Κεφάλαιο 2

Συστήματα γραμμικών εξισώσεων

Η επίλυση γραμμικών συστημάτων είναι το συχνότερο υπολογιστικό πρόβλημα στις επιστήμες του μηχανικού: σχεδόν κάθε αριθμητική μέθοδος (παρεμβολή, ελάχιστα τετράγωνα, διαφορικές εξισώσεις, πεπερασμένα στοιχεία) καταλήγει σε ένα σύστημα $A\mathbf{x} = \mathbf{b}$. Το κεφάλαιο καλύπτει τις βασικές πράξεις πινάκων, τις *άμεσες* μεθόδους (απαλοιφή Gauss και Gauss–Jordan), τις *επαναληπτικές* μεθόδους (Jacobi και Gauss–Seidel) και τον υπολογισμό ιδιοτιμών. Ως επεκτάσεις, στις ασκήσεις εμφανίζονται η παραγοντοποίηση LU, ο αλγόριθμος Thomas, η μέθοδος SOR, η παραγοντοποίηση Cholesky και ο δείκτης κατάστασης.

2.1 Σύνοψη θεωρίας

2.1.1 Πίνακες και βασικές πράξεις

Ένα σύστημα n γραμμικών εξισώσεων με n αγνώστους,

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1, \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2, \\ &\vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n &= b_n, \end{aligned} \tag{2.1}$$

γράφεται συνοπτικά $A\mathbf{x} = \mathbf{b}$, όπου $A = (a_{ij})$ είναι ο πίνακας (μητρώο) των συντελεστών, $\mathbf{x}$ το διάνυσμα των αγνώστων και $\mathbf{b}$ το διάνυσμα των δεύτερων μελών.

Βασικές πράξεις και έννοιες (αναλυτικά στην Ενότητα 1.6)

- **Ανάστροφος:** $(A^\top)_{ij} = a_{ji}$. Ο A είναι *συμμετρικός* αν $A = A^\top$.
- **Άθροισμα:** $(A+B)_{ij} = a_{ij} + b_{ij}$, για πίνακες ίδιων διαστάσεων.
- **Γινόμενο:** αν ο A είναι $m \times p$ και ο B είναι $p \times n$, τότε $(AB)_{ij} = \sum_{k=1}^{p} a_{ik} b_{kj}$. Γενικά $AB \neq BA$, ενώ $(AB)^\top = B^\top A^\top$.
- **Τάξη** $\operatorname{rank}(A)$: το μέγιστο πλήθος γραμμικά ανεξάρτητων γραμμών (ή στηλών).
- **Ορίζουσα:** για πίνακα 3×3, $\det A = a_{11}(a_{22}a_{33} - a_{23}a_{32}) - a_{12}(a_{21}a_{33} - a_{23}a_{31}) + a_{13}(a_{21}a_{32} - a_{22}a_{31})$.
- **Αντίστροφος:** αν $\det A \neq 0$ υπάρχει μοναδικός A^{-1} με $AA^{-1} = A^{-1}A = I$, και τότε $\mathbf{x} = A^{-1}\mathbf{b}$.
- **Ιδιοτιμές και ιδιοδιανύσματα:** $A\mathbf{v} = \lambda\mathbf{v}$ με $\mathbf{v} \neq \mathbf{0}$· οι ιδιοτιμές είναι οι ρίζες του χαρακτηριστικού πολυωνύμου $\det(\lambda I - A) = 0$.

Τα ακόλουθα είναι ισοδύναμα για τετραγωνικό πίνακα A τάξης n: το σύστημα $A\mathbf{x} = \mathbf{b}$ έχει μοναδική λύση για κάθε $\mathbf{b}$ $\iff$ $\det A \neq 0$ $\iff$ $\operatorname{rank}(A) = n$ $\iff$ υπάρχει ο A^{-1} $\iff$ το 0 δεν είναι ιδιοτιμή του A.

Ο Πίνακας 2.1 δίνει τις αντίστοιχες ενσωματωμένες εντολές του Octave/MATLAB (βλ. και την Ενότητα 1.6) και τις συναρτήσεις της βιβλιοθήκης του κεφαλαίου (Ενότητα 2.2), που υλοποιούν τους ίδιους υπολογισμούς με τους αλγορίθμους που περιγράφονται παρακάτω.

Πράξη	ενσωματωμένη εντολή	συνάρτηση του κεφαλαίου
ορισμός	`A = [1 2; 3 4]`	
ανάστροφος	`A'`	
άθροισμα, γινόμενο	`A + B`, `A * B`	
γινόμενο πίνακα επί διάνυσμα	`A * x`	`matvec(A, x)`
τάξη	`rank(A)`	`rank_gauss(A)`
ορίζουσα	`det(A)`	`det_gauss(A)`
αντίστροφος	`inv(A)`	`gauss_jordan(A, eye(n))`
κυρίαρχη ιδιοτιμή	`max(abs(eig(A)))`	`power_method(A, ...)`
επίλυση $A\mathbf{x} = \mathbf{b}$	`A \ b`	`gauss_solve(A, b)`

Πίνακας 2.1: Πράξεις πινάκων: ενσωματωμένες εντολές και συναρτήσεις του κεφαλαίου.

2.1.2 Νόρμες και δείκτης κατάστασης

Για να μετρηθεί το μέγεθος ενός σφάλματος χρειάζεται μια νόρμα. Στο κεφάλαιο χρησιμοποιείται η νόρμα μεγίστου

$$\|\mathbf{x}\|_\infty = \max_i |x_i|, \qquad \|A\|_\infty = \max_i \sum_{j=1}^{n} |a_{ij}| \quad \text{(μέγιστο άθροισμα γραμμής)}.$$

Ο *δείκτης κατάστασης* $\kappa(A) = \|A\|\,\|A^{-1}\| \geq 1$ μετρά την ευαισθησία της λύσης στα δεδομένα: αν το $\mathbf{b}$ διαταραχθεί κατά $\delta\mathbf{b}$, η λύση μεταβάλλεται κατά $\delta\mathbf{x}$ με

$$\frac{\|\delta\mathbf{x}\|}{\|\mathbf{x}\|} \leq \kappa(A)\,\frac{\|\delta\mathbf{b}\|}{\|\mathbf{b}\|}. \tag{2.2}$$

Πρακτικός κανόνας: αν $\kappa(A) \approx 10^p$, χάνονται περίπου p από τα 16 σημαντικά ψηφία της διπλής ακρίβειας, *όποια μέθοδος κι αν χρησιμοποιηθεί*. Το *υπόλοιπο* $\mathbf{r} = \mathbf{b} - A\tilde{\mathbf{x}}$ μιας υπολογισμένης λύσης μπορεί να είναι πολύ μικρό ακόμη και όταν το σφάλμα $\tilde{\mathbf{x}} - \mathbf{x}$ είναι μεγάλο.

2.1.3 Απαλοιφή Gauss

Η απαλοιφή Gauss μετατρέπει το σύστημα σε ισοδύναμο άνω τριγωνικό με τις στοιχειώδεις πράξεις γραμμών: εναλλαγή δύο εξισώσεων, πολλαπλασιασμό εξίσωσης με μη μηδενικό αριθμό και πρόσθεση πολλαπλασίου μιας εξίσωσης σε άλλη. Καμία από αυτές δεν αλλάζει τη λύση.

Απαλοιφή Gauss με μερική οδήγηση

Σχηματίζεται ο επαυξημένος πίνακας $[A \,|\, \mathbf{b}]$. Για $k = 1, \dots, n-1$:

1. **Οδήγηση:** βρίσκεται η γραμμή $p \geq k$ με το μεγαλύτερο $|a_{pk}|$ και εναλλάσσεται με τη γραμμή k. Το a_{kk} λέγεται *οδηγός*.
2. **Απαλοιφή:** για κάθε γραμμή $i = k+1, \dots, n$ υπολογίζεται ο πολλαπλασιαστής $m_{ik} = a_{ik}/a_{kk}$ και

$$a_{ij} \leftarrow a_{ij} - m_{ik}\, a_{kj} \quad (j = k, \dots, n), \qquad b_i \leftarrow b_i - m_{ik}\, b_k.$$

Πίσω αντικατάσταση: για $i = n, n-1, \dots, 1$: $\displaystyle x_i = \frac{1}{a_{ii}}\Big(b_i - \sum_{j=i+1}^{n} a_{ij} x_j\Big)$.

Κόστος: περίπου $n^3/3$ πολλαπλασιασμοί για την απαλοιφή και $n^2/2$ για την πίσω αντικατάσταση. **Ορίζουσα:** $\det A = (-1)^s\, a_{11} a_{22} \cdots a_{nn}$, όπου s το πλήθος των εναλλαγών και a_{ii} οι οδηγοί.

Η οδήγηση είναι απαραίτητη όταν εμφανιστεί μηδενικός οδηγός, αλλά και γενικά: ένας πολύ μικρός οδηγός δίνει τεράστιους πολλαπλασιαστές, που μεγεθύνουν τα σφάλματα στρογγύλευσης (Άσκηση 2.11). Με μερική οδήγηση ισχύει πάντα $|m_{ik}| \leq 1$.

2.1.4 Απαλοιφή Gauss–Jordan και αντίστροφος πίνακας

Στη μέθοδο Gauss–Jordan ο άγνωστος x_k απαλείφεται στο βήμα k από *όλες* τις άλλες εξισώσεις, και πάνω και κάτω από τη διαγώνιο. Ο πίνακας γίνεται διαγώνιος (μοναδιαίος, αν κάθε γραμμή οδηγού διαιρεθεί με τον οδηγό) και η λύση διαβάζεται απευθείας στην τελευταία στήλη, χωρίς πίσω αντικατάσταση. Το κόστος είναι περίπου $n^3/2$ πολλαπλασιασμοί, δηλαδή 50% μεγαλύτερο από της απαλοιφής Gauss.

Η μέθοδος είναι ιδιαίτερα βολική για τον *αντίστροφο*: εφαρμόζεται στον επαυξημένο πίνακα $[A \,|\, I]$, ο οποίος μετατρέπεται σε $[I \,|\, A^{-1}]$. Στην πράξη πάντως ο αντίστροφος σπάνια χρειάζεται· για την επίλυση $A\mathbf{x} = \mathbf{b}$ δεν υπολογίζεται ποτέ ο A^{-1}, επειδή αυτό κοστίζει τριπλάσια και είναι λιγότερο ακριβές.

2.1.5 Επαναληπτικές μέθοδοι Jacobi και Gauss–Seidel

Οι επαναληπτικές μέθοδοι ξεκινούν από μια αρχική εκτίμηση $\mathbf{x}^{(0)}$ και παράγουν ακολουθία προσεγγίσεων. Είναι προτιμότερες για πολύ μεγάλα και *αραιά* συστήματα (με ελάχιστα μη μηδενικά στοιχεία ανά γραμμή), όπου η απαλοιφή θα γέμιζε τον πίνακα με μη μηδενικά στοιχεία. Γράφουμε $A = D - L - U$, με D το διαγώνιο μέρος του A και $-L$, $-U$ το αυστηρά κάτω και άνω τριγωνικό μέρος.

Jacobi και Gauss–Seidel

Jacobi: $\mathbf{x}^{(k+1)} = D^{-1}(L+U)\,\mathbf{x}^{(k)} + D^{-1}\mathbf{b}$, δηλαδή

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\Big(b_i - \sum_{j\neq i} a_{ij}\,x_j^{(k)}\Big), \qquad i = 1,\dots,n.$$

Gauss–Seidel: $\mathbf{x}^{(k+1)} = (D-L)^{-1}U\,\mathbf{x}^{(k)} + (D-L)^{-1}\mathbf{b}$, δηλαδή

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\Big(b_i - \sum_{j<i} a_{ij}\,x_j^{(k+1)} - \sum_{j>i} a_{ij}\,x_j^{(k)}\Big), \qquad i = 1,\dots,n.$$

Η Gauss–Seidel χρησιμοποιεί αμέσως κάθε νέα συνιστώσα μόλις υπολογιστεί.

Σύγκλιση. Μια επανάληψη $\mathbf{x}^{(k+1)} = B\mathbf{x}^{(k)} + \mathbf{c}$ συγκλίνει για κάθε $\mathbf{x}^{(0)}$ αν και μόνο αν η *φασματική ακτίνα* $\rho(B) = \max|\lambda_i(B)|$ είναι μικρότερη του 1· το σφάλμα μειώνεται ασυμπτωτικά κατά τον παράγοντα $\rho(B)$ ανά βήμα. Ικανή συνθήκη και για τις δύο μεθόδους είναι η *αυστηρή διαγώνια κυριαρχία*: $|a_{ii}| > \sum_{j\neq i}|a_{ij}|$ για κάθε i. Η Gauss–Seidel συγκλίνει επίσης για κάθε συμμετρικό και θετικά ορισμένο A.

Κριτήριο τερματισμού για ακρίβεια m σημαντικών ψηφίων:

$$\frac{\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\|}{\|\mathbf{x}^{(k+1)}\|} \le \tfrac{1}{2}\,10^{-m}.$$

Και οι δύο μέθοδοι απαιτούν $a_{ii} \neq 0$. Το κριτήριο τερματισμού ελέγχει τη *μεταβολή* μεταξύ διαδοχικών προσεγγίσεων και όχι το σφάλμα· όταν η σύγκλιση είναι αργή (ρ κοντά στο 1), το πραγματικό σφάλμα είναι πολύ μεγαλύτερο από τη μεταβολή (Άσκηση 2.23).

2.1.6 Αριθμητικός υπολογισμός ιδιοτιμών

Για πίνακες μεγαλύτερους από 3×3 οι ιδιοτιμές δεν υπολογίζονται από το χαρακτηριστικό πολυώνυμο αλλά επαναληπτικά.

Μέθοδος των δυνάμεων και αντίστροφη επανάληψη

Μέθοδος των δυνάμεων. Από αρχικό διάνυσμα $\mathbf{v}_0$:

$$\mathbf{w} = A\mathbf{v}_k, \qquad \lambda^{(k)} = \mathbf{v}_k^{\top}\mathbf{w}, \qquad \mathbf{v}_{k+1} = \mathbf{w}/\|\mathbf{w}\|_2.$$

Αν $|\lambda_1| > |\lambda_2| \ge \dots$, τότε $\lambda^{(k)} \to \lambda_1$ (η κυρίαρχη ιδιοτιμή) και $\mathbf{v}_k$ τείνει στο αντίστοιχο ιδιοδιάνυσμα, με ρυθμό $|\lambda_2/\lambda_1|$.

Αντίστροφη επανάληψη με μετατόπιση μ. Η μέθοδος των δυνάμεων εφαρμόζεται στον $(A-\mu I)^{-1}$: σε κάθε βήμα λύνεται το σύστημα $(A-\mu I)\mathbf{w} = \mathbf{v}_k$ και $\lambda^{(k)} = \mu + 1/(\mathbf{v}_k^{\top}\mathbf{w})$. Συγκλίνει στην ιδιοτιμή του A που βρίσκεται *πλησιέστερα στο* μ.

2.2 Υλοποιήσεις αναφοράς

Οι αλγόριθμοι του κεφαλαίου υλοποιούνται ως συναρτήσεις Octave/MATLAB:

συνάρτηση	περιγραφή
`gauss_solve`	επίλυση $A\mathbf{x} = \mathbf{b}$ με απαλοιφή Gauss και μερική οδήγηση
`det_gauss`, `rank_gauss`	ορίζουσα και τάξη με απαλοιφή
`gauss_jordan`	επίλυση $AX = B$ με πολλά δεύτερα μέλη· με $B = I$ δίνει τον A^{-1}
`jacobi`, `gauss_seidel`	επαναληπτικές μέθοδοι
`power_method`, `inverse_power`	κυρίαρχη ιδιοτιμή· ιδιοτιμή πλησιέστερη σε μ

Όλες οι συναρτήσεις εργάζονται σε *αντίγραφο* του πίνακα, ώστε ο αρχικός A να μένει αμετάβλητος. Οι επαναληπτικές μέθοδοι επιστρέφουν, εκτός από τη λύση, το πλήθος των επαναλήψεων και μια ένδειξη σύγκλισης.

Οι πράξεις γραμμών γράφονται με υποπίνακες (Ενότητα 1.6): `M(i, k:n)` είναι τα στοιχεία της γραμμής i από τη στήλη k έως τη n, και `M([k p], :) = M([p k], :)` εναλλάσσει δύο γραμμές. Κάθε συνάρτηση βρίσκεται σε δικό της αρχείο.

Octave/MATLAB `gauss_solve.m`

```
function x = gauss_solve(A, b)
% GAUSS_SOLVE  Solve A x = b by Gaussian elimination with partial pivoting.
  n = size(A, 1);
  M = [A, b(:)];                          % augmented matrix [A | b]
  for k = 1:n - 1
    [~, p] = max(abs(M(k:n, k)));         % pivot row (relative index)
    p = p + k - 1;
    if M(p, k) == 0
      error('gauss_solve: singular matrix');
    end
    if p ~= k
      M([k p], :) = M([p k], :);          % row exchange
    end
    for i = k + 1:n
      m = M(i, k) / M(k, k);
      M(i, k:n + 1) = M(i, k:n + 1) - m * M(k, k:n + 1);
    end
  end
  if M(n, n) == 0
    error('gauss_solve: singular matrix');
  end
  x = zeros(n, 1);
  for i = n:-1:1                          % back substitution
    s = M(i, n + 1);
    for j = i + 1:n
      s = s - M(i, j) * x(j);
    end
    x(i) = s / M(i, i);
  end
end
```

Octave/MATLAB `det_gauss.m`

```
function d = det_gauss(A)
% DET_GAUSS  Determinant = (+-) product of the pivots of Gaussian elimination.
  n = size(A, 1);
  d = 1;
  for k = 1:n - 1
    [~, p] = max(abs(A(k:n, k)));
    p = p + k - 1;
    if A(p, k) == 0
```

```
      d = 0;  return;
    end
    if p ~= k
      A([k p], :) = A([p k], :);
      d = -d;                              % each exchange changes the sign
    end
    for i = k + 1:n
      m = A(i, k) / A(k, k);
      A(i, k:n) = A(i, k:n) - m * A(k, k:n);
    end
  end
  for i = 1:n
    d = d * A(i, i);
  end
end
```

Octave/MATLAB rank_gauss.m

```
function r = rank_gauss(A, tol)
% RANK_GAUSS  Rank = number of pivots larger than tol * max|a_ij|.
  if nargin < 2, tol = 1e-12; end
  [m, n] = size(A);
  scale = max(abs(A(:)));
  r = 0;
  for j = 1:n
    if r == m, break; end
    [piv, p] = max(abs(A(r + 1:m, j)));
    p = p + r;
    if piv <= tol * scale
      continue;                            % no pivot in this column
    end
    r = r + 1;
    A([r p], :) = A([p r], :);
    for i = r + 1:m
      f = A(i, j) / A(r, j);
      A(i, j:n) = A(i, j:n) - f * A(r, j:n);
    end
  end
end
```

Octave/MATLAB gauss_jordan.m

```
function X = gauss_jordan(A, B)
% GAUSS_JORDAN  Solve A X = B by Gauss-Jordan elimination.
%   B may have several columns; gauss_jordan(A, eye(n)) is the inverse of A.
  n  = size(A, 1);
  nc = n + size(B, 2);
  M  = [A, B];
  for k = 1:n
    [~, p] = max(abs(M(k:n, k)));
    p = p + k - 1;
    if M(p, k) == 0
      error('gauss_jordan: singular matrix');
    end
    M([k p], :) = M([p k], :);
    M(k, k:nc) = M(k, k:nc) / M(k, k);     % scale the pivot row
    for i = 1:n                            % clear the whole column
```

```
      if i ~= k
        M(i, k:nc) = M(i, k:nc) - M(i, k) * M(k, k:nc);
      end
    end
  end
  X = M(:, n + 1:nc);
end
```

Octave/MATLAB jacobi.m

```
function [x, k, ok] = jacobi(A, b, x0, tol, kmax)
% JACOBI  Jacobi iteration; ok = true if ||x_k - x_{k-1}|| <= tol*||x_k||.
  n = numel(b);
  x = x0(:);
  ok = false;
  for k = 1:kmax
    xn = zeros(n, 1);
    for i = 1:n
      s = b(i);
      for j = 1:n
        if j ~= i
          s = s - A(i, j) * x(j);          % old values only
        end
      end
      xn(i) = s / A(i, i);
    end
    d = max(abs(xn - x));
    x = xn;
    if d <= tol * max(abs(x))
      ok = true;  return;
    end
  end
end
```

Octave/MATLAB gauss_seidel.m

```
function [x, k, ok] = gauss_seidel(A, b, x0, tol, kmax)
% GAUSS_SEIDEL  Gauss-Seidel iteration; same interface as jacobi.
  n = numel(b);
  x = x0(:);
  ok = false;
  for k = 1:kmax
    d = 0;
    for i = 1:n
      s = b(i);
      for j = 1:n
        if j ~= i
          s = s - A(i, j) * x(j);          % newest values available
        end
      end
      xi = s / A(i, i);
      d = max(d, abs(xi - x(i)));
      x(i) = xi;
    end
    if d <= tol * max(abs(x))
      ok = true;  return;
    end
```

```
  end
end
```

Octave/MATLAB power_method.m

```
function [lam, v, k] = power_method(A, x0, tol, kmax)
% POWER_METHOD  Dominant eigenvalue and eigenvector by the power method.
  v = x0(:) / sqrt(dotp(x0, x0));
  lam = 0;
  for k = 1:kmax
    w = matvec(A, v);
    new = dotp(v, w);                      % Rayleigh quotient
    v = w / sqrt(dotp(w, w));
    if abs(new - lam) <= tol * abs(new)
      lam = new;  return;
    end
    lam = new;
  end
end
```

Octave/MATLAB inverse_power.m

```
function [lam, v, k] = inverse_power(A, mu, x0, tol, kmax)
% INVERSE_POWER  Eigenvalue of A closest to the shift mu (inverse iteration).
  n = size(A, 1);
  S = A - mu * eye(n);
  v = x0(:) / sqrt(dotp(x0, x0));
  lam = mu;
  for k = 1:kmax
    w = gauss_solve(S, v);                 % (A - mu I) w = v
    new = mu + 1 / dotp(v, w);
    v = w / sqrt(dotp(w, w));
    if abs(new - lam) <= tol * abs(new)
      lam = new;  return;
    end
    lam = new;
  end
end
```

Οι βοηθητικές συναρτήσεις `dotp`, `matvec`, `residual` και `print_matrix` βρίσκονται στον φάκελο του κεφαλαίου. Οι δύο πρώτες υπολογίζουν το εσωτερικό γινόμενο και το γινόμενο πίνακα επί διάνυσμα με ρητούς βρόχους, ώστε να φαίνεται η σειρά των πράξεων του αλγορίθμου· σε κανονική χρήση γράφουμε απλώς `u'*v` και `A*x`, που είναι πολύ ταχύτερα.

2.3 Λυμένα παραδείγματα

Παράδειγμα 2.1 (Βασικές πράξεις πινάκων)**.** Δίνονται οι πίνακες

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}, \qquad B = \begin{pmatrix} 5 & 2 & 2 \\ 4 & 8 & 1 \\ 3 & 5 & 2 \end{pmatrix}, \qquad C = \begin{pmatrix} 1 & 4 \\ 5 & 8 \\ 6 & 3 \end{pmatrix}.$$

Να υπολογιστούν οι $A^{\top}$, $A + B$, AC, AB, BA, καθώς και η τάξη και η ορίζουσα των A και B.

Λύση. Ο ανάστροφος προκύπτει με εναλλαγή γραμμών και στηλών και το άθροισμα στοιχείο προς

στοιχείο:

$$A^\top = \begin{pmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{pmatrix}, \qquad A + B = \begin{pmatrix} 6 & 4 & 5 \\ 8 & 13 & 7 \\ 10 & 13 & 11 \end{pmatrix}.$$

Το στοιχείο (i, j) του γινομένου είναι το εσωτερικό γινόμενο της γραμμής i του πρώτου πίνακα με τη στήλη j του δεύτερου· π.χ. για το AC:

$$AC = \begin{pmatrix} 1\cdot1+2\cdot5+3\cdot6 & 1\cdot4+2\cdot8+3\cdot3 \\ 4\cdot1+5\cdot5+6\cdot6 & 4\cdot4+5\cdot8+6\cdot3 \\ 7\cdot1+8\cdot5+9\cdot6 & 7\cdot4+8\cdot8+9\cdot3 \end{pmatrix} = \begin{pmatrix} 29 & 29 \\ 65 & 74 \\ 101 & 119 \end{pmatrix}.$$

Το γινόμενο CA δεν ορίζεται (ο C έχει 2 στήλες, ο A έχει 3 γραμμές). Τα AB και BA ορίζονται και τα δύο αλλά διαφέρουν, όπως δείχνει η έξοδος του προγράμματος.

Για τον A ισχύει γραμμή$_3 = 2\,$γραμμή$_2 -$ γραμμή$_1$, άρα οι γραμμές του είναι γραμμικά εξαρτημένες: $\operatorname{rank}(A) = 2$ και $\det A = 0$. Το πρόγραμμα όμως τυπώνει $\det A = 6.66 \times 10^{-16}$: λόγω στρογγύλευσης ο τελευταίος οδηγός της απαλοιφής δεν είναι ακριβώς μηδέν. Γι' αυτό *δεν ελέγχουμε ποτέ αν ένας πίνακας είναι αντιστρέψιμος με τη συνθήκη* `det(A) == 0`· η `rank_gauss` θεωρεί μηδενικό κάθε οδηγό μικρότερο από μια ανοχή και δίνει σωστά τάξη 2. Για τον B: $\det B = 37$, $\operatorname{rank}(B) = 3$.

Octave/MATLAB `ex21_basics.m`

```
% Example 2.1: transpose, sum, product, rank and determinant
A = [1 2 3; 4 5 6; 7 8 9];
B = [5 2 2; 4 8 1; 3 5 2];
C = [1 4; 5 8; 6 3];

print_matrix('A^T', A', '%4.0f');          % built-in operators
print_matrix('A + B', A + B, '%4.0f');
print_matrix('A C', A * C, '%4.0f');
print_matrix('A B', A * B, '%4.0f');
print_matrix('B A', B * A, '%4.0f');
fprintf('rank(A) = %d,  det(A) = %.3e\n', rank_gauss(A), det_gauss(A));
fprintf('rank(B) = %d,  det(B) = %.4f\n', rank_gauss(B), det_gauss(B));
```

Έξοδος

```
A^T =
   1    4    7
   2    5    8
   3    6    9
A + B =
   6    4    5
   8   13    7
  10   13   11
A C =
  29   29
  65   74
 101  119
A B =
  22   33   10
  58   78   25
  94  123   40
B A =
  27   36   45
  43   56   69
  37   47   57
rank(A) = 2,  det(A) = 6.661e-16
```

```
rank(B) = 3,  det(B) = 37.0000
```

Παράδειγμα 2.2 (Απαλοιφή Gauss με το χέρι)**.** Να λυθεί με απαλοιφή Gauss και μερική οδήγηση το σύστημα

$$A = \begin{pmatrix} 2 & 2 & 2 \\ 3 & 4 & 5 \\ 4 & 6 & 7 \end{pmatrix}, \qquad \mathbf{b} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix},$$

και να υπολογιστεί η $\det A$.

Λύση. *Βήμα* $k = 1$. Το μεγαλύτερο κατ' απόλυτη τιμή στοιχείο της πρώτης στήλης είναι το 4 (γραμμή 3), οπότε εναλλάσσονται οι γραμμές 1 και 3. Με πολλαπλασιαστές $m_{21} = 3/4$ και $m_{31} = 2/4 = 1/2$:

$$\left(\begin{array}{ccc|c} 2 & 2 & 2 & 1 \\ 3 & 4 & 5 & 2 \\ 4 & 6 & 7 & 3 \end{array}\right) \to \left(\begin{array}{ccc|c} 4 & 6 & 7 & 3 \\ 3 & 4 & 5 & 2 \\ 2 & 2 & 2 & 1 \end{array}\right) \to \left(\begin{array}{ccc|c} 4 & 6 & 7 & 3 \\ 0 & -\frac{1}{2} & -\frac{1}{4} & -\frac{1}{4} \\ 0 & -1 & -\frac{3}{2} & -\frac{1}{2} \end{array}\right).$$

Βήμα $k = 2$. Στη δεύτερη στήλη, από τη διαγώνιο και κάτω, το μεγαλύτερο στοιχείο είναι το -1 (γραμμή 3)· εναλλάσσονται οι γραμμές 2 και 3 και $m_{32} = (-\frac{1}{2})/(-1) = \frac{1}{2}$:

$$\left(\begin{array}{ccc|c} 4 & 6 & 7 & 3 \\ 0 & -1 & -\frac{3}{2} & -\frac{1}{2} \\ 0 & -\frac{1}{2} & -\frac{1}{4} & -\frac{1}{4} \end{array}\right) \to \left(\begin{array}{ccc|c} 4 & 6 & 7 & 3 \\ 0 & -1 & -\frac{3}{2} & -\frac{1}{2} \\ 0 & 0 & \frac{1}{2} & 0 \end{array}\right).$$

Πίσω αντικατάσταση. $\frac{1}{2}x_3 = 0 \Rightarrow x_3 = 0$· $-x_2 - \frac{3}{2}x_3 = -\frac{1}{2} \Rightarrow x_2 = \frac{1}{2}$· $4x_1 + 6x_2 + 7x_3 = 3 \Rightarrow x_1 = 0$. Άρα $\mathbf{x} = (0,\ \frac{1}{2},\ 0)^{\top}$.

Ορίζουσα. Έγιναν $s = 2$ εναλλαγές και οι οδηγοί είναι $4, -1, \frac{1}{2}$: $\det A = (-1)^2 \cdot 4 \cdot (-1) \cdot \frac{1}{2} = -2$.

Παράδειγμα 2.3 (Gauss–Jordan και αντίστροφος)**.** Να λυθεί το ίδιο σύστημα με απαλοιφή Gauss–Jordan και να υπολογιστεί ο A^{-1}.

Λύση. Στην παραλλαγή που υλοποιείται εδώ, η γραμμή του οδηγού διαιρείται πρώτα με τον οδηγό και μετά μηδενίζεται ολόκληρη η στήλη.

$k = 1$*:* εναλλαγή γραμμών 1 και 3, διαίρεση με 4, απαλοιφή από τις γραμμές 2 και 3:

$$\left(\begin{array}{ccc|c} 1 & \frac{3}{2} & \frac{7}{4} & \frac{3}{4} \\ 3 & 4 & 5 & 2 \\ 2 & 2 & 2 & 1 \end{array}\right) \to \left(\begin{array}{ccc|c} 1 & \frac{3}{2} & \frac{7}{4} & \frac{3}{4} \\ 0 & -\frac{1}{2} & -\frac{1}{4} & -\frac{1}{4} \\ 0 & -1 & -\frac{3}{2} & -\frac{1}{2} \end{array}\right).$$

$k = 2$*:* εναλλαγή γραμμών 2 και 3, διαίρεση με -1, απαλοιφή από τις γραμμές 1 και 3:

$$\left(\begin{array}{ccc|c} 1 & \frac{3}{2} & \frac{7}{4} & \frac{3}{4} \\ 0 & 1 & \frac{3}{2} & \frac{1}{2} \\ 0 & -\frac{1}{2} & -\frac{1}{4} & -\frac{1}{4} \end{array}\right) \to \left(\begin{array}{ccc|c} 1 & 0 & -\frac{1}{2} & 0 \\ 0 & 1 & \frac{3}{2} & \frac{1}{2} \\ 0 & 0 & \frac{1}{2} & 0 \end{array}\right).$$

$k = 3$*:* διαίρεση με $\frac{1}{2}$ και απαλοιφή από τις γραμμές 1 και 2:

$$\left(\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & \frac{1}{2} \\ 0 & 0 & 1 & 0 \end{array}\right) \quad \Longrightarrow \quad \mathbf{x} = (0,\ \tfrac{1}{2},\ 0)^{\top}.$$

Για τον αντίστροφο εκτελούνται ακριβώς οι ίδιες πράξεις στον πίνακα $[A \mid I]$, με τρεις στήλες δεύτερων μελών αντί για μία. Το πρόγραμμα δίνει

$$A^{-1} = \begin{pmatrix} 1 & 1 & -1 \\ \frac{1}{2} & -3 & 2 \\ -1 & 2 & -1 \end{pmatrix},$$

και πράγματι $A^{-1}\mathbf{b} = (1 + 2 - 3,\ \frac{1}{2} - 6 + 6,\ -1 + 4 - 3)^{\top} = (0, \frac{1}{2}, 0)^{\top}$.

Octave/MATLAB `ex22_gauss.m`

```
% Examples 2.2–2.3: Gaussian elimination, Gauss–Jordan and the inverse
A = [2 2 2; 3 4 5; 4 6 7];
b = [1; 2; 3];

x = gauss_solve(A, b);
fprintf('Gauss       : x = %.4f %.4f %.4f   residual = %.1e\n', x, residual(A, x, b));
X = gauss_jordan(A, b);
fprintf('Gauss–Jordan: x = %.4f %.4f %.4f\n', X);
Ainv = gauss_jordan(A, eye(3));
print_matrix('inverse(A)', Ainv, '%7.3f');
P = zeros(3);
for j = 1:3
  P(:, j) = matvec(A, Ainv(:, j));
end
print_matrix('A * inverse(A)', P, '%7.3f');
```

Έξοδος

```
Gauss       : x = 0.0000 0.5000 0.0000   residual = 0.0e+00
Gauss–Jordan: x = 0.0000 0.5000 0.0000
inverse(A) =
  1.000   1.000  -1.000
  0.500  -3.000   2.000
 -1.000   2.000  -1.000
A * inverse(A) =
  1.000   0.000   0.000
  0.000   1.000   0.000
  0.000   0.000   1.000
```

Παράδειγμα 2.4 (Jacobi και Gauss–Seidel)**.** Να εφαρμοστούν οι μέθοδοι Jacobi και Gauss–Seidel στα δύο συστήματα των εργαστηριακών σημειώσεων:

$$(\Sigma 1)\quad A=\begin{pmatrix}1&2&-2\\1&1&1\\2&2&1\end{pmatrix},\quad \mathbf{b}=\begin{pmatrix}1\\1\\1\end{pmatrix},\quad \mathbf{x}^{(0)}=\begin{pmatrix}1\\1\\1\end{pmatrix};$$

$$(\Sigma 2)\quad A=\begin{pmatrix}3&2&0\\0&2&1\\1&0&2\end{pmatrix},\quad \mathbf{b}=\begin{pmatrix}5\\3\\3\end{pmatrix},\quad \mathbf{x}^{(0)}=\mathbf{0},$$

με ακρίβεια $m=3$ σημαντικών ψηφίων για το (Σ1) και $m=2$ για το (Σ2).

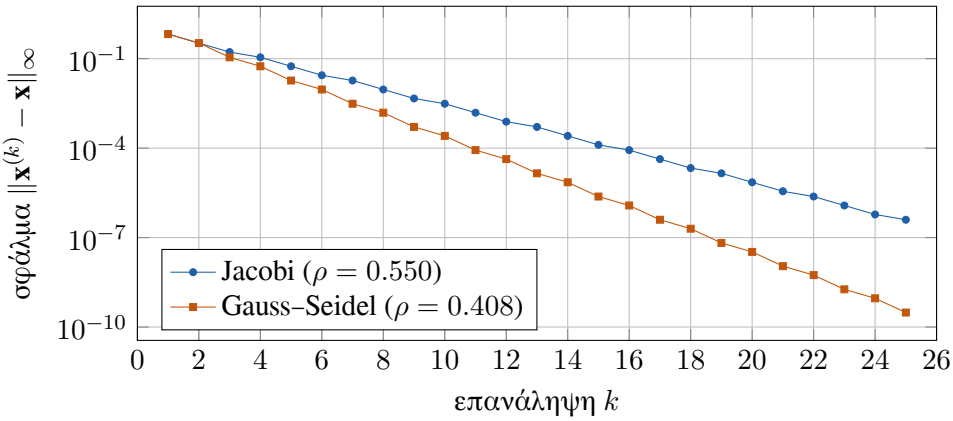
Λύση. *Σύστημα (Σ2).* Οι τύποι της μεθόδου Jacobi είναι

$$x_1^{(k+1)}=\tfrac{1}{3}\big(5-2x_2^{(k)}\big),\qquad x_2^{(k+1)}=\tfrac{1}{2}\big(3-x_3^{(k)}\big),\qquad x_3^{(k+1)}=\tfrac{1}{2}\big(3-x_1^{(k)}\big),$$

ενώ στη Gauss–Seidel η τρίτη εξίσωση χρησιμοποιεί το νέο $x_1^{(k+1)}$. Οι πρώτες επαναλήψεις είναι:

	Jacobi			Gauss–Seidel		
k	x_1	x_2	x_3	x_1	x_2	x_3
1	1.6667	1.5000	1.5000	1.6667	1.5000	0.6667
2	0.6667	0.7500	0.6667	0.6667	1.1667	1.1667
3	1.1667	1.1667	1.1667	0.8889	0.9167	1.0556
4	0.8889	0.9167	0.9167	1.0556	0.9722	0.9722

Και οι δύο ακολουθίες τείνουν στη λύση $\mathbf{x} = (1, 1, 1)^\top$. Με το κριτήριο τερματισμού για $m = 2$ (ανοχή 0.5×10^{-2}) η Jacobi σταματά μετά από 11 επαναλήψεις και η Gauss–Seidel μετά από 8. Ο πίνακας έχει αυστηρή διαγώνια κυριαρχία ($3 > 2, 2 > 1, 2 > 1$), άρα η σύγκλιση είναι εγγυημένη. Οι φασματικές ακτίνες των πινάκων επανάληψης είναι $\rho_J = 0.550$ και $\rho_{GS} = 0.408$: η Gauss–Seidel συγκλίνει ταχύτερα (Σχήμα 2.1).



Σχήμα 2.1: Σφάλμα των δύο μεθόδων για το σύστημα (Σ2). Η κλίση κάθε καμπύλης αντιστοιχεί στη φασματική ακτίνα του πίνακα επανάληψης.

Σύστημα (Σ1). Η Jacobi δίνει $\mathbf{x}^{(1)} = (1, -1, -3)^\top$, $\mathbf{x}^{(2)} = (-3, 3, 1)^\top$ και $\mathbf{x}^{(3)} = \mathbf{x}^{(2)}$: η ακριβής λύση $(-3, 3, 1)^\top$ βρίσκεται σε δύο μόλις βήματα και το κριτήριο τερματισμού ικανοποιείται στο τρίτο. Αντίθετα η Gauss–Seidel *αποκλίνει*: οι προσεγγίσεις διπλασιάζονται σε κάθε βήμα. Ο πίνακας δεν έχει διαγώνια κυριαρχία και καμία από τις ικανές συνθήκες δεν ισχύει· η συμπεριφορά εξηγείται από τις φασματικές ακτίνες, $\rho_J = 0$ και $\rho_{GS} = 2$ (Άσκηση 2.22). Το παράδειγμα δείχνει ότι η Gauss–Seidel δεν είναι *πάντα* καλύτερη από τη Jacobi.

Octave/MATLAB `report.m`

```
function report(name, x, k, ok)
% REPORT  Prints the result of an iterative method in one line.
  if ok, status = 'converged'; else, status = 'NOT converged'; end
  fprintf('%-13s: %-13s k = %3d  x = %.5g %.5g %.5g\n', name, status, k, x);
end
```

Octave/MATLAB `ex24_iterative.m`

```
% Example 2.4: Jacobi and Gauss-Seidel on the two systems of the lab notes
% uses the functions report.m

A1 = [1 2 -2; 1 1 1; 2 2 1];  b1 = [1; 1; 1];  x1 = [1; 1; 1];
fprintf('system 1 (tol = 0.5e-3)\n');
[x, k, ok] = jacobi(A1, b1, x1, 0.5e-3, 25);        report('Jacobi', x, k, ok);
[x, k, ok] = gauss_seidel(A1, b1, x1, 0.5e-3, 25);  report('Gauss-Seidel', x, k, ok);

A2 = [3 2 0; 0 2 1; 1 0 2];  b2 = [5; 3; 3];  x2 = [0; 0; 0];
fprintf('system 2 (tol = 0.5e-2)\n');
[x, k, ok] = jacobi(A2, b2, x2, 0.5e-2, 100);        report('Jacobi', x, k, ok);
[x, k, ok] = gauss_seidel(A2, b2, x2, 0.5e-2, 100);  report('Gauss-Seidel', x, k, ok);
fprintf('system 2 (tol = 1e-10)\n');
[x, k, ok] = jacobi(A2, b2, x2, 1e-10, 100);         report('Jacobi', x, k, ok);
[x, k, ok] = gauss_seidel(A2, b2, x2, 1e-10, 100);   report('Gauss-Seidel', x, k, ok);
```

Έξοδος

```
system 1 (tol = 0.5e-3)
Jacobi       : converged     k =   3  x = -3 3 1
Gauss-Seidel : NOT converged k =  25  x = 6.7109e+07 -6.7109e+07 1
system 2 (tol = 0.5e-2)
Jacobi       : converged     k =  11  x = 1.0015 1.0012 1.0015
Gauss-Seidel : converged     k =   8  x = 1.0015 0.99923 0.99923
system 2 (tol = 1e-10)
Jacobi       : converged     k =  41  x = 1 1 1
Gauss-Seidel : converged     k =  28  x = 1 1 1
```

Παράδειγμα 2.5 (Ανάμιξη κραμάτων)**.** Ένα χυτήριο διαθέτει τρία κράματα ανακύκλωσης με τις παρακάτω συστάσεις κατά βάρος και θέλει να παρασκευάσει 100 kg κράματος με 65 % Cu, 20 % Zn και 15 % Ni. Πόση μάζα πρέπει να χρησιμοποιηθεί από το καθένα;

	κράμα 1	κράμα 2	κράμα 3
Cu	60 %	80 %	50 %
Zn	30 %	10 %	20 %
Ni	10 %	10 %	30 %

Λύση. Αν m_1, m_2, m_3 οι ζητούμενες μάζες, το ισοζύγιο μάζας κάθε στοιχείου δίνει

$$\begin{pmatrix} 0.60 & 0.80 & 0.50 \\ 0.30 & 0.10 & 0.20 \\ 0.10 & 0.10 & 0.30 \end{pmatrix} \begin{pmatrix} m_1 \\ m_2 \\ m_3 \end{pmatrix} = \begin{pmatrix} 65 \\ 20 \\ 15 \end{pmatrix}.$$

Κάθε *στήλη* του πίνακα είναι η σύσταση ενός κράματος. Η απαλοιφή Gauss δίνει $m_1 = 37{,}5$ kg, $m_2 = 37{,}5$ kg, $m_3 = 25$ kg. Επειδή οι στήλες αθροίζουν στη μονάδα, η πρόσθεση των τριών εξισώσεων δίνει αυτόματα $m_1 + m_2 + m_3 = 100$, ένας χρήσιμος έλεγχος. Η λύση έχει φυσικό νόημα μόνο αν όλες οι μάζες είναι μη αρνητικές· με διαφορετικές επιθυμητές συστάσεις μπορεί να προκύψει αρνητική μάζα, που σημαίνει ότι η σύσταση δεν επιτυγχάνεται με τα διαθέσιμα κράματα.

Octave/MATLAB `ex25_alloy.m`

```
% Example 2.5: blending three scrap alloys to a target composition
%    alloy 1  alloy 2  alloy 3       (weight fractions)
A = [0.60     0.80     0.50          % Cu
     0.30     0.10     0.20          % Zn
     0.10     0.10     0.30];        % Ni
b = [65; 20; 15];                   % kg of Cu, Zn, Ni in 100 kg of product

m = gauss_solve(A, b);
fprintf('masses (kg): %.2f %.2f %.2f   total = %.2f\n', m, m(1) + m(2) + m(3));
fprintf('residual = %.1e\n', residual(A, m, b));
% check with the built-in solver:  A \ b
```

Έξοδος

```
masses (kg): 37.50 37.50 25.00   total = 100.00
residual = 3.6e-15
```

Παράδειγμα 2.6 (Κύριες τάσεις)**.** Η κατάσταση τάσης σε ένα σημείο ενός υλικού περιγράφεται από τον συμμετρικό τανυστή

$$\sigma = \begin{pmatrix} 50 & 30 & 20 \\ 30 & -20 & -10 \\ 20 & -10 & 10 \end{pmatrix} \text{ MPa}.$$

Να υπολογιστούν οι κύριες τάσεις και οι κύριες διευθύνσεις.

Λύση. Οι κύριες τάσεις είναι οι ιδιοτιμές του σ και οι κύριες διευθύνσεις τα ιδιοδιανύσματά του· επειδή ο τανυστής είναι συμμετρικός, οι ιδιοτιμές είναι πραγματικές και τα ιδιοδιανύσματα κάθετα μεταξύ τους. Η μέθοδος των δυνάμεων δίνει την ιδιοτιμή με τη μεγαλύτερη απόλυτη τιμή, $\sigma_1 = 65{,}53\,\text{MPa}$. Οι άλλες δύο βρίσκονται με αντίστροφη επανάληψη: με μετατόπιση $\mu = 0$ η πλησιέστερη στο μηδέν, $\sigma_2 = 11{,}53\,\text{MPa}$, και με $\mu = -30$ η $\sigma_3 = -37{,}06\,\text{MPa}$. Έλεγχος: το άθροισμα των ιδιοτιμών ισούται με το ίχνος του πίνακα, $65.53 + 11.53 - 37.06 = 40 = 50 - 20 + 10$.

Η αντίστροφη επανάληψη λύνει ένα γραμμικό σύστημα σε κάθε βήμα (με τη `gauss_solve`), συγκλίνει όμως πολύ γρήγορα όταν η μετατόπιση είναι κοντά στη ζητούμενη ιδιοτιμή: 8 επαναλήψεις για τη σ_3, έναντι 22 της μεθόδου των δυνάμεων για τη σ_1.

Octave/MATLAB `ex26_stress.m`

```
% Example 2.6: principal stresses = eigenvalues of the stress tensor
S  = [50 30 20; 30 -20 -10; 20 -10 10];              % MPa
x0 = [1; 1; 1];

[lam, v, k] = power_method(S, x0, 1e-10, 1000);
fprintf('power method        : %9.4f  (%d iterations)  v = %7.4f %7.4f %7.4f\n', lam, k,
    v);
for mu = [0 -30]
  [lam, v, k] = inverse_power(S, mu, x0, 1e-10, 1000);
  fprintf('inverse, shift %5.0f: %9.4f  (%d iterations)  v = %7.4f %7.4f %7.4f\n', mu,
    lam, k, v);
end
% check with the built-in function:  eig(S)
```

Έξοδος

```
power method        :   65.5269  (22 iterations)  v =  0.9161  0.2889  0.2780
inverse, shift     0:   11.5308  (13 iterations)  v = -0.1404 -0.4182  0.8974
inverse, shift   -30:  -37.0577  (8 iterations)  v = -0.3755  0.8612  0.3426
```

Παράδειγμα 2.7 (Μόνιμη αγωγή θερμότητας με πηγή)**.** Πλάκα πάχους $L = 0{,}1\,\text{m}$ με θερμική αγωγιμότητα $k = 50\,\text{W/(m K)}$ θερμαίνεται ομοιόμορφα στον όγκο της με ισχύ $q = 2 \times 10^6\,\text{W/m}^3$, ενώ οι δύο επιφάνειές της διατηρούνται στους $T(0) = 100\,^\circ\text{C}$ και $T(L) = 50\,^\circ\text{C}$. Η θερμοκρασία ικανοποιεί την εξίσωση $-k\,T''(x) = q$. Να υπολογιστεί η κατανομή της θερμοκρασίας με πεπερασμένες διαφορές και να μελετηθεί το πλήθος των επαναλήψεων των μεθόδων Jacobi και Gauss–Seidel ως συνάρτηση του πλήθους των κόμβων.

Λύση. Το διάστημα χωρίζεται σε $n+1$ ίσα τμήματα μήκους $h = L/(n+1)$ με εσωτερικούς κόμβους $x_i = ih$. Αντικαθιστώντας τη δεύτερη παράγωγο με την κεντρική διαφορά $T''(x_i) \approx (T_{i-1} - 2T_i + T_{i+1})/h^2$ (Κεφάλαιο 6) προκύπτει για κάθε κόμβο η εξίσωση

$$-T_{i-1} + 2T_i - T_{i+1} = \frac{h^2 q}{k}, \qquad i = 1, \dots, n,$$

με $T_0 = T(0)$ και $T_{n+1} = T(L)$ γνωστά. Το σύστημα είναι *τριδιαγώνιο*:

$$\begin{pmatrix} 2 & -1 & & \\ -1 & 2 & -1 & \\ & \ddots & \ddots & \ddots \\ & & -1 & 2 \end{pmatrix} \begin{pmatrix} T_1 \\ T_2 \\ \vdots \\ T_n \end{pmatrix} = \frac{h^2 q}{k} \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} + \begin{pmatrix} T(0) \\ 0 \\ \vdots \\ T(L) \end{pmatrix}.$$

Για $n = 4$ η απαλοιφή Gauss δίνει $T = (122,\ 128,\ 118,\ 92)\,^\circ\mathrm{C}$, που συμπίπτει με την ακριβή λύση $T(x) = T(0) + \big(T(L) - T(0)\big)\frac{x}{L} + \frac{q}{2k}x(L-x)$: επειδή η ακριβής λύση είναι πολυώνυμο δευτέρου βαθμού, η κεντρική διαφορά δεν έχει σφάλμα.

Οι επαναληπτικές μέθοδοι συγκλίνουν (ο πίνακας είναι συμμετρικός και θετικά ορισμένος), αλλά όλο και πιο αργά όσο πυκνώνει το πλέγμα:

n	Jacobi	Gauss–Seidel
4	60	32
9	228	117
19	813	414
39	2809	1435

Όταν το n διπλασιάζεται, οι επαναλήψεις σχεδόν τετραπλασιάζονται, και η Gauss–Seidel χρειάζεται σταθερά τις μισές. Η εξήγηση δίνεται στην Άσκηση 2.23: $\rho_J = \cos\frac{\pi}{n+1} \approx 1 - \frac{\pi^2}{2(n+1)^2}$ και $\rho_{GS} = \rho_J^2$. Για τριδιαγώνια συστήματα η κατάλληλη μέθοδος είναι ο αλγόριθμος Thomas (Άσκηση 2.13), με κόστος ανάλογο του n.

Octave/MATLAB `ex27_heat.m`

```
% Example 2.7: steady heat conduction with a source, -k T'' = q
L = 0.1;  kc = 50;  q = 2.0e6;       % m, W/(m K), W/m^3
T0 = 100;  TL = 50;                  % boundary temperatures, deg C

first = true;
for n = [4 4 9 19 39]
  h = L / (n + 1);                   % n interior nodes
  A = 2 * eye(n) - diag(ones(n - 1, 1), 1) - diag(ones(n - 1, 1), -1);
  b = h^2 * q / kc * ones(n, 1);
  b(1) = b(1) + T0;
  b(n) = b(n) + TL;
  if first                           % direct solution, n = 4
    T = gauss_solve(A, b);
    for i = 1:n
      x = i * h;
      fprintf('x = %.2f m: T = %8.3f  exact = %8.3f\n', x, T(i), ...
              T0 + (TL - T0) * x / L + q / (2 * kc) * x * (L - x));
    end
    fprintf('   n  Jacobi  Gauss-Seidel\n');
    first = false;
  else                               % iteration counts
    [~, kj] = jacobi(A, b, zeros(n, 1), 1e-6, 100000);
    [~, kg] = gauss_seidel(A, b, zeros(n, 1), 1e-6, 100000);
    fprintf('%4d  %6d  %6d\n', n, kj, kg);
  end
end
```

Έξοδος

```
x = 0.02 m: T =  122.000  exact =  122.000
x = 0.04 m: T =  128.000  exact =  128.000
x = 0.06 m: T =  118.000  exact =  118.000
x = 0.08 m: T =   92.000  exact =   92.000
   n  Jacobi  Gauss-Seidel
   4      60      32
   9     228     117
  19     813     414
  39    2809    1435
```

2.4 Ασκήσεις

Α. Βασικές ασκήσεις

Άσκηση 2.1 ★ (πράξεις πινάκων)**.** Για τους πίνακες A, B του Παραδείγματος 2.1 υπολογίστε με το χέρι τα AB και BA και επαληθεύστε ότι $(AB)^{\top} = B^{\top}A^{\top}$. Υπολογίστε επίσης το $A\mathbf{u}$ για $\mathbf{u} = (1, -2, 1)^{\top}$. Τι συμπεραίνετε για τον A;

Άσκηση 2.2 ★ (τάξη και ορίζουσα)**.** Υπολογίστε με απαλοιφή την τάξη και την ορίζουσα των πινάκων

$$\begin{pmatrix} 1 & 0 & 3 \\ 0 & 1 & 2 \\ 0 & 0 & 0 \end{pmatrix}, \qquad \begin{pmatrix} 2 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 2 \end{pmatrix}, \qquad \begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 3 & 6 & 4 \end{pmatrix}.$$

Επαληθεύστε με τις `rank_gauss` και `det_gauss`.

Άσκηση 2.3 ★ (ιδιοτιμές πίναχα 2×2)**.** Υπολογίστε από το χαρακτηριστικό πολυώνυμο τις ιδιοτιμές και τα ιδιοδιανύσματα των πινάκων

$$\begin{pmatrix} 3 & 0 \\ 8 & -1 \end{pmatrix}, \qquad \begin{pmatrix} 3 & 2 \\ -1 & 0 \end{pmatrix}, \qquad \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}.$$

Για τον τρίτο πίνακα επαληθεύστε ότι τα ιδιοδιανύσματα είναι κάθετα.

Άσκηση 2.4 ★ (αντίστροφος)**.** Υπολογίστε με Gauss–Jordan τον αντίστροφο των $\begin{pmatrix} 4 & 7 \\ 2 & 6 \end{pmatrix}$ και $\begin{pmatrix} 2 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 2 \end{pmatrix}$ και επαληθεύστε ότι $AA^{-1} = I$. Ποιος είναι ο αντίστροφος του διαγώνιου πίνακα $\operatorname{diag}(2, 3, 4)$;

Άσκηση 2.5 ★ (απαλοιφή Gauss με το χέρι)**.** Λύστε με απαλοιφή Gauss και μερική οδήγηση το σύστημα

$$\begin{aligned} 2x_1 + x_2 - x_3 &= 8, \\ -3x_1 - x_2 + 2x_3 &= -11, \\ -2x_1 + x_2 + 2x_3 &= -3, \end{aligned}$$

γράφοντας τον επαυξημένο πίνακα μετά από κάθε βήμα. Υπολογίστε και την ορίζουσα.

Άσκηση 2.6 ★ (Gauss–Jordan με το χέρι)**.** Λύστε το σύστημα της Άσκησης 2.5 με τη μέθοδο Gauss–Jordan. Μετρήστε τους πολλαπλασιασμούς και τις διαιρέσεις που εκτελέσατε σε κάθε μέθοδο.

Άσκηση 2.7 ★ (Jacobi και Gauss–Seidel με το χέρι)**.** Για το σύστημα

$$\begin{aligned} 4x_1 - x_2 + x_3 &= 7, \\ 4x_1 - 8x_2 + x_3 &= -21, \\ -2x_1 + x_2 + 5x_3 &= 15, \end{aligned}$$

(α) ελέγξτε αν ο πίνακας έχει αυστηρή διαγώνια κυριαρχία· (β) εκτελέστε με το χέρι τρεις επαναλήψεις Jacobi και τρεις Gauss–Seidel από $\mathbf{x}^{(0)} = \mathbf{0}$· (γ) βρείτε με το πρόγραμμα πόσες επαναλήψεις χρειάζεται κάθε μέθοδος για ανοχή 10^{-6}.

Άσκηση 2.8 ★ (καμία ή άπειρες λύσεις)**.** Για τον πίνακα A του Παραδείγματος 2.1 εξετάστε με απαλοιφή το σύστημα $A\mathbf{x} = \mathbf{b}$ όταν (α) $\mathbf{b} = (2, 4, 5)^{\top}$ και (β) $\mathbf{b} = (2, 4, 6)^{\top}$. Διατυπώστε το κριτήριο ύπαρξης λύσης με τη βοήθεια της τάξης των A και $[A \,|\, \mathbf{b}]$ και βρείτε όλες τις λύσεις στην περίπτωση (β).

Άσκηση 2.9 ★ (μηδενικός οδηγός)**.** Λύστε με το χέρι το σύστημα με $A = \begin{pmatrix} 0 & 2 & 1 \\ 1 & 1 & 1 \\ 2 & 1 & 3 \end{pmatrix}$ και $\mathbf{b} = (7, 6, 13)^{\top}$. Γιατί η απαλοιφή *χωρίς* εναλλαγή γραμμών αποτυγχάνει στο πρώτο βήμα, αν και ο πίνακας είναι αντιστρέψιμος;

Β. Προγραμματιστικές ασκήσεις

Άσκηση 2.10 ★★ (κόστος της απαλοιφής)**.** Προσθέστε στη `gauss_solve` έναν μετρητή πολλαπλασιασμών και διαιρέσεων. Καταγράψτε το πλήθος για $n = 5, 10, 20, 40, 80$ και δείξτε ότι ο λόγος του προς το $n^3/3$ τείνει στο 1. Μετρήστε επίσης τον χρόνο εκτέλεσης για $n = 100, 200, 400, 800$ με τις `tic`/`toc` και συγκρίνετε με τον ενσωματωμένο επιλυτή `A \ b`. Πόσο αυξάνει ο χρόνος όταν το n διπλασιάζεται;

Άσκηση 2.11 ★★ (γιατί χρειάζεται η οδήγηση)**.** Το σύστημα $\varepsilon x_1 + x_2 = 1$, $x_1 + x_2 = 2$ έχει για μικρό ε λύση $x_1 \approx x_2 \approx 1$. Λύστε το με απαλοιφή *χωρίς* εναλλαγή γραμμών και με μερική οδήγηση, για $\varepsilon = 10^{-4}, 10^{-12}, 10^{-16}, 10^{-20}$. Εξηγήστε τα αποτελέσματα.

Άσκηση 2.12 ★★ (παραγοντοποίηση LU)**.** Αν αποθηκευτούν οι πολλαπλασιαστές m_{ik} της απαλοιφής στη θέση των στοιχείων που μηδενίζονται, προκύπτει η παραγοντοποίηση $PA = LU$, όπου L κάτω τριγωνικός με μονάδες στη διαγώνιο, U άνω τριγωνικός και P ο πίνακας των εναλλαγών. Τότε το $A\mathbf{x} = \mathbf{b}$ λύνεται με δύο τριγωνικά συστήματα: $L\mathbf{y} = P\mathbf{b}$ και $U\mathbf{x} = \mathbf{y}$.

(α′) Υλοποιήστε τις `lu_factor` και `lu_solve`.

(β′) Παραγοντοποιήστε τον πίνακα του Παραδείγματος 2.2 και λύστε για $\mathbf{b} = (1, 2, 3)^{\top}$, $(6, 12, 17)^{\top}$ και $(2, 3, 4)^{\top}$.

(γ′) Υπολογίστε την ορίζουσα από τους παράγοντες.

(δ′) Πόσο κοστίζει η επίλυση για m διαφορετικά δεύτερα μέλη με LU και πόσο με m ανεξάρτητες απαλοιφές;

Άσκηση 2.13 ★★ (αλγόριθμος Thomas)**.** Για τριδιαγώνιο σύστημα $a_i x_{i-1} + b_i x_i + c_i x_{i+1} = d_i$ η απαλοιφή Gauss απλοποιείται στον αλγόριθμο Thomas:

$$c'_1 = \frac{c_1}{b_1}, \quad d'_1 = \frac{d_1}{b_1}, \qquad c'_i = \frac{c_i}{b_i - a_i c'_{i-1}}, \quad d'_i = \frac{d_i - a_i d'_{i-1}}{b_i - a_i c'_{i-1}}, \qquad x_n = d'_n, \quad x_i = d'_i - c'_i x_{i+1}.$$

Υλοποιήστε τον, αποθηκεύοντας μόνο τις τρεις διαγωνίους, και λύστε το πρόβλημα του Παραδείγματος 2.7 για $n = 1000$. Πόσες πράξεις απαιτούνται; Πόση μνήμη θα χρειαζόταν η `gauss_solve` για $n = 10^6$;

Άσκηση 2.14 ★★ (αντίστροφος ή επίλυση;)**.** Λύστε το σύστημα $H\mathbf{x} = \mathbf{b}$ με τον πίνακα Hilbert $h_{ij} = 1/(i + j - 1)$ τάξης $n = 8$ και $\mathbf{b}$ τέτοιο ώστε η ακριβής λύση να είναι $\mathbf{x} = (1, \dots, 1)^{\top}$, (α) με τη `gauss_solve` και (β) ως $\mathbf{x} = H^{-1}\mathbf{b}$. Συγκρίνετε τα σφάλματα, τα υπόλοιπα $\|\mathbf{b} - H\mathbf{x}\|$ και το πλήθος των πράξεων.

Άσκηση 2.15 ★★ (μέθοδος SOR)**.** Η μέθοδος διαδοχικής υπερχαλάρωσης (SOR) τροποποιεί τη Gauss–Seidel με μια παράμετρο ω:

$$x_i^{(k+1)} = (1 - \omega)\, x_i^{(k)} + \frac{\omega}{a_{ii}} \Big(b_i - \sum_{j<i} a_{ij} x_j^{(k+1)} - \sum_{j>i} a_{ij} x_j^{(k)} \Big).$$

Υλοποιήστε τη και εφαρμόστε τη στον τριδιαγώνιο πίνακα του Παραδείγματος 2.7 με $n = 19$, $\mathbf{b} = (1, \dots, 1)^{\top}$ και ανοχή 10^{-8}, για $\omega = 1.0, 1.2, \dots, 1.9$. Σχεδιάστε το πλήθος των επαναλήψεων ως προς ω και συγκρίνετε τη βέλτιστη τιμή με τη θεωρητική $\omega_{\text{opt}} = 2/\big(1 + \sin \tfrac{\pi}{n+1}\big)$.

Άσκηση 2.16 ★★ (κριτήρια τερματισμού)**.** Τροποποιήστε τη `jacobi` ώστε να τερματίζει (α) με τη σχετική μεταβολή, όπως τώρα, ή (β) με το σχετικό υπόλοιπο $\|\mathbf{b}-A\mathbf{x}^{(k)}\|/\|\mathbf{b}\| \le \varepsilon$. Για το πρόβλημα του Παραδείγματος 2.7 με $n=39$ και $\varepsilon=10^{-6}$ συγκρίνετε το *πραγματικό* σφάλμα κατά τον τερματισμό στις δύο περιπτώσεις. Ποιο κριτήριο είναι πιο αξιόπιστο;

Άσκηση 2.17 ★★ (διανυσματική μορφή)**.** Γράψτε τη μέθοδο Jacobi σε μορφή πινάκων, χωρίς βρόχο στις συνιστώσες, `x = (b - R*x) ./ d`, όπου `d` η διαγώνιος του A και `R = A - diag(d)`. Συγκρίνετε τον χρόνο εκτέλεσης με την εκδοχή με βρόχους για $n=200$. Γιατί η Gauss–Seidel δεν γράφεται εξίσου απλά;

Άσκηση 2.18 ★★ (παραγοντοποίηση Cholesky)**.** Κάθε συμμετρικός και θετικά ορισμένος πίνακας γράφεται $A=LL^{\top}$ με L κάτω τριγωνικό:

$$l_{jj}=\Big(a_{jj}-\sum_{k<j}l_{jk}^2\Big)^{1/2},\qquad l_{ij}=\frac{1}{l_{jj}}\Big(a_{ij}-\sum_{k<j}l_{ik}l_{jk}\Big),\quad i>j.$$

Υλοποιήστε τη μέθοδο, εφαρμόστε τη στον $\begin{pmatrix}4&2&2\\2&5&3\\2&3&6\end{pmatrix}$ και χρησιμοποιήστε τη για να ελέγξετε αν ένας συμμετρικός πίνακας είναι θετικά ορισμένος. Πόσο φθηνότερη είναι από την LU;

Άσκηση 2.19 ★★ (δείκτης κατάστασης: πίνακες Hilbert)**.** Για τους πίνακες Hilbert τάξης $n=3,6,9,12$ υπολογίστε τον δείκτη κατάστασης $\kappa_\infty(H)=\|H\|_\infty\|H^{-1}\|_\infty$ και το σφάλμα της λύσης του $H\mathbf{x}=\mathbf{b}$, όταν η ακριβής λύση είναι $(1,\dots,1)^{\top}$. Επαληθεύστε τον κανόνα «χάνονται $\log_{10}\kappa$ ψηφία».

Γ. Προχωρημένα μαθηματικά προβλήματα

Άσκηση 2.20 ★★★ (ακριβές κόστος της απαλοιφής)**.** Δείξτε ότι η απαλοιφή Gauss για το $A\mathbf{x}=\mathbf{b}$ απαιτεί ακριβώς $\tfrac{1}{6}n(n-1)(2n+5)$ πολλαπλασιασμούς και διαιρέσεις για την απαλοιφή και $\tfrac{1}{2}n(n+1)$ για την πίσω αντικατάσταση, συνολικά $\tfrac{1}{3}(n^3+3n^2-n)$. Συγκρίνετε με τη μέθοδο Gauss–Jordan και με τον κανόνα του Cramer.

Άσκηση 2.21 ★★★ (διαγώνια κυριαρχία και σύγκλιση)**.** Αποδείξτε ότι αν ο A έχει αυστηρή διαγώνια κυριαρχία κατά γραμμές, τότε (α) η μέθοδος Jacobi συγκλίνει, με $\|\mathbf{e}^{(k+1)}\|_\infty \le q\,\|\mathbf{e}^{(k)}\|_\infty$ και $q=\max_i\sum_{j\neq i}|a_{ij}|/|a_{ii}|<1$· (β) η μέθοδος Gauss–Seidel συγκλίνει.

Άσκηση 2.22 ★★★ (ανάλυση του Παραδείγματος 2.4)**.** Για το σύστημα (Σ1): (α) υπολογίστε τον πίνακα επανάληψης $B_J=D^{-1}(L+U)$ της μεθόδου Jacobi και δείξτε ότι $B_J^3=0$· συμπεράνετε ότι η μέθοδος δίνει την ακριβή λύση σε το πολύ τρία βήματα, για *κάθε* αρχική εκτίμηση. (β) Υπολογίστε τον $B_{GS}=(D-L)^{-1}U$ και τις ιδιοτιμές του και εξηγήστε την απόκλιση. (γ) Δείξτε ότι για τον πίνακα $\begin{pmatrix}2&-1&1\\2&2&2\\-1&-1&2\end{pmatrix}$ συμβαίνει το αντίθετο.

Άσκηση 2.23 ★★★ (ρυθμός σύγκλισης στο πρόβλημα αγωγής)**.** Για τον τριδιαγώνιο πίνακα A του Παραδείγματος 2.7:

(α′) Δείξτε ότι τα διανύσματα με συνιστώσες $v_i^{(j)}=\sin\frac{ij\pi}{n+1}$ είναι ιδιοδιανύσματα του B_J με ιδιοτιμές $\cos\frac{j\pi}{n+1}$, άρα $\rho_J=\cos\frac{\pi}{n+1}$.

(β′) Με δεδομένο ότι για τέτοιους πίνακες $\rho_{GS}=\rho_J^2$, εκτιμήστε το πλήθος των επαναλήψεων που μειώνουν το σφάλμα κατά 10^{-6} και συγκρίνετε με τον πίνακα του παραδείγματος.

(γ′) Εξηγήστε γιατί οι παρατηρούμενες επαναλήψεις είναι *λιγότερες* από τις προβλεπόμενες και εκτιμήστε το πραγματικό σφάλμα κατά τον τερματισμό.

Άσκηση 2.24 ★★★ (φράγμα σφάλματος με τον δείκτη κατάστασης). (α) Αποδείξτε την ανισότητα (2.2). (β) Για τον πίνακα $\begin{pmatrix} 1 & 1 \\ 1 & 1.0001 \end{pmatrix}$ λύστε το σύστημα με $\mathbf{b} = (2,\, 2.0001)^{\top}$ και με $\mathbf{b} = (2,\, 2.0002)^{\top}$. Υπολογίστε τον κ_∞ και επαληθεύστε το φράγμα. Δώστε γεωμετρική ερμηνεία.

Άσκηση 2.25 ★★★ (αύξηση των στοιχείων κατά την απαλοιφή). Έστω ο πίνακας $n \times n$ με 1 στη διαγώνιο και στην τελευταία στήλη, -1 κάτω από τη διαγώνιο και 0 αλλού. Δείξτε ότι η απαλοιφή με μερική οδήγηση δεν εκτελεί καμία εναλλαγή και ότι το τελευταίο στοιχείο του U είναι 2^{n-1}. Τι συνέπειες έχει αυτό για την ακρίβεια και γιατί η μερική οδήγηση θεωρείται παρ' όλα αυτά ασφαλής στην πράξη;

Άσκηση 2.26 ★★★ (θεώρημα Gershgorin). (α) Αποδείξτε ότι κάθε ιδιοτιμή του A βρίσκεται σε έναν τουλάχιστον από τους δίσκους $|z - a_{ii}| \le \sum_{j \neq i} |a_{ij}|$. (β) Εντοπίστε με το θεώρημα τις ιδιοτιμές του $\begin{pmatrix} 10 & 1 & 0.5 \\ 1 & -4 & 1 \\ 0.5 & 1 & 2 \end{pmatrix}$ και χρησιμοποιήστε τα κέντρα των δίσκων ως μετατοπίσεις στην αντίστροφη επανάληψη. (γ) Συμπεράνετε ότι κάθε πίνακας με αυστηρή διαγώνια κυριαρχία είναι αντιστρέψιμος.

Άσκηση 2.27 ★★★ (ρυθμός σύγκλισης της μεθόδου των δυνάμεων). Έστω A διαγωνοποιήσιμος με $|\lambda_1| > |\lambda_2| \ge \cdots \ge |\lambda_n|$. (α) Δείξτε ότι το σφάλμα της διεύθυνσης του $\mathbf{v}_k$ μειώνεται ως $|\lambda_2/\lambda_1|^{k}$. (β) Δείξτε ότι για συμμετρικό A το πηλίκο Rayleigh συγκλίνει ως $|\lambda_2/\lambda_1|^{2k}$. (γ) Επαληθεύστε με το Παράδειγμα 2.6: πόσες επαναλήψεις προβλέπονται για ανοχή 10^{-10};

Άσκηση 2.28 ★★★ (τύπος Sherman–Morrison). Δείξτε ότι αν ο A είναι αντιστρέψιμος και $1 + \mathbf{v}^{\top} A^{-1}\mathbf{u} \neq 0$, τότε

$$(A + \mathbf{u}\mathbf{v}^{\top})^{-1} = A^{-1} - \frac{A^{-1}\mathbf{u}\,\mathbf{v}^{\top}A^{-1}}{1 + \mathbf{v}^{\top}A^{-1}\mathbf{u}}.$$

Χρησιμοποιήστε τον τύπο για να λύσετε με δύο κλήσεις του αλγορίθμου Thomas το «κυκλικό» τριδιαγώνιο σύστημα μιας αλυσίδας ατόμων με περιοδικές συνοριακές συνθήκες (στοιχεία -1 και στις δύο γωνίες του πίνακα).

Άσκηση 2.29 ★★★ (ύπαρξη της παραγοντοποίησης LU). (α) Δείξτε ότι ο $\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$ δεν έχει παραγοντοποίηση $A = LU$, ενώ έχει $PA = LU$. (β) Δείξτε ότι αν όλοι οι κύριοι ελάσσονες $\det A_k$, $k = 1, \dots, n-1$, είναι μη μηδενικοί, η απαλοιφή χωρίς εναλλαγές ολοκληρώνεται και οι οδηγοί είναι $u_{kk} = \det A_k / \det A_{k-1}$. (γ) Συμπεράνετε ότι για πίνακες με αυστηρή διαγώνια κυριαρχία και για θετικά ορισμένους πίνακες δεν απαιτείται οδήγηση.

Άσκηση 2.30 ★★★ (επαναληπτική βελτίωση). Αν $\tilde{\mathbf{x}}$ είναι η υπολογισμένη λύση, το σφάλμα $\mathbf{e} = \mathbf{x} - \tilde{\mathbf{x}}$ ικανοποιεί $A\mathbf{e} = \mathbf{r}$ με $\mathbf{r} = \mathbf{b} - A\tilde{\mathbf{x}}$. Υλοποιήστε τη διαδικασία «υπολόγισε $\mathbf{r}$, λύσε $A\mathbf{e} = \mathbf{r}$ με τους ίδιους παράγοντες LU, διόρθωσε $\tilde{\mathbf{x}} \leftarrow \tilde{\mathbf{x}} + \mathbf{e}$» και δοκιμάστε τη στους πίνακες Hilbert της Άσκησης 2.19. Γιατί το υπόλοιπο πρέπει να υπολογίζεται με μεγαλύτερη ακρίβεια από τα υπόλοιπα βήματα;

2.5 Απαντήσεις επιλεγμένων ασκήσεων

2.1 $A\mathbf{u} = \mathbf{0}$: ο A δεν είναι αντιστρέψιμος.

2.2 Τάξεις $2,\ 3,\ 2\cdot$ ορίζουσες $0,\ 4,\ 0$.

2.3 $\{3, -1\},\ \{1, 2\},\ \{1, 3\}$.

2.4 $\dfrac{1}{10}\begin{pmatrix} 6 & -7 \\ -2 & 4 \end{pmatrix}$, $\dfrac{1}{4}\begin{pmatrix} 3 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 3 \end{pmatrix}$, $\operatorname{diag}(\tfrac{1}{2}, \tfrac{1}{3}, \tfrac{1}{4})$.

2.5 $\mathbf{x} = (2, 3, -1)^\top$, $\det A = -1$.

2.7 Λύση $(2, 4, 3)^\top$· 14 επαναλήψεις Jacobi, 8 Gauss–Seidel.

2.8 (α) καμία λύση· (β) $\mathbf{x} = (-\frac{2}{3} + t,\ \frac{4}{3} - 2t,\ t)^\top$.

2.9 $\mathbf{x} = (1, 2, 3)^\top$.

2.12 $\det A = -2$· λύσεις $(0, \frac{1}{2}, 0)$, $(1, 1, 1)$, $(1, 0, 0)$.

2.15 Ελάχιστο κοντά στο $\omega = 1.73$ (67 επαναλήψεις έναντι 597 για $\omega = 1$).

2.19 $\kappa_\infty = 7.5 \times 10^2$, 2.9×10^7, 1.1×10^{12}, 3.9×10^{16}.

2.24 $(1, 1)^\top$ και $(0, 2)^\top$· $\kappa_\infty = 4.0 \times 10^4$.

2.6 Πλήρεις λύσεις

Οι ασκήσεις της ομάδας Α λύνονται με το χέρι· τα αποτελέσματα επαληθεύονται με τις συναρτήσεις της Ενότητας 2.2. Για την ομάδα Β δίνεται το πρόγραμμα `sol_methods`. Για τις Ασκήσεις 2.14, 2.16, 2.17, 2.18, 2.25, 2.28 και 2.30 δίνονται υποδείξεις στο τέλος.

2.6.1 Ομάδα Α

Λύση της Άσκησης 2.1.

$$AB = \begin{pmatrix} 22 & 33 & 10 \\ 58 & 78 & 25 \\ 94 & 123 & 40 \end{pmatrix}, \qquad BA = \begin{pmatrix} 27 & 36 & 45 \\ 43 & 56 & 69 \\ 37 & 47 & 57 \end{pmatrix},$$

π.χ. $(AB)_{11} = 1 \cdot 5 + 2 \cdot 4 + 3 \cdot 3 = 22$. Προφανώς $AB \neq BA$. Ο $B^\top A^\top$ έχει στη θέση (i, j) το εσωτερικό γινόμενο της στήλης i του B με τη γραμμή j του A, δηλαδή το $(AB)_{ji}$· άρα $B^\top A^\top = (AB)^\top$. Τέλος $A\mathbf{u} = (1 - 4 + 3,\ 4 - 10 + 6,\ 7 - 16 + 9)^\top = \mathbf{0}$ με $\mathbf{u} \neq \mathbf{0}$: οι στήλες του A είναι γραμμικά εξαρτημένες (στήλη$_1$ $-$ 2 στήλη$_2$+στήλη$_3$ $= \mathbf{0}$), ο A δεν είναι αντιστρέψιμος και το 0 είναι ιδιοτιμή του με ιδιοδιάνυσμα $\mathbf{u}$.

Λύση της Άσκησης 2.2. Ο πρώτος πίνακας είναι ήδη σε κλιμακωτή μορφή με δύο μη μηδενικές γραμμές: τάξη 2, ορίζουσα 0 (πράγματι, στήλη$_3$ $= 3$ στήλη$_1 + 2$ στήλη$_2$). Για τον δεύτερο, η απαλοιφή χωρίς εναλλαγές δίνει

$$\begin{pmatrix} 2 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 2 \end{pmatrix} \to \begin{pmatrix} 2 & 1 & 0 \\ 0 & \frac{3}{2} & 1 \\ 0 & 1 & 2 \end{pmatrix} \to \begin{pmatrix} 2 & 1 & 0 \\ 0 & \frac{3}{2} & 1 \\ 0 & 0 & \frac{4}{3} \end{pmatrix},$$

άρα τάξη 3 και $\det = 2 \cdot \frac{3}{2} \cdot \frac{4}{3} = 4$. Στον τρίτο η δεύτερη γραμμή είναι διπλάσια της πρώτης· η απαλοιφή δίνει τις γραμμές $(1, 2, 1)$, $(0, 0, 0)$, $(0, 0, 1)$: τάξη 2, ορίζουσα 0.

Λύση της Άσκησης 2.3. (α) $\det(\lambda I - A) = (\lambda - 3)(\lambda + 1)$, άρα $\lambda_1 = 3$, $\lambda_2 = -1$. Για $\lambda = 3$: $8v_1 - 4v_2 = 0$, $\mathbf{v} = (1, 2)^\top$. Για $\lambda = -1$: $4v_1 = 0$, $\mathbf{v} = (0, 1)^\top$.

(β) $\det(\lambda I - A) = \lambda(\lambda - 3) + 2 = \lambda^2 - 3\lambda + 2$, άρα $\lambda = 1, 2$. Για $\lambda = 1$: $2v_1 + 2v_2 = 0$, $\mathbf{v} = (1, -1)^\top$. Για $\lambda = 2$: $v_1 + 2v_2 = 0$, $\mathbf{v} = (2, -1)^\top$.

(γ) $(\lambda - 2)^2 - 1 = 0$, άρα $\lambda = 1, 3$ με $\mathbf{v} = (1, -1)^\top$ και $(1, 1)^\top$, που είναι κάθετα ($1 \cdot 1 + (-1) \cdot 1 = 0$), όπως πάντα για συμμετρικό πίνακα με διαφορετικές ιδιοτιμές. Σε όλες τις περιπτώσεις το άθροισμα των ιδιοτιμών ισούται με το ίχνος και το γινόμενό τους με την ορίζουσα.

Λύση της Άσκησης 2.4. Για τον πίνακα 2×2:

$$\left(\begin{array}{cc|cc} 4 & 7 & 1 & 0 \\ 2 & 6 & 0 & 1 \end{array}\right) \to \left(\begin{array}{cc|cc} 1 & \frac{7}{4} & \frac{1}{4} & 0 \\ 0 & \frac{5}{2} & -\frac{1}{2} & 1 \end{array}\right) \to \left(\begin{array}{cc|cc} 1 & 0 & \frac{6}{10} & -\frac{7}{10} \\ 0 & 1 & -\frac{2}{10} & \frac{4}{10} \end{array}\right),$$

σε συμφωνία με τον τύπο $\begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} = \frac{1}{ad-bc}\begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$ με $ad - bc = 10$. Για τον 3×3 η ίδια διαδικασία δίνει

$$A^{-1} = \frac{1}{4}\begin{pmatrix} 3 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 3 \end{pmatrix}.$$

Παρατηρήστε ότι ο αντίστροφος ενός τριδιαγώνιου πίνακα είναι *πλήρης*: ένας ακόμη λόγος για τον οποίο οι αντίστροφοι αποφεύγονται. Ο αντίστροφος διαγώνιου πίνακα έχει στη διαγώνιο τα αντίστροφα στοιχεία: $\mathrm{diag}(\frac{1}{2}, \frac{1}{3}, \frac{1}{4})$.

Λύση της Άσκησης 2.5. $k = 1$: μεγαλύτερο στοιχείο το -3, εναλλαγή των γραμμών 1 και 2· $m_{21} = -\frac{2}{3}$, $m_{31} = \frac{2}{3}$:

$$\left(\begin{array}{ccc|c} -3 & -1 & 2 & -11 \\ 2 & 1 & -1 & 8 \\ -2 & 1 & 2 & -3 \end{array}\right) \to \left(\begin{array}{ccc|c} -3 & -1 & 2 & -11 \\ 0 & \frac{1}{3} & \frac{1}{3} & \frac{2}{3} \\ 0 & \frac{5}{3} & \frac{2}{3} & \frac{13}{3} \end{array}\right).$$

$k = 2$: μεγαλύτερο το $\frac{5}{3}$, εναλλαγή των γραμμών 2 και 3· $m_{32} = \frac{1}{5}$:

$$\left(\begin{array}{ccc|c} -3 & -1 & 2 & -11 \\ 0 & \frac{5}{3} & \frac{2}{3} & \frac{13}{3} \\ 0 & 0 & \frac{1}{5} & -\frac{1}{5} \end{array}\right).$$

Πίσω αντικατάσταση: $x_3 = -1$· $\frac{5}{3}x_2 = \frac{13}{3} + \frac{2}{3} \Rightarrow x_2 = 3$· $-3x_1 = -11 + 3 + 2 \Rightarrow x_1 = 2$. Με δύο εναλλαγές, $\det A = (-3) \cdot \frac{5}{3} \cdot \frac{1}{5} = -1$.

Λύση της Άσκησης 2.6. Η Gauss–Jordan δίνει απευθείας $[I \,|\, \mathbf{x}]$ με $\mathbf{x} = (2, 3, -1)^\top$. Στο βήμα k η γραμμή του οδηγού διαιρείται ($n - k + 1$ διαιρέσεις, μαζί με το δεύτερο μέλος) και ενημερώνονται οι άλλες $n-1$ γραμμές ($n-k+1$ πολλαπλασιασμοί η καθεμία), συνολικά $\sum_{k=1}^{n} n(n-k+1) = \frac{1}{2}n^2(n+1)$ πράξεις: 18 για $n = 3$, έναντι $\frac{1}{3}(n^3 + 3n^2 - n) = 17$ της απαλοιφής Gauss με πίσω αντικατάσταση (Άσκηση 2.20). Για μικρό n η διαφορά είναι αμελητέα, για μεγάλο n όμως η Gauss–Jordan κοστίζει 50% περισσότερο ($n^3/2$ έναντι $n^3/3$).

Λύση της Άσκησης 2.7. (α) $|4| > 1 + 1$, $|-8| > 4 + 1$, $|5| > 2 + 1$: υπάρχει αυστηρή διαγώνια κυριαρχία, άρα και οι δύο μέθοδοι συγκλίνουν.

(β) Οι τύποι είναι $x_1 = (7 + x_2 - x_3)/4$, $x_2 = (21 + 4x_1 + x_3)/8$, $x_3 = (15 + 2x_1 - x_2)/5$.

	Jacobi			Gauss–Seidel		
k	x_1	x_2	x_3	x_1	x_2	x_3
1	1.7500	2.6250	3.0000	1.7500	3.5000	3.0000
2	1.6563	3.8750	3.1750	1.8750	3.9375	2.9625
3	1.9250	3.8500	2.8875	1.9938	3.9922	2.9991

Η ακριβής λύση είναι $(2, 4, 3)^\top$. Μετά από τρία βήματα η Gauss–Seidel έχει σφάλμα 0.008, η Jacobi 0.15.

(γ) Για ανοχή 10^{-6}: 14 επαναλήψεις Jacobi και 8 Gauss–Seidel.

Λύση της Άσκησης 2.8. Η απαλοιφή (χωρίς εναλλαγές, για ευκολία) δίνει

$$\left(\begin{array}{ccc|c} 1 & 2 & 3 & 2 \\ 4 & 5 & 6 & 4 \\ 7 & 8 & 9 & b_3 \end{array}\right) \to \left(\begin{array}{ccc|c} 1 & 2 & 3 & 2 \\ 0 & -3 & -6 & -4 \\ 0 & -6 & -12 & b_3 - 14 \end{array}\right) \to \left(\begin{array}{ccc|c} 1 & 2 & 3 & 2 \\ 0 & -3 & -6 & -4 \\ 0 & 0 & 0 & b_3 - 6 \end{array}\right).$$

(α) Για $b_3 = 5$ η τελευταία εξίσωση είναι $0 = -1$: το σύστημα είναι αδύνατο, $\operatorname{rank}[A\,|\,\mathbf{b}] = 3 > \operatorname{rank} A = 2$. (β) Για $b_3 = 6$ είναι $0 = 0$ και οι δύο τάξεις είναι 2: υπάρχουν άπειρες λύσεις. Με $x_3 = t$ ελεύθερο: $x_2 = \frac{4}{3} - 2t$, $x_1 = 2 - 2x_2 - 3t = -\frac{2}{3} + t$.

Κριτήριο: το $A\mathbf{x} = \mathbf{b}$ έχει λύση αν και μόνο αν $\operatorname{rank} A = \operatorname{rank}[A\,|\,\mathbf{b}]$· η λύση είναι μοναδική όταν η κοινή τάξη ισούται με το πλήθος n των αγνώστων, αλλιώς υπάρχουν $n - \operatorname{rank} A$ ελεύθερες παράμετροι. Σε αριθμητική κινητής υποδιαστολής η διάκριση γίνεται με ανοχή, όπως στη `rank_gauss`.

Λύση της Άσκησης 2.9. Το $a_{11} = 0$ δεν μπορεί να χρησιμοποιηθεί ως οδηγός (διαίρεση με το μηδέν), αν και $\det A = -3 \neq 0$. Με μερική οδήγηση η γραμμή 3 έρχεται πρώτη:

$$\left(\begin{array}{ccc|c} 2 & 1 & 3 & 13 \\ 1 & 1 & 1 & 6 \\ 0 & 2 & 1 & 7 \end{array}\right) \to \left(\begin{array}{ccc|c} 2 & 1 & 3 & 13 \\ 0 & \frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} \\ 0 & 2 & 1 & 7 \end{array}\right) \to \left(\begin{array}{ccc|c} 2 & 1 & 3 & 13 \\ 0 & 2 & 1 & 7 \\ 0 & 0 & -\frac{3}{4} & -\frac{9}{4} \end{array}\right),$$

όπου στο δεύτερο βήμα εναλλάχθηκαν οι γραμμές 2 και 3. Άρα $x_3 = 3$, $x_2 = 2$, $x_1 = 1$. Η ύπαρξη μη μηδενικού οδηγού σε κάθε βήμα είναι εγγυημένη για κάθε αντιστρέψιμο πίνακα μόνο όταν επιτρέπονται εναλλαγές γραμμών (Άσκηση 2.29).

2.6.2 Ομάδα Β

Οι συναρτήσεις `lu_factor`, `lu_solve`, `thomas`, `sor` και οι βοηθητικές συναρτήσεις για τον δείκτη κατάστασης δίνονται μετά τη λύση της Άσκησης 2.15, και το πρόγραμμα που τις καλεί μετά τη λύση της 2.19.

Λύση της Άσκησης 2.10. Στο βήμα k, για καθεμία από τις $n - k$ γραμμές γίνεται μία διαίρεση (πολλαπλασιαστής) και $n - k + 1$ πολλαπλασιασμοί (οι στήλες $k+1, \dots, n$ και το δεύτερο μέλος). Ο μετρητής δίνει:

n	πράξεις	$n^3/3$	λόγος
5	50	42	1.200
10	375	333	1.125
20	2850	2667	1.069
40	22100	21333	1.036
80	173800	170667	1.018

σε συμφωνία με τον ακριβή τύπο $\frac{1}{6}n(n-1)(2n+5)$ της Άσκησης 2.20. Επειδή το κόστος είναι ανάλογο του n^3, ο χρόνος *οκταπλασιάζεται* όταν το n διπλασιάζεται. Οι ενσωματωμένοι επιλυτές εκτελούν τον ίδιο αλγόριθμο (LAPACK), αλλά σε βελτιστοποιημένο μεταγλωττισμένο κώδικα· είναι εκατοντάδες φορές ταχύτεροι από τους βρόχους της `gauss_solve`, που διερμηνεύονται εντολή προς εντολή.

Λύση της Άσκησης 2.11. *Χωρίς οδήγηση* ο οδηγός είναι το ε και ο πολλαπλασιαστής $m = 1/\varepsilon$:

$$\left(1 - \frac{1}{\varepsilon}\right)x_2 = 2 - \frac{1}{\varepsilon}, \qquad x_1 = \frac{1 - x_2}{\varepsilon}.$$

Για $\varepsilon = 10^{-20}$ οι ποσότητες $1 - 10^{20}$ και $2 - 10^{20}$ στρογγυλεύονται και οι δύο στο -10^{20} (τα 1 και 2 είναι αμελητέα σε σχέση με το 10^{20}), άρα $x_2 = 1$ και $x_1 = (1 - 1)/\varepsilon = 0$: εντελώς λάθος. Τα αποτελέσματα του προγράμματος:

ε	χωρίς οδήγηση (x_1, x_2)	με μερική οδήγηση (x_1, x_2)
10^{-4}	(1.000100, 0.999900)	(1.000100, 0.999900)
10^{-12}	(0.999978, 1.000000)	(1.000000, 1.000000)
10^{-16}	(2.220446, 1.000000)	(1.000000, 1.000000)
10^{-20}	(0.000000, 1.000000)	(1.000000, 1.000000)

Το x_2 υπολογίζεται σωστά, αλλά το μικρό σφάλμα στρογγύλευσής του πολλαπλασιάζεται επί $1/\varepsilon$ στον υπολογισμό του x_1. *Με οδήγηση* οι εξισώσεις εναλλάσσονται, ο πολλαπλασιαστής είναι ε και κανένα σφάλμα δεν μεγεθύνεται. Το σύστημα είναι καλής κατάστασης ($\kappa \approx 4$)· το πρόβλημα ήταν αποκλειστικά του *αλγορίθμου*.

Λύση της Άσκησης 2.12. (β) Η παραγοντοποίηση του πίνακα του Παραδείγματος 2.2 δίνει, με τη σειρά γραμμών $(3,1,2)$ του αρχικού πίνακα,

$$PA = \begin{pmatrix} 4 & 6 & 7 \\ 2 & 2 & 2 \\ 3 & 4 & 5 \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ \frac{1}{2} & 1 & 0 \\ \frac{3}{4} & \frac{1}{2} & 1 \end{pmatrix}}_{L} \underbrace{\begin{pmatrix} 4 & 6 & 7 \\ 0 & -1 & -\frac{3}{2} \\ 0 & 0 & \frac{1}{2} \end{pmatrix}}_{U}.$$

Οι πολλαπλασιαστές $\frac{1}{2}, \frac{3}{4}, \frac{1}{2}$ είναι ακριβώς αυτοί του Παραδείγματος 2.2 και ο U ο άνω τριγωνικός πίνακας που προέκυψε εκεί. Οι τρεις λύσεις είναι $(0, \frac{1}{2}, 0)$, $(1,1,1)$ και $(1,0,0)$.

(γ) $\det A = \pm u_{11}u_{22}u_{33}$, με πρόσημο $+$ για άρτιο πλήθος εναλλαγών: $\det A = 4 \cdot (-1) \cdot \frac{1}{2} = -2$.

(δ) Η παραγοντοποίηση κοστίζει περίπου $n^3/3$ πράξεις και γίνεται *μία* φορά· κάθε επίλυση με `lu_solve` κοστίζει μόνο n^2. Για m δεύτερα μέλη: $n^3/3 + mn^2$ με LU, έναντι $m\,n^3/3$ με ανεξάρτητες απαλοιφές. Για $n = 1000$ και $m = 100$ η διαφορά είναι ένας παράγοντας 77. Αυτή είναι η συνηθέστερη περίπτωση στην πράξη: ο ίδιος πίνακας με πολλά δεύτερα μέλη (διαδοχικά χρονικά βήματα, αντίστροφη επανάληψη, επαναληπτική βελτίωση).

Λύση της Άσκησης 2.13. Ο αλγόριθμος Thomas είναι η απαλοιφή Gauss χωρίς οδήγηση, στην οποία κάθε βήμα τροποποιεί μόνο μία γραμμή και δύο στοιχεία. Απαιτεί περίπου $5n$ πολλαπλασιασμούς και διαιρέσεις (3 ανά βήμα προς τα εμπρός, 1 προς τα πίσω) και μνήμη για 4 διανύσματα. Για $n = 1000$ το πρόγραμμα δίνει μέγιστη θερμοκρασία 128,125 °C και μέγιστη απόκλιση από την ακριβή λύση 4.4×10^{-11} (μόνο σφάλμα στρογγύλευσης). Η `gauss_solve` θα χρειαζόταν για $n = 10^6$ πίνακα 10^{12} στοιχείων, δηλαδή 8 TB μνήμης, και 3×10^{17} πράξεις· ο Thomas χρειάζεται 32 MB και 5×10^6 πράξεις. Η απουσία οδήγησης είναι ασφαλής όταν ο πίνακας έχει διαγώνια κυριαρχία ή είναι θετικά ορισμένος (Άσκηση 2.29), όπως εδώ.

Λύση της Άσκησης 2.15.

ω	1.0	1.2	1.4	1.6	1.7	1.7295	1.8	1.9
επαναλήψεις	597	407	265	150	93	67	82	161

Για $\omega = 1$ η μέθοδος συμπίπτει με τη Gauss–Seidel. Το πλήθος των επαναλήψεων μειώνεται μέχρι τη θεωρητική βέλτιστη τιμή $\omega_{\text{opt}} = 2/(1+\sin\frac{\pi}{20}) = 1.7295$ και μετά αυξάνει ξανά απότομα· η καμπύλη είναι πολύ ευαίσθητη κοντά στο ελάχιστο, και είναι προτιμότερο να υπερεκτιμηθεί ελαφρά το ω παρά να υποεκτιμηθεί. Η φασματική ακτίνα πέφτει από $\rho_{GS} = 0.9755$ σε $\rho_{\text{SOR}} = \omega_{\text{opt}} - 1 = 0.7295$, πράγμα που εξηγεί τον παράγοντα 9 στο πλήθος των επαναλήψεων. Η μέθοδος συγκλίνει μόνο για $0 < \omega < 2$.

Octave/MATLAB `lu_factor.m`

```
function [LU, perm, sgn] = lu_factor(A)
% LU_FACTOR  PA = LU with partial pivoting.
%   L (unit diagonal) is stored below the diagonal of LU, U on and above it.
  n = size(A, 1);
  LU = A;  perm = 1:n;  sgn = 1;
  for k = 1:n - 1
    [~, p] = max(abs(LU(k:n, k)));
    p = p + k - 1;
    if LU(p, k) == 0
      error('lu_factor: singular matrix');
```

```
    end
    if p ~= k
      LU([k p], :) = LU([p k], :);
      perm([k p]) = perm([p k]);
      sgn = -sgn;
    end
    for i = k + 1:n
      LU(i, k) = LU(i, k) / LU(k, k);      % multiplier l_ik
      LU(i, k + 1:n) = LU(i, k + 1:n) - LU(i, k) * LU(k, k + 1:n);
    end
  end
end
```

Octave/MATLAB `lu_solve.m`

```
function x = lu_solve(LU, perm, b)
% LU_SOLVE  Solve A x = b with the factors from lu_factor (O(n^2) operations).
  n = size(LU, 1);
  y = zeros(n, 1);  x = zeros(n, 1);
  for i = 1:n                              % forward: L y = P b
    s = b(perm(i));
    for j = 1:i - 1
      s = s - LU(i, j) * y(j);
    end
    y(i) = s;
  end
  for i = n:-1:1                           % backward: U x = y
    s = y(i);
    for j = i + 1:n
      s = s - LU(i, j) * x(j);
    end
    x(i) = s / LU(i, i);
  end
end
```

Octave/MATLAB `thomas.m`

```
function x = thomas(a, b, c, d)
% THOMAS  Tridiagonal system: a = sub-, b = main, c = super-diagonal, d = rhs.
%   a(1) and c(n) are not used. O(n) operations, no pivoting.
  n = numel(b);
  cp = zeros(n, 1);  dp = zeros(n, 1);  x = zeros(n, 1);
  cp(1) = c(1) / b(1);
  dp(1) = d(1) / b(1);
  for i = 2:n
    den = b(i) - a(i) * cp(i - 1);
    if i < n
      cp(i) = c(i) / den;
    end
    dp(i) = (d(i) - a(i) * dp(i - 1)) / den;
  end
  x(n) = dp(n);
  for i = n - 1:-1:1
    x(i) = dp(i) - cp(i) * x(i + 1);
  end
end
```

Octave/MATLAB `sor.m`

```
function [x, k, ok] = sor(A, b, x0, omega, tol, kmax)
% SOR  Successive over-relaxation (omega = 1 gives Gauss-Seidel).
  n = numel(b);
  x = x0(:);
  ok = false;
  for k = 1:kmax
    d = 0;
    for i = 1:n
      s = b(i);
      for j = 1:n
        if j ~= i
          s = s - A(i, j) * x(j);
        end
      end
      xi = (1 - omega) * x(i) + omega * s / A(i, i);
      d = max(d, abs(xi - x(i)));
      x(i) = xi;
    end
    if d <= tol * max(abs(x))
      ok = true;  return;
    end
  end
end
```

Octave/MATLAB `cholesky_factor.m`

```
function L = cholesky_factor(A)
% CHOLESKY_FACTOR  A = L*L' for a symmetric positive definite matrix.
  n = size(A, 1);
  L = zeros(n);
  for j = 1:n
    s = A(j, j);
    for k = 1:j - 1
      s = s - L(j, k)^2;
    end
    if s <= 0
      error('cholesky_factor: matrix is not positive definite');
    end
    L(j, j) = sqrt(s);
    for i = j + 1:n
      s = A(i, j);
      for k = 1:j - 1
        s = s - L(i, k) * L(j, k);
      end
      L(i, j) = s / L(j, j);
    end
  end
end
```

Octave/MATLAB `norm_inf_mat.m`

```
function nrm = norm_inf_mat(A)
% NORM_INF_MAT  Matrix infinity norm = maximum absolute row sum.
  nrm = 0;
  for i = 1:size(A, 1)
    s = 0;
    for j = 1:size(A, 2)
      s = s + abs(A(i, j));
    end
    nrm = max(nrm, s);
  end
end
```

Octave/MATLAB `cond_inf.m`

```
function c = cond_inf(A)
% COND_INF  Condition number in the infinity norm (maximum absolute row sum).
  Ainv = gauss_jordan(A, eye(size(A, 1)));
  c = norm_inf_mat(A) * norm_inf_mat(Ainv);
end
```

Λύση της Άσκησης 2.19.

n	$\kappa_\infty(H)$	μέγιστο σφάλμα	$\kappa \cdot \epsilon_{\text{mach}}$
3	7.5×10^{2}	1.0×10^{-14}	1.7×10^{-13}
6	2.9×10^{7}	5.3×10^{-10}	6.5×10^{-9}
9	1.1×10^{12}	2.0×10^{-5}	2.4×10^{-4}
12	3.9×10^{16}	3.6×10^{-1}	8.6

Ο δείκτης κατάστασης αυξάνει περίπου κατά 3×10^4 για κάθε τρεις επιπλέον γραμμές και το σφάλμα τον ακολουθεί: μένει περίπου μία τάξη μεγέθους κάτω από το φράγμα $\kappa\,\epsilon_{\text{mach}}$. Για $n = 12$ ο κ ξεπερνά το $1/\epsilon_{\text{mach}}$ και η λύση δεν έχει ούτε ένα σωστό ψηφίο, αν και το υπόλοιπο $\mathbf{b} - H\tilde{\mathbf{x}}$ παραμένει της τάξης του 10^{-16}. Κανένας αλγόριθμος δεν μπορεί να κάνει κάτι καλύτερο σε διπλή ακρίβεια: το πρόβλημα είναι κακής κατάστασης από τη φύση του. (Για $n = 12$ ακόμη και η τιμή του κ που υπολογίζεται μέσω του αντιστρόφου είναι αναξιόπιστη.)

Octave/MATLAB `sol_methods.m`

```
% Solutions 2.10–2.13, 2.15 and 2.19 (group B)
fprintf('--- 2.10 operation count of the elimination\n');
for n = [5 10 20 40 80]
  count = 0;
  for k = 1:n - 1
    for i = k + 1:n
      count = count + 1;                % one division for the multiplier
      count = count + n - k + 1;        % row update (columns k+1..n and rhs)
    end
  end
  fprintf('n = %3d: %7d operations,  n^3/3 = %9.0f,  ratio = %.3f\n', n, count, n^3 / 3,
    3 * count / n^3);
end

fprintf('--- 2.11 pivoting\n');
for e = [1e-4 1e-12 1e-16 1e-20]
  % e*x1 + x2 = 1,  x1 + x2 = 2, eliminated WITHOUT row exchange
```

```
  m  = 1 / e;
  x2 = (2 - m * 1) / (1 - m * 1);
  x1 = (1 - x2) / e;
  xp = gauss_solve([e 1; 1 1], [1; 2]);
  fprintf('eps = %.0e: no pivoting x = (%.6f, %.6f),  pivoting x = (%.6f, %.6f)\n', e,
    x1, x2, xp);
end

fprintf('--- 2.12 LU factorization\n');
A = [2 2 2; 3 4 5; 4 6 7];
[LU, perm, sgn] = lu_factor(A);
for i = 1:3
  fprintf('%8.4f %8.4f %8.4f\n', LU(i, :));
end
fprintf('perm = [%d, %d, %d]  det = %.4f\n', perm, sgn * LU(1, 1) * LU(2, 2) * LU(3, 3));
B = [1 2 3; 6 12 17; 2 3 4];
for c = 1:3
  x = lu_solve(LU, perm, B(c, :));
  fprintf('b = [%d, %d, %d]  x = %.4f %.4f %.4f\n', B(c, :), x);
end

fprintf('--- 2.13 Thomas algorithm\n');
n = 1000;
h = 0.1 / (n + 1);
d = h^2 * 2.0e6 / 50 * ones(n, 1);
d(1) = d(1) + 100;
d(n) = d(n) + 50;
T = thomas(-ones(n, 1), 2 * ones(n, 1), -ones(n, 1), d);
x = (1:n)' * h;
err = max(abs(T - (100 - 500 * x + 2.0e4 * x .* (0.1 - x))));
fprintf('n = %d: T_max = %.4f, max error = %.1e\n', n, max(T), err);

fprintf('--- 2.15 SOR on tridiag(-1, 2, -1), n = 19\n');
n = 19;
A = 2 * eye(n) - diag(ones(n - 1, 1), 1) - diag(ones(n - 1, 1), -1);
b = ones(n, 1);
for omega = [1.0 1.2 1.4 1.6 1.7 1.8 1.9]
  [~, k] = sor(A, b, zeros(n, 1), omega, 1e-8, 10000);
  fprintf('omega = %.2f: %4d iterations\n', omega, k);
end
wopt = 2 / (1 + sin(pi / (n + 1)));
[~, k] = sor(A, b, zeros(n, 1), wopt, 1e-8, 10000);
fprintf('omega_opt = %.4f: %4d iterations\n', wopt, k);

fprintf('--- 2.19 Hilbert matrices\n');
for n = [3 6 9 12]
  H = zeros(n);
  for i = 1:n
    for j = 1:n
      H(i, j) = 1 / (i + j - 1);
    end
  end
  b = zeros(n, 1);                      % exact solution: all ones
  for i = 1:n
    for j = 1:n
      b(i) = b(i) + H(i, j);
    end
  end
  x = gauss_solve(H, b);
```

```
  fprintf('n = %2d: cond = %.2e, max error = %.1e\n', n, cond_inf(H), max(abs(x - 1)));
end
```

Έξοδος

```
--- 2.10 operation count of the elimination
n =   5:      50 operations,  n^3/3 =        42,  ratio = 1.200
n =  10:     375 operations,  n^3/3 =       333,  ratio = 1.125
n =  20:    2850 operations,  n^3/3 =      2667,  ratio = 1.069
n =  40:   22100 operations,  n^3/3 =     21333,  ratio = 1.036
n =  80:  173800 operations,  n^3/3 =    170667,  ratio = 1.018
--- 2.11 pivoting
eps = 1e-04: no pivoting x = (1.000100, 0.999900),  pivoting x = (1.000100, 0.999900)
eps = 1e-12: no pivoting x = (0.999978, 1.000000),  pivoting x = (1.000000, 1.000000)
eps = 1e-16: no pivoting x = (2.220446, 1.000000),  pivoting x = (1.000000, 1.000000)
eps = 1e-20: no pivoting x = (0.000000, 1.000000),  pivoting x = (1.000000, 1.000000)
--- 2.12 LU factorization
  4.0000   6.0000   7.0000
  0.5000  -1.0000  -1.5000
  0.7500   0.5000   0.5000
perm = [3, 1, 2]  det = -2.0000
b = [1, 2, 3]  x = 0.0000 0.5000 0.0000
b = [6, 12, 17]  x = 1.0000 1.0000 1.0000
b = [2, 3, 4]  x = 1.0000 -0.0000 0.0000
--- 2.13 Thomas algorithm
n = 1000: T_max = 128.1250, max error = 4.4e-11
--- 2.15 SOR on tridiag(-1, 2, -1), n = 19
omega = 1.00:  597 iterations
omega = 1.20:  407 iterations
omega = 1.40:  265 iterations
omega = 1.60:  150 iterations
omega = 1.70:   93 iterations
omega = 1.80:   82 iterations
omega = 1.90:  161 iterations
omega_opt = 1.7295:   67 iterations
--- 2.19 Hilbert matrices
n =  3: cond = 7.48e+02, max error = 1.0e-14
n =  6: cond = 2.91e+07, max error = 5.3e-10
n =  9: cond = 1.10e+12, max error = 2.0e-05
n = 12: cond = 3.89e+16, max error = 3.6e-01
```

2.6.3 Ομάδα Γ

Λύση της Άσκησης 2.20. *Απαλοιφή.* Στο βήμα k τροποποιούνται $n-k$ γραμμές· για καθεμία γίνεται 1 διαίρεση (πολλαπλασιαστής), $n-k$ πολλαπλασιασμοί για τα στοιχεία των στηλών $k+1,\dots,n$ και 1 για το δεύτερο μέλος, δηλαδή $(n-k)(n-k+2)$ πράξεις. Με $j=n-k$:

$$\sum_{j=1}^{n-1} j(j+2) = \frac{(n-1)n(2n-1)}{6} + n(n-1) = \frac{n(n-1)(2n+5)}{6}.$$

Πίσω αντικατάσταση. Για τον x_i χρειάζονται $n-i$ πολλαπλασιασμοί και 1 διαίρεση: $\sum_{i=1}^{n}(n-i+1) = \frac{1}{2}n(n+1)$. *Σύνολο:*

$$\frac{n(n-1)(2n+5)}{6} + \frac{n(n+1)}{2} = \frac{n^3+3n^2-n}{3} \approx \frac{n^3}{3}.$$

Για $n = 3$ προκύπτουν 17 πράξεις και για $n = 10$ $375 + 55 = 430$, σε συμφωνία με τον μετρητή της Άσκησης 2.10. Η Gauss–Jordan απαιτεί $\frac{1}{2}n^2(n+1) \approx n^3/2$ (Άσκηση 2.6). Ο κανόνας του Cramer απαιτεί $n+1$ ορίζουσες· αν υπολογιστούν με ανάπτυγμα κατά γραμμή, καθεμία κοστίζει περίπου $n!$ πολλαπλασιασμούς, δηλαδή συνολικά $(n+1)!$: για $n = 20$ περίπου 5×10^{19} πράξεις, έναντι 3×10^3 της απαλοιφής.

Λύση της Άσκησης 2.21. (α) Το σφάλμα $\mathbf{e}^{(k)} = \mathbf{x}^{(k)} - \mathbf{x}$ ικανοποιεί $\mathbf{e}^{(k+1)} = B_J \mathbf{e}^{(k)}$ με $(B_J)_{ij} = -a_{ij}/a_{ii}$ για $j \neq i$ και μηδενική διαγώνιο. Άρα

$$|e_i^{(k+1)}| \le \sum_{j \neq i} \frac{|a_{ij}|}{|a_{ii}|} |e_j^{(k)}| \le q \, \|\mathbf{e}^{(k)}\|_\infty,$$

και παίρνοντας το μέγιστο ως προς i: $\|\mathbf{e}^{(k+1)}\|_\infty \le q \|\mathbf{e}^{(k)}\|_\infty$, οπότε $\|\mathbf{e}^{(k)}\|_\infty \le q^k \|\mathbf{e}^{(0)}\|_\infty \to 0$ αφού $q < 1$.

(β) Έστω λ ιδιοτιμή του $B_{GS} = (D-L)^{-1}U$ με ιδιοδιάνυσμα $\mathbf{v}$, κανονικοποιημένο ώστε $\|\mathbf{v}\|_\infty = |v_i| = 1$. Από $U\mathbf{v} = \lambda(D-L)\mathbf{v}$, η γραμμή i δίνει

$$-\sum_{j>i} a_{ij} v_j = \lambda \Big(a_{ii} v_i + \sum_{j<i} a_{ij} v_j \Big) \implies |\lambda| \le \frac{\sum_{j>i} |a_{ij}|}{|a_{ii}| - \sum_{j<i} |a_{ij}|} < 1,$$

όπου η τελευταία ανισότητα ισοδυναμεί με τη διαγώνια κυριαρχία της γραμμής i (ο παρονομαστής είναι θετικός για τον ίδιο λόγο). Άρα $\rho(B_{GS}) < 1$.

Λύση της Άσκησης 2.22. (α) Με $D = I$:

$$B_J = \begin{pmatrix} 0 & -2 & 2 \\ -1 & 0 & -1 \\ -2 & -2 & 0 \end{pmatrix}, \qquad B_J^2 = \begin{pmatrix} -2 & -4 & 2 \\ 2 & 4 & -2 \\ 2 & 4 & -2 \end{pmatrix}, \qquad B_J^3 = 0.$$

Το χαρακτηριστικό πολυώνυμο είναι λ^3, όλες οι ιδιοτιμές μηδενικές και $\rho_J = 0$. Επειδή $\mathbf{e}^{(k)} = B_J^k \mathbf{e}^{(0)}$, ισχύει $\mathbf{e}^{(3)} = \mathbf{0}$ για κάθε $\mathbf{x}^{(0)}$: η «επαναληπτική» μέθοδος γίνεται εδώ άμεση. (Με τη συγκεκριμένη αρχική εκτίμηση η λύση βρέθηκε ήδη στο δεύτερο βήμα, επειδή $B_J^2 \mathbf{e}^{(0)} = \mathbf{0}$ για $\mathbf{e}^{(0)} = (4, -2, 0)^\top$.)

(β)

$$B_{GS} = (D-L)^{-1}U = \begin{pmatrix} 0 & -2 & 2 \\ 0 & 2 & -3 \\ 0 & 0 & 2 \end{pmatrix},$$

άνω τριγωνικός με ιδιοτιμές $0, 2, 2$: $\rho_{GS} = 2 > 1$ και το σφάλμα περίπου διπλασιάζεται σε κάθε βήμα. Πράγματι, μετά από 25 επαναλήψεις οι προσεγγίσεις είναι της τάξης του $2^{26} = 6.7 \times 10^7$.

(γ) Για τον δεύτερο πίνακα οι ιδιοτιμές του B_J είναι $0, \pm i\sqrt{5}/2$, άρα $\rho_J = 1.118 > 1$ (απόκλιση), ενώ του B_{GS} είναι $0, -\frac{1}{2}, -\frac{1}{2}$, άρα $\rho_{GS} = \frac{1}{2}$ (σύγκλιση). Δεν υπάρχει γενική σχέση μεταξύ της σύγκλισης των δύο μεθόδων· η σχέση $\rho_{GS} = \rho_J^2$ ισχύει μόνο για ειδικές κατηγορίες πινάκων, όπως οι τριδιαγώνιοι.

Λύση της Άσκησης 2.23. (α) Εδώ $D = 2I$ και $B_J = I - \frac{1}{2}A$, δηλαδή $(B_J\mathbf{v})_i = \frac{1}{2}(v_{i-1} + v_{i+1})$ με $v_0 = v_{n+1} = 0$. Για $v_i = \sin(i\theta)$, $\theta = \frac{j\pi}{n+1}$, η ταυτότητα $\sin(i-1)\theta + \sin(i+1)\theta = 2\sin(i\theta)\cos\theta$ δίνει $(B_J\mathbf{v})_i = \cos\theta \, v_i$, και οι συνοριακές συνθήκες ικανοποιούνται αφού $\sin 0 = \sin(j\pi) = 0$. Οι n ιδιοτιμές είναι $\cos\frac{j\pi}{n+1}$, $j = 1, \dots, n$, και η μεγαλύτερη κατ' απόλυτη τιμή είναι $\rho_J = \cos\frac{\pi}{n+1} \approx 1 - \frac{\pi^2}{2(n+1)^2}$.

(β) Για μείωση του σφάλματος κατά ε χρειάζονται $k \approx \ln\varepsilon / \ln\rho$ επαναλήψεις, δηλαδή για τη Jacobi $k \approx \frac{2(n+1)^2}{\pi^2} \ln\frac{1}{\varepsilon}$ και για τη Gauss–Seidel οι μισές:

			πρόβλεψη ($\varepsilon = 10^{-6}$)		παρατήρηση	
n	ρ_J	ρ_{GS}	Jacobi	G–S	Jacobi	G–S
4	0.8090	0.6545	65	33	60	32
9	0.9511	0.9045	275	138	228	117
19	0.9877	0.9755	1115	558	813	414
39	0.9969	0.9938	4475	2237	2809	1435

Η εξάρτηση από το $(n+1)^2$ και ο παράγοντας 2 μεταξύ των δύο μεθόδων επιβεβαιώνονται.

(γ) Το πρόγραμμα σταματά όταν η *μεταβολή* γίνει μικρή, όχι το σφάλμα. Όταν η σύγκλιση είναι γεωμετρική με λόγο ρ,

$$\|\mathbf{x}^{(k)} - \mathbf{x}\| \approx \frac{\rho}{1-\rho}\,\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\|,$$

και για $n = 39$ ο παράγοντας $\rho_J/(1-\rho_J)$ είναι περίπου 320. Πράγματι, κατά τον τερματισμό με ανοχή 10^{-6} το πραγματικό σχετικό σφάλμα της Jacobi είναι 2.0×10^{-4}: η μέθοδος «δηλώνει» έξι σωστά ψηφία και έχει λιγότερα από τέσσερα. Όσο πιο αργή η σύγκλιση, τόσο πιο παραπλανητικό το κριτήριο.

Λύση της Άσκησης 2.24. (α) Από $A\mathbf{x} = \mathbf{b}$ και $A(\mathbf{x}+\delta\mathbf{x}) = \mathbf{b}+\delta\mathbf{b}$ προκύπτει $\delta\mathbf{x} = A^{-1}\delta\mathbf{b}$, άρα $\|\delta\mathbf{x}\| \le \|A^{-1}\|\,\|\delta\mathbf{b}\|$. Επίσης $\|\mathbf{b}\| = \|A\mathbf{x}\| \le \|A\|\,\|\mathbf{x}\|$, δηλαδή $1/\|\mathbf{x}\| \le \|A\|/\|\mathbf{b}\|$. Πολλαπλασιάζοντας:

$$\frac{\|\delta\mathbf{x}\|}{\|\mathbf{x}\|} \le \|A\|\,\|A^{-1}\|\,\frac{\|\delta\mathbf{b}\|}{\|\mathbf{b}\|}.$$

(β) Οι λύσεις είναι $(1,1)^{\top}$ και $(0,2)^{\top}$: μεταβολή 10^{-4} σε ένα στοιχείο του $\mathbf{b}$ άλλαξε τη λύση κατά μία μονάδα. Εδώ $\|A\|_\infty = 2.0001$ και $A^{-1} = 10^{4}\begin{pmatrix} 1.0001 & -1 \\ -1 & 1 \end{pmatrix}$ με $\|A^{-1}\|_\infty = 20001$, άρα $\kappa_\infty = 4.0 \times 10^{4}$. Το φράγμα δίνει $\|\delta\mathbf{x}\|/\|\mathbf{x}\| \le 4.0\times10^{4}\cdot(10^{-4}/2.0001) = 2.0$, και η πραγματική τιμή είναι 1.0. Γεωμετρικά, οι δύο εξισώσεις παριστάνουν σχεδόν παράλληλες ευθείες· μια ελάχιστη μετατόπιση της μίας μετακινεί το σημείο τομής τους πολύ μακριά.

Λύση της Άσκησης 2.26. (α) Έστω $A\mathbf{v} = \lambda\mathbf{v}$ και i ο δείκτης με $|v_i| = \|\mathbf{v}\|_\infty > 0$. Η γραμμή i γράφεται $(\lambda - a_{ii})v_i = \sum_{j\neq i} a_{ij}v_j$, οπότε

$$|\lambda - a_{ii}| \le \sum_{j\neq i} |a_{ij}|\,\frac{|v_j|}{|v_i|} \le \sum_{j\neq i} |a_{ij}|.$$

(β) Οι δίσκοι είναι: κέντρο 10, ακτίνα 1.5· κέντρο -4, ακτίνα 2· κέντρο 2, ακτίνα 1.5. Είναι ξένοι μεταξύ τους, άρα καθένας περιέχει ακριβώς μία ιδιοτιμή, και επειδή ο πίνακας είναι συμμετρικός οι ιδιοτιμές είναι πραγματικές: $\lambda_1 \in [8.5, 11.5]$, $\lambda_2 \in [0.5, 3.5]$, $\lambda_3 \in [-6, -2]$. Η αντίστροφη επανάληψη με μετατοπίσεις 10, 2 και -4 δίνει 10.1114, 2.1090 και -4.2204, σε λίγες επαναλήψεις η καθεμία.

(γ) Αν $|a_{ii}| > \sum_{j\neq i}|a_{ij}|$ για κάθε i, κανένας δίσκος δεν περιέχει το 0, άρα το 0 δεν είναι ιδιοτιμή και ο A είναι αντιστρέψιμος.

Λύση της Άσκησης 2.27. (α) Γράφουμε $\mathbf{v}_0 = \sum_i c_i\mathbf{u}_i$ στη βάση των ιδιοδιανυσμάτων, με $c_1 \neq 0$. Τότε

$$A^k\mathbf{v}_0 = \lambda_1^k\Big(c_1\mathbf{u}_1 + \sum_{i\ge 2} c_i\Big(\frac{\lambda_i}{\lambda_1}\Big)^k\mathbf{u}_i\Big),$$

και το $\mathbf{v}_k$ είναι το ίδιο διάνυσμα κανονικοποιημένο. Οι συνιστώσες εκτός της $\mathbf{u}_1$ μειώνονται τουλάχιστον ως $|\lambda_2/\lambda_1|^k$.

(β) Για συμμετρικό A τα $\mathbf{u}_i$ είναι ορθοκανονικά. Αν $\mathbf{v}_k = \mathbf{u}_1 + \boldsymbol{\epsilon}$ (πριν από την κανονικοποίηση), με $\boldsymbol{\epsilon} \perp \mathbf{u}_1$ και $\|\boldsymbol{\epsilon}\| = O(|\lambda_2/\lambda_1|^k)$, τότε

$$\frac{\mathbf{v}_k^{\top}A\mathbf{v}_k}{\mathbf{v}_k^{\top}\mathbf{v}_k} = \frac{\lambda_1 + \boldsymbol{\epsilon}^{\top}A\boldsymbol{\epsilon}}{1+\|\boldsymbol{\epsilon}\|^2} = \lambda_1 + O(\|\boldsymbol{\epsilon}\|^2),$$

αφού οι όροι πρώτης τάξης $\mathbf{u}_1^{\top} A\boldsymbol{\epsilon} = \lambda_1 \mathbf{u}_1^{\top}\boldsymbol{\epsilon}$ μηδενίζονται. Το σφάλμα της ιδιοτιμής είναι το *τετράγωνο* του σφάλματος του ιδιοδιανύσματος.

(γ) Στο Παράδειγμα 2.6 η δεύτερη κατ' απόλυτη τιμή ιδιοτιμή είναι η -37.06, άρα $|\lambda_2/\lambda_1| = 0.5655$ και ο λόγος για το πηλίκο Rayleigh είναι $0.5655^2 = 0.320$. Για ανοχή 10^{-10} προβλέπονται

$$k \approx \frac{\ln 10^{-10}}{\ln 0.320} \approx 20$$

επαναλήψεις, σε συμφωνία με τις 22 που παρατηρήθηκαν.

Λύση της Άσκησης 2.29. (α) Αν $\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ l & 1 \end{pmatrix}\begin{pmatrix} u_{11} & u_{12} \\ 0 & u_{22} \end{pmatrix}$, τότε $u_{11} = 0$ και $l\,u_{11} = 1$, άτοπο. Με εναλλαγή των γραμμών, $PA = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$, που είναι ήδη άνω τριγωνικός ($L = I$).

(β) Με επαγωγή ως προς k. Η απαλοιφή χωρίς εναλλαγές προσθέτει σε κάθε γραμμή πολλαπλάσια *προηγούμενων* γραμμών, πράξη που δεν αλλάζει την ορίζουσα κανενός κύριου υποπίνακα A_k (των πρώτων k γραμμών και στηλών). Μετά από $k-1$ βήματα ο A_k έχει γίνει άνω τριγωνικός με διαγώνια στοιχεία $u_{11}, \dots, u_{kk}$, άρα $\det A_k = u_{11}\cdots u_{kk}$ και

$$u_{kk} = \frac{\det A_k}{\det A_{k-1}} \qquad (\det A_0 = 1).$$

Αν $\det A_k \neq 0$ για $k \le n-1$, όλοι οι οδηγοί $u_{11}, \dots, u_{n-1,n-1}$ είναι μη μηδενικοί και η απαλοιφή ολοκληρώνεται.

(γ) Αν ο A έχει αυστηρή διαγώνια κυριαρχία, το ίδιο ισχύει για κάθε A_k, που επομένως είναι αντιστρέψιμος (Άσκηση 2.26γ). Αν ο A είναι συμμετρικός και θετικά ορισμένος, κάθε A_k είναι επίσης θετικά ορισμένος (αρκεί να θεωρηθούν διανύσματα με μηδενικές τις τελευταίες $n-k$ συνιστώσες), άρα $\det A_k > 0$. Και στις δύο περιπτώσεις η απαλοιφή χωρίς οδήγηση είναι δυνατή, και μάλιστα ευσταθής· σε αυτό βασίζονται ο αλγόριθμος Thomas και η παραγοντοποίηση Cholesky.

2.6.4 Υποδείξεις για τις υπόλοιπες ασκήσεις

2.14 Για $n = 8$ τα δύο σφάλματα είναι συγκρίσιμα (4×10^{-7} και 5×10^{-7}), το υπόλοιπο όμως είναι 2×10^{-16} με την απαλοιφή και 1×10^{-7} με τον αντίστροφο: η απαλοιφή δίνει πάντα μικρό υπόλοιπο, ο πολλαπλασιασμός με τον A^{-1} όχι. Ο αντίστροφος κοστίζει περίπου n^3 πράξεις, τριπλάσιες από την απαλοιφή.

2.16 Το υπόλοιπο είναι πιο αξιόπιστο από τη μεταβολή, αλλά και αυτό υποεκτιμά το σφάλμα κατά παράγοντα έως $\kappa(A)$· για τον τριδιαγώνιο πίνακα $\kappa \approx 4(n+1)^2/\pi^2$. Βλ. τη λύση της Άσκησης 2.23(γ).

2.17 Στη Gauss–Seidel κάθε συνιστώσα εξαρτάται από τις προηγούμενες του ίδιου βήματος, οπότε ο υπολογισμός είναι εκ φύσεως διαδοχικός· σε μορφή πινάκων απαιτεί επίλυση τριγωνικού συστήματος, `x = (D - L) \ (U*x + b)`.

2.18 $L = \begin{pmatrix} 2 & 0 & 0 \\ 1 & 2 & 0 \\ 1 & 1 & 2 \end{pmatrix}$. Αν σε κάποιο βήμα η υπόρριζη ποσότητα δεν είναι θετική, ο πίνακας δεν είναι θετικά ορισμένος. Το κόστος είναι $n^3/6$ πολλαπλασιασμοί, το μισό της LU.

2.25 Σε κάθε βήμα όλοι οι πολλαπλασιαστές είναι -1 και τα στοιχεία της τελευταίας στήλης διπλασιάζονται, οπότε γίνονται διαδοχικά $1, 2, 4, \dots, 2^{n-1}$. Σφάλματα στρογγύλευσης μπορούν να μεγεθυνθούν κατά τον ίδιο παράγοντα (10^{15} για $n = 50$). Τέτοιοι πίνακες είναι όμως εξαιρετικά σπάνιοι· στην πράξη ο παράγοντας αύξησης σπάνια ξεπερνά το 10.

2.28 Για την απόδειξη, πολλαπλασιάστε το δεξί μέλος του τύπου με τον $A + \mathbf{u}\mathbf{v}^\top$. Για το κυκλικό σύστημα επιλέξτε

$$\mathbf{u} = (\gamma, 0, \ldots, 0, -1)^\top, \qquad \mathbf{v} = (1, 0, \ldots, 0, -1/\gamma)^\top,$$

ώστε ο $A = A_{\text{κυκλ}} - \mathbf{u}\mathbf{v}^\top$ να είναι τριδιαγώνιος. Λύστε $A\mathbf{y} = \mathbf{b}$ και $A\mathbf{z} = \mathbf{u}$ με τον αλγόριθμο Thomas και θέστε

$$\mathbf{x} = \mathbf{y} - \frac{\mathbf{v}^\top \mathbf{y}}{1 + \mathbf{v}^\top \mathbf{z}}\, \mathbf{z}.$$

2.30 Το υπόλοιπο είναι διαφορά σχεδόν ίσων αριθμών ($\mathbf{b}$ και $A\tilde{\mathbf{x}}$) και, αν υπολογιστεί με την ίδια ακρίβεια, αποτελείται κυρίως από σφάλμα στρογγύλευσης (βλ. Ενότητα 1.8). Με υπόλοιπο σε μεγαλύτερη ακρίβεια (π.χ. με αθροίσεις τύπου Kahan, Άσκηση 1.29) κάθε βήμα κερδίζει περίπου $\log_{10}(1/(\kappa\, \epsilon_{\text{mach}}))$ ψηφία, εφόσον $\kappa\, \epsilon_{\text{mach}} < 1$.