ΕΜ-181 – Εισαγωγή στην Αριθμητική Ανάλυση
Η συλλογή προγραμμάτων LAPACK

Η συλλογή προγραμμάτων LAPACK (Linear Algebra PACKage) αποτελείται απο υποπρογράμματα γραμμένα σε γλώσσα Fortran 77. Η συλλογή LAPACK παρέχει ρουτίνες για τη λύση γραμμικών συστημάτων, για το πρόβλημα ελαχίστων τετραγώνων, για προβλήματα ιδιοτιμών και για προβλήματα ιδιάζουσων τιμών.

Η συλλογή LAPACK είναι εγκατεστημένη στους υπολογιστές του εργαστηρίου στην Η-205. Για να χρησιμοποιείσετε κάποια απο τα υποπρογράμματα της συλλογής στο δικό σας πρόγραμμα, δεν έχετε παρά να το συνδέσετε με την βιβλιοθήκη `lapack`:

```
> g77 -o myprog myprog.f -llapack
> ./myprog
```

Η συλλογή LAPACK περιέχει δύο ειδών υποπρογράμματα: τα **υποπρογράμματα–οδηγούς** τα οποία λύνουν προβλήματα όπως τριγωνοποίηση ενός πίνακα και λύση ενός γραμμικού συστήματος, και τα **υπολογιστικά υποπρογράμματα** τα οποία εκτελούν συγκεκριμένα καθήκοντα, όπως ανάλυση LU, οπισθοδρόμηση, κλπ. Και οι δύο αυτές κατηγορίες υποπρογραμμάτων χρησιμοποιούν τα λεγόμενα Βασικά Υποπρογράμματα Γραμμικής Αλγεβρας (Basic Linear Algebra Subroutines, BLAS) για να εκτελέσουν βασικές λειτουργίες, όπως εναλλαγή δυο γραμμών, στάθμιση γραμμής ή στήλης ενός πίνακα, πολλαπλασιασμό πίνακα επι διάνυσμα, κλπ.

Η συλλογή LAPACK παρέχεται δωρεάν, στη διεύθυνση http://www.netlib.org/lapack/. Εκει μπορεί να βρεί κανείς περισσότερες πληροφορίες για την εγκατάσταση της συλλογής σε διάφορες αρχιτεκτονικές υπολογιστών και να μάθει για τη χρήση των υποπρογραμμάτων της. Εμείς εδω θα ασχοληθούμε με τα υποπρογράμματα του LAPACK που έχουν σχέση με τα θέματα της γραμμικής άλγεβρας που καλύπτονται στο μάθημα.

1 Λύση γραμμικών συστημάτων

Η συλλογή LAPACK παρέχει ρουτίνες για τη λύση πραγματικών και μιγαδικών συστημάτων σε απλή και διπλή ακρίβεια. Τα ονόματα των υποπρογραμμάτων της είναι της μορφής `XYYZZZ`, όπου

- το πρώτο γράμμα `X` δηλώνει τον αριθμητικό τύπο των δεδομένων (πινάκων ή διανυσμάτων) τον οποίο χειρίζεται το υποπρόγραμμα:
  - S για δεδομένα τύπου `REAL`
  - D για δεδομένα τύπου `REAL*8` ή `DOUBLE PRECISION`
  - C για δεδομένα τύπου `COMPLEX`
  - Z για δεδομένα τύπου `COMPLEX*16` ή `DOUBLE COMPLEX`
- τα δύο επόμενα γράμματα `YY` αναφέρονται στον τύπο του πίνακα τον οποίο χειρίζεται το υποπρόγραμμα:
  - BD για διδιαγώνιους πίνακες (bidiagonal)
  - DI για διαγώνιους πίνακες (diagonal)
  - GB για γενικούς πίνακες ζώνης (general band matrices)
  - GE για γενικούς, δηλαδή, μη συμμετρικούς πίνακες

– GT για γενικούς τριδιαγώνιους πίνακες
– HB για μιγαδικούς Ερμιτιανούς πίνακες με ζώνη (complex Hermitian band matrices)
– HE για μιγαδικούς Ερμιτιανούς πίνακες (complex Hermitian matrices)
– HP για μιγαδικούς Ερμιτιανούς πίνακες αποθηκευμένους σε συμπαγή μορφή, (complex Hermitian, packed storage)
– PB για συμμετρικούς ή Ερμιτιανούς θετικά ορισμένους πίνακες ζώνης
– PO για συμμετρικούς ή Ερμιτιανούς θετικά ορισμένους πίνακες
– PP για συμμετρικούς ή Ερμιτιανούς θετικά ορισμένους πίνακες σε συμπαγή μορφή
– PT για συμμετρικούς ή Ερμιτιανούς θετικά ορισμένους τριδιαγώνιους πίνακες
– SB για πραγματικούς συμμετρικούς πίνακες ζώνης
– SP για πραγματικούς συμμετρικούς πίνακες σε συμπαγή μορφή
– ST για πραγματικούς συμμετρικούς τριδιαγώνιους πίνακες
– SY για πραγματικούς συμμετρικούς πίνακες
– TB για τριγωνικούς πίνακες ζώνης
– TP για τριγωνικούς πίνακες σε συμπαγή μορφή
– TR για τριγωνικούς πίνακες
– ... πολλές άλλες κατηγορίες πινάκων

• τα τρία τελευταία γράμματα δηλώνουν τη λειτουργία του υποπρογράμματος
  – TRF για την ανάλυση ενός πίνακα, π.χ, Cholesky, ή LU, κλπ.
  – TRS για τα υποπρογράμματα που εκτελούν οπισθοδρόμηση, λύση δηλαδή συστημάτων με χρήση της ανάλυσης του πίνακα που έγινε απο το υποπρόγραμμα xxxTRF
  – CON για τα υποπρογράμματα που εκτιμούν τον δείκτη κατάστασης ενός πίνακα
  – TRI για τα υποπρογράμματα που υπολογίζουν τον αντίστροφο ενός πίνακα

Για παράδειγμα, `SGECON` είναι ένα υποπρόγραμμα το οποίο εκτιμά τον δείκτη κατάστασης ενός πραγματικού, γενικού πίνακα και `DPBTRF` είναι ένα υποπρόγραμμα το οποίο υπολογίζει την ανάλυση Cholesky ενός συμμετρικού, θετικά ορισμένου πίνακα ζώνης διπλής ακρίβειας, ο οποίος είναι αποθηκευμένος σε συμπαγή μορφή.

1.1 Τρόποι αποθήκευσης πινάκων

Το LAPACK μπορεί να χειριστεί πίνακες οι οποίοι είναι αποθηκευμένοι με έναν απο τους ακόλουθους τρόπους:

1. Κλασσική μορφή σε διδιάστατο πίνακα. Το στοιχείο a_{ij} του πίνακα A αποθηκεύεται στη θέση `A(i,j)` του διδιάστατου πίνακα (array) `A`. Αν ο πίνακας είναι τριγωνικός τότε μόνο τα στοιχεία του πάνω ή κάτω απο την κύρια διαγώνιο χρειάζεται να αποθηκευτούν. Το ίδιο ισχύει και για συμμετρικούς ή Ερμιτιανούς πίνακες.

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ & a_{22} & a_{23} & a_{24} \\ & & a_{33} & a_{34} \\ & & & a_{44} \end{pmatrix} \quad \rightarrow \quad \begin{bmatrix} \mathtt{a}_{11} & \mathtt{a}_{12} & \mathtt{a}_{13} & \mathtt{a}_{14} \\ \star & \mathtt{a}_{22} & \mathtt{a}_{23} & \mathtt{a}_{24} \\ \star & \star & \mathtt{a}_{33} & \mathtt{a}_{34} \\ \star & \star & \star & \mathtt{a}_{44} \end{bmatrix}$$

2. Συμπαγής μορφή. Συμμετρικοί ή Ερμιτιανοί πίνακες μπορούν να αποθηκευτούν στη λεγόμενη συμπαγή μορφή, με τον ακόλουθο τρόπο: το στοιχείο a_{ij}, όπου $i \leq j$, αποθηκεύεται στη θέση $i + j(j-1)/2$ του διανύσματος `AP`. Ανάλογα, θα μπορούσαμε να αποθηκεύσουμε το στοιχείο a_{ij}, όπου $i \geq j$, στη θέση $i + (2n-j)(j-1)/2$ του διανύσματος `AP`.

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ & a_{22} & a_{23} & a_{24} \\ & & a_{33} & a_{34} \\ & & & a_{44} \end{pmatrix} \quad \rightarrow \quad \begin{bmatrix} \mathtt{a}_{11} & \mathtt{a}_{12} & \mathtt{a}_{22} & \mathtt{a}_{13} & \mathtt{a}_{23} & \mathtt{a}_{33} & \mathtt{a}_{14} & \mathtt{a}_{24} & \mathtt{a}_{34} & \mathtt{a}_{44} \end{bmatrix}$$

3. Αποθήκευση τύπου ζώνης. Ενας $m \times n$ πίνακας ζώνης με k_l υποδιαγωνίους και k_u υπερδιαγωνίους μπορεί να αποθηκευτεί σε ένα διδιάστατο πίνακα `AB` με $k_l + k_u + 1$ γραμμές και n στήλες ως εξης: το στοιχείο a_{ij}, όπου $\max\{1, j-k_u\} \leq i \leq \max\{m, j+k_l\}$, αποθηκεύεται στη θέση `AB(`k_u`+1+i-j,j)`. Για παράδειγμα, όταν $m = n = 5$ και $k_l = 2$, $k_u = 1$

$$\begin{pmatrix} a_{11} & a_{12} & & & \\ a_{21} & a_{22} & a_{23} & & \\ a_{31} & a_{32} & a_{33} & a_{34} & \\ & a_{42} & a_{43} & a_{44} & a_{45} \\ & & a_{53} & a_{54} & a_{55} \end{pmatrix} \quad \rightarrow \quad \begin{bmatrix} \star & \mathtt{a}_{12} & \mathtt{a}_{23} & \mathtt{a}_{34} & \mathtt{a}_{45} \\ \mathtt{a}_{11} & \mathtt{a}_{22} & \mathtt{a}_{33} & \mathtt{a}_{44} & \mathtt{a}_{55} \\ \mathtt{a}_{21} & \mathtt{a}_{32} & \mathtt{a}_{43} & \mathtt{a}_{54} & \star \\ \mathtt{a}_{31} & \mathtt{a}_{42} & \mathtt{a}_{53} & \star & \star \end{bmatrix}$$

   Προφανώς, συμμετρικοί ή Ερμιτιανοί πίνακες ζώνης μπορούν να αποθηκευτούν με παρόμοιο τρόπο και με ακόμα μεγαλύτερη οικονομία μνήμης.

   Τριδιαγώνιοι, μη συμμετρικοί πίνακες τάξης n αποθηκεύονται σε τρία διανύσματα: το πρώτο, μήκους n, περιέχει τα διαγώνια στοιχεία και τα άλλα δύο, μήκους $n-1$, τα στοιχεία της υπο- και υπερ-διαγωνίου στις θέσεις 1 ως $n-1$. Αν ο τριδιαγώνιος πίνακας είναι συμμετρικός, τότε μόνο δύο διανύσματα χρησιμοποιούνται.

1.2 Υπολογιστικά υποπρογράμματα

Παραθέτουμε εδώ τον τρόπο κλήσης μερικών απο τα συχνότερα χρησιμοποιούμενα υποπρογράμματα του LAPACK. Για τα υπόλοιπα μπορείται να συμβουλεύεστε τον οδηγό χρήσης στην ηλεκτρονική διεύθυνση http://www.netlib.org/lapack/lug.

- `SUBROUTINE DGBTRF(M, N, KL, KU, AB, LDAB, IPIV, INFO)`

  Υπολογίζει την ανάλυση LU ενός πραγματικού $m \times n$ πίνακα ζώνης A χρησιμοποιώντας μερική οδήγηση.

Ορισμα	Ερμηνεία
`M,N`	Αριθμός γραμμών και στηλών του A
`KL,KU`	Αριθμός υπο- και υπερ-διαγωνίων του A
`AB`	Πίνακας διπλής ακρίβειας διάστασης `(LDAB,N)`. Στην είσοδο περιέχει τα στοιχεία του A με αποθήκευση τύπου ζώνης. Στην έξοδο περιέχει την ανάλυση LU του A.
`LDAB`	Αριθμός γραμμών του πίνακα `AB`. Πρέπει `LDAB` $\geq$ `2*LK+KU+1`.
`IPIV`	Ακέραιο διάνυσμα μήκους min(`M,N`) που περιέχει πληροφορίες για τις εναλλαγές γραμμών κατα τη διάρκεια της ανάλυσης LU.
`INFO`	Ακέραιος αριθμός. Αν `INFO` $= 0$ τότε η ανάλυση LU υπολογίστηκε χωρίς πρόβλημα. Αν `INFO` $= -i$ τότε το i-στό όρισμα είχε μη επιτρεπτή τιμη. Αν `INFO` $= i$ τότε το U_{ii} είναι μηδέν οπότε ο πίνακας U της ανάλυσης LU είναι μη αντιστρέψιμος.

- `SUBROUTINE DGBTRS(TRANS, N, KL, KU, NRHS, AB, LDAB, IPIV, B, LDB, INFO)`

  Λύνει το σύστημα εξισώσεων $AX = B$ ή $A^T X = B$ χρησιμοποιώντας την ανάλυση LU που υπολόγισε η DGBTRF.

Ορισμα	Ερμηνεία
TRANS	Σταθερά τύπου χαρακτήρα (CHARACTER*1). Αν TRANS = 'N' τότε λύνεται το σύστημα $AX = B$. Διαφορετικά, αν TRANS = 'T' ή TRANS = 'C' τότε λύνεται το σύστημα $A^T X = B$.
NRHS	Αριθμός των δεξιών μελών ή ο αριθμός των στηλών του πίνακα B.
LDB	Αριθμός γραμμών του πίνακα B. Πρέπει LDB $\geq \max(1, \mathtt{N})$.
INFO	Ακέραιος αριθμός. Αν INFO $= 0$ τότε η λύση LU υπολογίστηκε χωρίς πρόβλημα. Αν INFO $= -i$ τότε το i-στό όρισμα είχε μη επιτρεπτή τιμη.

  Τα υπόλοιπα ορίσματα είναι όπως ακριβώς και στην DGBTRF.

- `SUBROUTINE DGETRF(M, N, A, LDA, IPIV, INFO)`

  Υπολογίζει την ανάλυση LU ενός γενικού $m \times n$ πίνακα A χρησιμοποιώντας μερική οδήγηση με εναλλαγές γραμμών. Η ανάλυση έχει τη μορφή $A = PLU$, όπου P είναι ένας πίνακας μετάθεσης, L είναι ένα κάτω τριγωνικός πίνακας με μοναδιαία διαγώνια στοιχεία και U είναι ένας άνω τριγωνικός πίνακας.

Ορισμα	Ερμηνεία
M,N	Αριθμός γραμμών και στηλών του A
A	Πίνακας διπλής ακρίβειας διάστασης (LDA,N). Στην είσοδο περιέχει τα στοιχεία του A. Στην έξοδο περιέχει την ανάλυση PLU του A.
LDA	Αριθμός γραμμών του πίνακα A. Πρέπει LDA $\geq \max(1, \mathtt{M})$.
IPIV	Ακέραιο διάνυσμα μήκους $\min(\mathtt{M,N})$ που περιέχει πληροφορίες για τις εναλλαγές γραμμών κατα τη διάρκεια της ανάλυσης PLU.
INFO	Ακέραιος αριθμός. Αν INFO $= 0$ τότε η ανάλυση PLU υπολογίστηκε χωρίς πρόβλημα. Αν INFO $= -i$ τότε το i-στό όρισμα είχε μη επιτρεπτή τιμη. Αν INFO $= i$ τότε το U_{ii} είναι μηδέν οπότε ο πίνακας U της ανάλυσης PLU είναι μη αντιστρέψιμος.

- `SUBROUTINE DGETRS(TRANS, N, NRHS, A, LDA, IPIV, B, LDB, INFO)`

  Λύνει το σύστημα εξισώσεων $AX = B$ ή $A^T X = B$ χρησιμοποιώντας την ανάλυση PLU που υπολόγισε η DGETRF.

Ορισμα	Ερμηνεία
TRANS	Σταθερά τύπου χαρακτήρα (CHARACTER*1). Αν TRANS = 'N' τότε λύνεται το σύστημα $AX = B$. Διαφορετικά, αν TRANS = 'T' ή TRANS = 'C' τότε λύνεται το σύστημα $A^T X = B$.
NRHS	Αριθμός των δεξιών μελών ή ο αριθμός των στηλών του πίνακα B.
LDB	Αριθμός γραμμών του πίνακα B. Πρέπει LDB $\geq \max(1, \mathtt{N})$.
INFO	Ακέραιος αριθμός. Αν INFO $= 0$ τότε η λύση υπολογίστηκε χωρίς πρόβλημα. Αν INFO $= -i$ τότε το i-στό όρισμα είχε μη επιτρεπτή τιμη.

  Τα υπόλοιπα ορίσματα είναι όπως ακριβώς και στην DGETRF.

- `SUBROUTINE DGECON(NORM, N, A, LDA, ANORM, RCOND, WORK, IWORK, INFO)`

  Εκτιμά τον αντίστροφο του δείκτη κατάστασης ενός γενικού $n \times n$ πίνακα A είτε στη νόρμα $\|\cdot\|_1$ είτε στη νόρμα $\|\cdot\|_\infty$, χρησιμοποιώντας την ανάλυση PLU που υπολόγισε η DGETRF.

Ορισμα	Ερμηνεία
NORM	Σταθερά τύπου χαρακτήρα (CHARACTER*1). Αν NORM = '1' τότε χρησιμοποείται η νόρμα $\|\cdot\|_1$. Διαφορετικά, αν NORM = 'I' τότε χρησιμοποείται η νόρμα $\|\cdot\|_\infty$.
ANORM	Μεταβλητή διπλής ακρίβειας. Αν NORM = '1' τότε ANORM $= \|A\|_1$, διαφορετικά, αν NORM = 'I' τότε ANORM $= \|A\|_\infty$.
RCOND	Μεταβλητή διπλής ακρίβειας. Περιέχει τον αντίστροφο του δείκτη κατάσταση του πίνακα A.
WORK	Βοηθητικό διάνυσμα διπλής ακρίβειας, μήκους τουλάχιστον 4N.
IWORK	Ακέραιο βοηθητικό διάνυσμα, μήκους τουλάχιστον N.
INFO	Ακέραιος αριθμός. Αν INFO $= 0$ τότε ο δείκτης κατάστασης εκτιμήθηκε χωρίς πρόβλημα. Αν INFO $= -i$ τότε το i-στό όρισμα είχε μη επιτρεπτή τιμη.

  Τα υπόλοιπα ορίσματα είναι όπως ακριβώς και στην DGETRF.

- `SUBROUTINE DPPTRF(UPLO, N, AP, INFO)`

  Υπολογίζει την ανάλυση Cholesky ενός πραγματικού, συμμετρικού και θετικά ορισμένου πίνακα A ο οποίος αποθηκεύεται σε συμπαγή μορφή. Η ανάλυση έχει τη μορφή $A = U^T U$ αν στο διάνυσμα AP έχει αποθηκευτεί το πάνω τρίγωνο του A ή τη μορφή $A = LL^T$ αν στο διάνυσμα AP έχει αποθηκευτεί το κάτω τρίγωνο του A.

Ορισμα	Ερμηνεία
UPLO	Σταθερά τύπου χαρακτήρα (CHARACTER*1). Αν UPLO = 'U' τότε αποθηκεύεται το πάνω τρίγωνο του A. Διαφορετικά, αν UPLO = 'L' τότε αποθηκεύεται το κάτω τρίγωνο του A.
N	Η τάξη του πίνακα A.
AP	Διάνυσμα τύπου DOUBLE PRECISION, μήκους N*(N+1)/2. Στην είσοδο περιέχει το κάτω ή το πάνω τρίγωνο του A, ανάλογα με την τιμή του ορίσματος UPLO. Στην έξοδο, περιέχει τον πίνακα U ή τον πίνακα L της ανάλυσης Cholesky.
INFO	Ακέραιος αριθμός. Αν INFO $= 0$ τότε η ανάλυση Cholesky έγινε χωρίς πρόβλημα. Αν INFO $= -i$ τότε το i-στό όρισμα είχε μη επιτρεπτή τιμη. Αν INFO $= i$, τότε ο κύριος υποπίνακας τάξης i δεν είναι θετικά ορισμένος.

- `SUBROUTINE DPPTRS(UPLO, N, NRHS, AP, B, LDB, INFO)`

  Λύνει το γραμμικό σύστημα $AX = B$, όπου A ένας συμμετρικός και θετικά ορισμένος πίνακας του οποίου η ανάλυση Cholesky έχει υπολογιστεί απο την DPPTRF.

Ορισμα	Ερμηνεία
NRHS	Αριθμός των δεξιών μελών ή ο αριθμός των στηλών του πίνακα B.
LDB	Αριθμός γραμμών του πίνακα B. Πρέπει LDB $\geq \max(1, \mathtt{N})$.
INFO	Ακέραιος αριθμός. Αν INFO $= 0$ τότε η λύση υπολογίστηκε χωρίς πρόβλημα. Αν INFO $= -i$ τότε το i-στό όρισμα είχε μη επιτρεπτή τιμη.

  Τα υπόλοιπα ορίσματα είναι όπως ακριβώς και στην DPPTRF.

1.3 Παράδειγμα χρήσης των DPPTRF και DPPTRS

Εστω $A_n(c,d)$ ο πίνακας

$$A_n(c,d) = \begin{bmatrix} d & c & & \\ c & d & \ddots & \\ & \ddots & \ddots & c \\ & & c & d \end{bmatrix}$$

Πίνακες αυτής της μορφής εμφανίζονται συχνά στην αριθμητική επίλυση προβλημάτων αρχικών και συνοριακών τιμών για μερικές διαφορικές εξισώσεις, με τη μέθοδο πεπερασμένων διαφορών ή πεπερασμένων στοιχείων.

Ας υποθέσουμε ότι θέλουμε να λύσουμε το σύστημα $A_n x = b$ για δοσμένο διάνυσμα $b \in \mathbb{R}^n$, με την χρήση των **DPPTRF** για τον υπολογισμό της ανάλυσης Cholesky, και της **DPPTRS** για την οπισθοδρόμηση. Και οι δυο αυτές υπορουτίνες υποθέτουν οτι έχουμε δώσει τον πίνακα σε συμπαγή μορφή, δηλαδή ότι έχουμε αποθηκεύσει σε ένα διάνυσμα είτε τις στήλες του άνω τριγώνου του A_n, είτε τις γραμμές του κάτω τριγώνου του A_n. Αυτό, στην περίπτωση που αποθηκεύουμε τις στήλες του άνω τριγώνου του A_n μπορεί να γίνει με τις παρακάτω εντολές:

```
      c = ...
      d = ...
c
c     Oi sthles tou anw trigwnou tou A_n apo8hkeuontai sto dianysma AP
c
      k = 1
      ap(k) = d
      k = k + 1

      do j=2, n
         do i=1, j-2
            ap(k) = 0.0d0
            k = k + 1
         enddo

         ap(k) = c
         k = k + 1

         ap(k) = d
         k = k + 1
      enddo
```

Στο παρακάτω πρόγραμμα λύνουμε το σύστημα $A_n(1,4)$ με δεξί μέλος το διάνυμα $b = (1,1,\ldots,1)$:

```
c
      program lapacktest
c
c     Paradeigma xrhshs twn DPPTRF kai DPPTRS
c     Lynoume to systhma A_n(1,4) x = b, me dexi melos b=(1,1,...,1)
c
      implicit double precision (a-h, o-z)
```

```
      parameter (LDA = 40)
      double precision ap((LDA*(LDA+1))/2), b(LDA)
c
c     Kataskeuh tou pinaka AP
c
      c = 1.0d0
      d = 4.0d0

      n = 10
c
      k = 1
      ap(k) = d
      k = k + 1

      do j=2, n
         do i=1, j-2
            ap(k) = 0.0d0
            k = k + 1
         enddo

         ap(k) = c
         k = k + 1

         ap(k) = d
         k = k + 1
      enddo
c
c     Klhsh ths DPPTRF gia thn analysh Cholesky
c
      call dpptrf('U', n, ap, info)

      if (info .lt. 0) then
         print*, 'To ', i, 'orisma eixe mh epitrepth timh'
         stop
      else if (info .gt. 0) then
         print*, 'O ypopinakas taxhs ', i, ' den einai 8etika orismenos'
         stop
      endif
c
c     Kataskeuh tou dexiou melous kai lysh tou systhmatos
c
      do i=1, n
         b(i) = 1.0d0
      enddo
c
      LDB  = LDA
      NRHS = 1
```

```
      call dpptrs('U', n, NRHS, ap, b, LDB, info)
c
      if (info .lt. 0) then
         print*, 'To ', i, 'orisma eixe mh epitrepth timh'
         stop
      endif
c
      print*, 'Lysh tou systhmatos'
      write(6,10) (b(i), i=1, n)
 10   format(3d16.9)
c
      stop
      end
```

2 Κλήση υποπρογραμμάτων του LAPACK απο προγράμματα σε C

Αν και το LAPACK είναι γραμμένο σε Fortran 77, είναι δυνατόν να χρησιμοποιηθεί μέσα απο ένα πρόγραμμα σε C, αλλά απαιτείται λίγο παραπάνω δουλειά.

Η δυσκολία βρίσκεται στον διαφορετικό τρόπο με τον οποίο η C και η Fortran 77 αποθηκεύουν πολυδιάστατους πίνακες αλλά και στον τρόπο με τον οποίο μεταβιβάζονται τα ορίσματα μιας συνάρτησης. Η πρώτη δυσκολία είναι σχετικά εύκολη να αντιμετωπιστεί: μια και η Fortran 77 αποθηκεύει τους διδιάστατους πίνακες κατα στήλες και η C κατά γραμμές, θα πρέπει να δώσουμε στη συνάρτηση του LAPACK τον ανάστροφο του πίνακα αντί για τον ίδιο τον πίνακα. Ενα παράδειγμα: Εστω ότι θέλουμε να υπολογίσουμε την ανάλυση LU του 2×2 πίνακα

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$$

χρησιμοποιώντας το υποπρόγραμμα DGETRF (δείτε τα ορίσματά του στην παράγραφο 1.2). Επειδή η Fortran 77 αποθηκεύει διδιάστατους πίνακες κατά στήλες, αυτό που μεταβιβάζεται στο υποπρόγραμμα DGETRF είναι η διεύθυνση του πρώτου στοιχείου του διανύσματος

a_{11}	a_{21}	a_{12}	a_{22}

Σε αντίθεση, ο ίδιος πίνακας στην C αποθηκεύεται ως

a_{11}	a_{12}	a_{21}	a_{22}

άρα, αυτό που μεταβιβάζεται στην DGETRF είναι ο A^T και όχι ο A!! Η λύση είναι λοιπόν απλή: Αντί για τον ίδιο τον A αποθηκεύουμε τον A^T. Ως γνωστόν, η Fortran 77 και η C διαφέρουν επίσης και στον τρόπο με τον οποίο μεταβιβάζουν ορίσματα σε υποπρογράμματα ή συναρτήσεις. Στην Fortran 77 τα ορίσματα μεταβιβάζονται κατ΄ αναφορά, δηλαδή μεταβιβάζονται στη συνάρτηση οι διευνθύσεις των ορισμάτων. Στην C, τα ορίσματα μεταβιβάζονται κατα τιμή, δηλαδή τα ορίσματα γράφονται σε προσωρινές θέσεις μνήμης και οι διευθύνσεις τους μεταβιβάζονται στη συνάρτηση.

Μια τελευταία λεπτομέρεια: Επειδή η Fortran 77 προσθέτει τον χαρακτήρα ΄_΄ στο τέλος κάθε συνάρτησης, αντί για

```
dgetrf(...)
```

θα πρέπει να γράψετε

```
dgetrf_(...)
```

για να καλέσετε το υποπρόγραμμα DGETRF απο ένα C πρόγραμμα. Η μεταγλώττιση του προγράμματος και η σύνδεση του με τη βιβλιοθήκη LAPACK γίνεται με το συνηθισμένο τρόπο, αλλά χρειάζεται να συνδέσουμε το πρόγραμμα μας και με την βιβλιοθήκη g2c.

```
> gcc -o myprog myprog.c -llapack -lg2c
> ./myprog
```

2.1 Παράδειγμα κλήσης των DGETRF και DGETRS απο προγράμματα σε C

```
/*
 * Paradeigma xrhshs twn DGETRF kai DGETRS. Lynoume to systhma A x = b
 * opou
 *
 *            [  1  0  3 ]           [ 2 ]
 *        A = [ -1  0  1 ]  kai  b = [ 0 ]
 *            [ -2 -1  0 ]           [ 6 ]
 *
 * H akribhs lysh einai [1/2, -7, 1/2].
*/

#include <stdio.h>

#define MAXDIM  10

int main()
{
   double A[MAXDIM][MAXDIM], b[MAXDIM];
   int    ipiv[MAXDIM];

   int    n, lda, info, nrhs, ldb, i;
   char   trans;
/*
 * Bazoume sto didiastato pinaka A ton pinaka toy systhmatos kata sthles!!
*/
   A[0][0] =  1.0;
   A[0][1] = -1.0;
   A[0][2] = -2.0;

   A[1][0] =  0.0;
   A[1][1] =  0.0;
   A[1][2] = -1.0;

   A[2][0] =  3.0;
   A[2][1] =  1.0;
   A[2][2] =  0.0;
```

```
/*
 * Kaloume thn DGETRF gia thn analysh LU kai elegxoume thn timh tou info
*/
   n = 3;  lda = MAXDIM;  info = 0;

   dgetrf_(&n, &n, A, &lda, ipiv, &info);

   if (info < 0) {
      printf("To %d orisma eixe mh epitrepth timh\n", info);
      exit(2);
   }
   else if (info > 0) {
      printf("To %d stoixeio ths diagvniou tou U einai mhden\n", info);
      exit(2);
   }
/*
 * To dexi melos
*/
   b[0] = 2.0;
   b[1] = 0.0;
   b[2] = 6.0;
/*
 * Lynoume to systhma twra me thn DGETRS
*/
   trans = 'N';  nrhs = 1;  ldb = MAXDIM;

   dgetrs_(&trans, &n, &nrhs, A, &lda, ipiv, b, &ldb, &info);

   if (info < 0) {
      printf("To %d orisma eixe mh epitrepth timh\n", info);
      exit(2);
   }

   printf("\nLysh tou systhmatos:\n");
   for (i = 0; i < n; i++)
      printf("%16.9e\n", b[i]);

   return 0;
}
```