7]
A =
       1             3             3
```

[4] Δηλαδή όταν δεν υπάρχουν τόσα ιδιοδιανύσματα όσες και οι ιδιοτιμές μαζί με την πολλαπλότητά τους.

```
      -2            -5             1
       0             1             7
>> [L,U] = lu(A)
L =
      -1/2          1/2            1
       1            0              0
       0            1              0
U =
      -2            -5             1
       0             1             7
       0             0             0
>> L*U-A
ans =
       0             0             0
       0             0             0
       0             0             0
```

Είναι αξιοσημείωτο ότι το MATLAB δεν κατάφερε να κάνει την ανάλυση που θέλαμε, προσέξτε ότι ο L δεν είναι κάτω τριγωνικός. Η σωστή απάντηση με τη χρήση π.χ. του wxMaxima είναι:

$$L = \begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}, U = \begin{pmatrix} 1 & 3 & 3 \\ 0 & 1 & 7 \\ 0 & 0 & 0 \end{pmatrix}$$

4. **LUP** παραγοντοποίηση οποιουδήποτε πίνακα

$$PA = LU \Leftrightarrow A = P^{-1}LU, \text{ U άνω τριγωνικός, P αντιστρέψιμος,}$$

Ο L είναι τώρα κάτω τριγωνικός με διαγώνια στοιχεία ίσα με 1. Στο MATLAB γίνεται με την εντολή lu(A), για τον ίδιο πίνακα με πριν έχουμε:

```
>> [L,U,P] = lu(A)
L =
       1             0             0
       0             1             0
      -1/2           1/2           1
U =
      -2            -5             1
       0             1             7
       0             0             0
P =
       0             1             0
```

```
       0             0             1
       1             0             0
>> L*U-P*A
ans =
       0             0             0
       0             0             0
       0             0             0
```

Η παραγοντοποίηση LU είναι χρήσιμη στην επίλυση γραμμικών συστημάτων σε δύο στάδια:

$$\left\{\begin{matrix} Ax = b \\ A = LU \end{matrix}\right\} \Rightarrow \left\{\begin{matrix} L\left(Ux\right) = b \\ Ux = y \end{matrix}\right\} \Rightarrow \left\{\begin{matrix} Ly = b \\ Ux = y \end{matrix}\right\}$$

Το σύστημα $Ly = b$ λύνεται με *εμπρός αντικατάσταση*, δηλ. βρίσκουμε πρώτα το y_1 από την πρώτη εξίσωση και στο τέλος το y_n:

$$\left\{\begin{matrix} L_{1,1}y_1 & & & & = & b_1 \\ L_{2,1}y_1 & +L_{2,2}y_2 & & & = & b_2 \\ & & \ddots & & & \\ L_{n,1}y_1 & +L_{n,2}y_2 & +\cdots & +L_{n,n}y_n & = & b_n \end{matrix}\right\}$$

ενώ κατόπιν το σύστημα $Ux = y$ λύνεται με *πίσω αντικατάσταση*, δηλ. βρίσκουμε πρώτα το x_n από την τελευταία εξίσωση και στο τέλος το x_1:

$$\left\{\begin{matrix} U_{1,1}x_1 & +U_{1,2}x_2 & +\cdots & +U_{1,n}x_n & = & y_1 \\ & U_{2,2}x_2 & +\cdots & +U_{2,n}x_n & = & y_2 \\ & & \ddots & & & \\ & & & U_{n,n}x_n & = & y_n \end{matrix}\right\}$$

5. **LDL$^{\mathbf{T}}$** παραγοντοποίηση συμμετρικού πίνακα

   $A = LDL^T$, L κάτω τριγωνικός, D διαγώνιος

   Στο MATLAB γίνεται με την εντολή ldl(A):

```
>> A=[1,3,5;3,-2,4;5,4,7]
A =
       1             3             5
       3            -2             4
       5             4             7
```

```
>> [L,D] = ldl(A)
L =
       5/7           -1/30            1
       4/7            1               0
       1              0               0
D =
       7              0               0
       0            -30/7             0
       0              0             -77/30
>> L*D*L'-A
ans =
       *              *               *
       0              0               0
       *              0               0
```

6. **Cholesky** παραγοντοποίηση συμμετρικού θετικά ορισμένου[5] πίνακα

$$A = U^T U, \text{ A συμμετρικός και θετικά ορισμένος,U άνω τριγωνικός}$$

Η παραγοντοποίηση Cholesky είναι χρήσιμη στην επίλυση γραμμικών συστημάτων σε δύο στάδια:

$$\left\{ \begin{array}{c} Ax = b \\ A = U^T U \end{array} \right\} \Rightarrow \left\{ \begin{array}{c} U^T (Ux) = b \\ Ux = y \end{array} \right\} \Rightarrow \left\{ \begin{array}{c} U^T y = b \\ Ux = y \end{array} \right\}$$

Το σύστημα $U^T y = b$ λύνεται με *εμπρός αντικατάσταση* (βρίσκουμε πρώτα το y_1 και στο τέλος το y_n), ενώ κατόπιν το σύστημα $Ux = y$ με *πίσω αντικατάσταση* (βρίσκουμε πρώτα το x_n και στο τέλος το x_1), όπως και στην παραγοντοποίηση LU. Στο MATLAB γίνεται με την εντολή chol(A):

```
>> A=[2 -1 0 0; -1 2 -1 0; 0 -1 2 -1; 0 0 -1 2]
A =
       2             -1              0              0
      -1              2             -1              0
       0             -1              2             -1
       0              0             -1              2
>> U=chol(A)
U =
    1393/985       -985/1393         0              0
       0           1079/881       -881/1079         0
       0              0           1351/1170     -1170/1351
       0              0              0           2889/2584
```

[5] Ένας $n \times n$ πίνακας A λέγεται θετικά ορισμένος όταν ισχύει $x^T Ax > 0, \forall x \in R^n$

```
>> U'*U-A
ans =
       *              0              0              0
       0              *              0              0
       0              0              0              0
       0              0              0              *
```

Προσέξτε την έλλειψη ακρίβειας του MATLAB, το αστεράκι στην επαλήθευση. Η πραγματική τιμή του L με την χρήση του wxMaxima είναι:

$$L = \begin{pmatrix} \sqrt{2} & -\frac{1}{\sqrt{2}} & 0 & 0 \\ 0 & \frac{\sqrt{3}}{\sqrt{2}} & -\frac{\sqrt{2}}{\sqrt{3}} & 0 \\ 0 & 0 & \frac{2}{\sqrt{3}} & -\frac{\sqrt{3}}{2} \\ 0 & 0 & 0 & \frac{\sqrt{5}}{2} \end{pmatrix}$$

7. **QR** παραγοντοποίηση οποιουδήποτε πίνακα

$$A = QR, \text{ Q ορθογώνιος , R άνω τριγωνικός}$$

Ορθογώνιος λέγεται ο πίνακας Q όταν $QQ^T = I$, όπου I ο αντίστοιχος μοναδιαίος $n \times n$ πίνακας. Εάν οι στήλες του A είναι *γραμμικά ανεξάρτητες* , τότε μπορούμε να αποδείξουμε [6] ότι το σύστημα μπορεί να γραφεί στη μορφή:

$$\left\{ \begin{matrix} Rx = Q^T b \\ c = Q^T b \end{matrix} \right\} \Rightarrow Rx = c$$

και κατόπιν να λυθεί με πίσω αντικατάσταση.

Στο MATLAB γίνεται με την εντολή qr(A):

```
>> A=[1,3,3,8;-2,-5,1,-8;0,1,7,8]
A =
       1              3              3              8
      -2             -5              1             -8
       0              1              7              8
>> [Q,R]=qr(A)
Q =
    1292/2889       505/1383       881/1079
   -2584/2889       461/2525       881/2158
       0            461/505       -881/2158
R =
```

[6] Κοιτάξτε λ.χ. εδώ.

```
    2889/1292        2279/392         1292/2889        3499/326
          0           505/461          1756/229        2261/258
          0                 0                 *               *
>> Q*R-A
ans =
          *                 *                 0               *
          0                 *                 *               *
          0                 0                 *               *
```

Προσέξτε ξανά την έλλειψη ακρίβειας του MATLAB, στα αστεράκια της επαλήθευσης. Οι πραγματικές τιμές των Q,R με την χρήση του Mathimatica είναι:

$$Q = \begin{pmatrix} \frac{1}{\sqrt{5}} & \sqrt{\frac{2}{15}} \\ -\frac{2}{\sqrt{5}} & \frac{1}{\sqrt{30}} \\ 0 & \sqrt{\frac{5}{6}} \end{pmatrix}$$

$$R = \begin{pmatrix} \sqrt{5} & \frac{13}{\sqrt{5}} & \frac{1}{\sqrt{5}} & \frac{24}{\sqrt{5}} \\ 0 & \sqrt{\frac{6}{5}} & 7\sqrt{\frac{6}{5}} & 8\sqrt{\frac{6}{5}} \end{pmatrix}$$

Ανακεφαλαιώνοντας παρουσιάζουμε στον Πίνακα 2 τις βασικές αναλύσεις στις οποίες μπορούμε να προβούμε για έναν πίνακα A με το MATLAB.

Ανάλυση	**Μαθηματικά**	**MATLAB**
Διαγωνοποίηση	$A = PDP^{-1}$	[P,D]=eig(A)
Τριγωνοποίηση	$A = PTP^{-1}$	[P,T]=schur(A)
LU	$A = LU$	[L,U]=lu(A)
LUP	$PA = LU$	[L,U,P]=lu(A)
LDL^T	$A = LDL^T$	[L,D]=ldl(A)
Cholesky	$A = U^TU$	U=chol(A)
QR	$A = QR$	[Q,R]=qr(A)

Πίνακας 2: Ανάλυση Πινάκων στο MATLAB

1.4 Ασκήσεις

1. Έστω οι πίνακες:

$$A = \begin{pmatrix} \frac{1}{8} & -\frac{3}{7} & \frac{5}{9} & -\frac{7}{11} \\ \frac{1}{18} & \frac{2}{9} & \frac{5}{8} & \frac{3}{4} \\ \frac{7}{11} & -\frac{3}{17} & \frac{1}{20} & \frac{5}{7} \\ \frac{6}{13} & \frac{2}{7} & -\frac{1}{8} & \frac{1}{4} \end{pmatrix}, B = \begin{pmatrix} -2 & 1 & 11 \\ 3 & 0 & -5 \\ -6 & -11 & 2 \\ 2 & 3 & -7 \end{pmatrix}, v = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix}$$

   (α′) Να κάνετε τις ακόλουθες πράξεις στο MATLAB αφού πρώτα έχετε θέσει format rat:

$$A \cdot B, A^{23}, B^T A, B^T B, Av, v^T v, vv^T, v^T B, v^T Av$$

   (β′) Να επαναλάβετε τις ίδιες πράξεις με το Octave και να συγκρίνετε τα αποτελέσματα με αυτά του MATLAB.

   (γ′) Να κάνετε τώρα με κάποιο πακέτο υπολογιστικής άλγεβρας τις ίδιες πράξεις με απόλυτη ακρίβεια και να υπολογίσετε το απόλυτο και το σχετικό σφάλμα του MATLAB και του Octave. Βρίσκετε κάποια αξιοσημείωτη διαφορά ανάμεσα στα δύο προγράμματα;

2. Δίνονται οι πίνακες :

$$A = \begin{pmatrix} -1 & 2 & 5 & 8 & -9 \\ 7 & 3 & 4 & -2 & 6 \\ 1 & 0 & 3 & 2 & 8 \\ 7 & 4 & -2 & 3 & 5 \\ 1 & 0 & 5 & 0 & 6 \end{pmatrix}, b = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{pmatrix}$$

   (α′) Ξεκινώντας από τον A να εξάγετε με κατάλληλες εντολές του MATLAB τα ακόλουθα:

      i. Τα τρία πρώτα στοιχεία της δεύτερης γραμμής του
      ii. Τα τέσσερα τελευταία στοιχεία της τρίτης στήλης του
      iii. Ολόκληρη την τέταρτη γραμμή αυτού

   (β′) Να δημιουργήσετε τους υποπίνακες:

      i. Τον υποπίνακα που προκύπτει με διαγραφή της δεύτερης γραμμής και δεύτερης στήλης του A
      ii. Τους δύο διαγώνιους υποπίνακες 3×3 και 2×2 αντίστοιχα του A
      iii. Τον επαυξημένο πίνακα $[A|b]$

3. Δίνεται ο πίνακας:

$$A = \begin{pmatrix} \frac{751}{1386} & \frac{625}{1386} & \frac{323}{1386} & -\frac{172}{693} & -\frac{151}{693} \\ \frac{3305}{2772} & \frac{3557}{2772} & \frac{821}{1386} & -\frac{4817}{5544} & -\frac{1751}{5544} \\ -\frac{142}{63} & -\frac{142}{63} & -\frac{263}{252} & \frac{1261}{504} & \frac{295}{504} \\ \frac{5}{42} & \frac{5}{42} & \frac{5}{84} & \frac{95}{168} & -\frac{31}{168} \\ -\frac{1}{3} & -\frac{1}{3} & -\frac{1}{6} & \frac{5}{12} & \frac{5}{12} \end{pmatrix}$$

   (α′) Να διαγωνοποιηθεί ο πίνακας με όλες τις διαθέσιμες εντολές των προγραμμάτων MATLAB και Octave, πάντοτε σε format rat.

   (β′) Να γίνει το ίδιο με κάποιο πακέτο CAS της αρεσκείας σας και να συγκριθούν τα αποτελέσματα με εκείνα του α′ ερωτήματος.

   (γ′) Να τριγωνοποιηθεί ο πίνακας με MATLAB, Octave και κάποιο CAS. Να συγκριθούν τα αποτελέσματα.

   (δ′) Να γίνει η LU παραγοντοποίηση του A με MATLAB, Octave και κάποιο CAS. Να συγκριθούν τα αποτελέσματα.

   (ε′) Να γίνει η QR παραγοντοποίηση του A με MATLAB, Octave και κάποιο CAS. Να συγκριθούν τα αποτελέσματα.

4. Έστω ο πίνακας:

$$A = \begin{pmatrix} 106 & 87 & 38 & 34 & 16 \\ 87 & 102 & 49 & 60 & 24 \\ 38 & 49 & 34 & 48 & 12 \\ 34 & 60 & 48 & 97 & 16 \\ 16 & 24 & 12 & 16 & 16 \end{pmatrix}$$

   (α′) Να γίνει η LDL^T παραγοντοποίηση του A με MATLAB και κάποιο CAS. Να συγκριθούν τα αποτελέσματα.

   (β′) Να γίνει η παραγοντοποίηση Cholesky του A με MATLAB, Octave και κάποιο CAS. Να συγκριθούν τα αποτελέσματα.