Κεφάλαιο 1

Εισαγωγή στο Octave/MATLAB

Το κεφάλαιο αυτό συγκεντρώνει τα προγραμματιστικά εργαλεία που χρειάζονται στα επόμενα: πώς δουλεύουμε στο περιβάλλον του Octave/MATLAB, πώς γράφεται και εκτελείται ένα πρόγραμμα και μια συνάρτηση, τις βασικές πράξεις και τις δομές ελέγχου και επανάληψης. Ιδιαίτερη έμφαση δίνεται στους *πίνακες και τα διανύσματα*, που είναι ο βασικός τύπος δεδομένων της γλώσσας: δημιουργία, δείκτες, πράξεις, ανάστροφος, ορίζουσα, αντίστροφος, ιδιοτιμές και νόρμες. Το κεφάλαιο κλείνει με τις ιδιαιτερότητες της αριθμητικής κινητής υποδιαστολής, που καθορίζουν την ακρίβεια κάθε αριθμητικής μεθόδου.

1.1 Το περιβάλλον εργασίας Octave/MATLAB

Το **MATLAB** (MATrix LABoratory) είναι εμπορικό λογισμικό σχεδιασμένο για υπολογισμούς με πίνακες. Το **GNU Octave** είναι ελεύθερο λογισμικό με σχεδόν πλήρως συμβατή γλώσσα· όλα τα αρχεία `.m` του φυλλαδίου εκτελούνται και στα δύο. Η γλώσσα είναι *διερμηνευόμενη*: κάθε εντολή εκτελείται αμέσως, χωρίς μεταγλώττιση, και οι μεταβλητές δεν χρειάζεται να δηλωθούν· ο τύπος και το μέγεθός τους καθορίζονται από την τιμή που τους ανατίθεται.

Το γραφικό περιβάλλον έχει τέσσερα κύρια μέρη:

- τον *περιηγητή αρχείων* (File Browser), από τον οποίο επιλέγεται ο *τρέχων φάκελος*· εκεί αναζητούνται πρώτα τα αρχεία `.m`,
- το *παράθυρο εντολών* (Command Window), όπου οι εντολές πληκτρολογούνται μετά το σύμβολο `>>` και εκτελούνται μία προς μία,
- τον *χώρο εργασίας* (Workspace), με τις μεταβλητές που υπάρχουν και τις τιμές τους,
- τον *επεξεργαστή* (Editor), όπου γράφονται και αποθηκεύονται τα προγράμματα και οι συναρτήσεις.

Μια εντολή που τελειώνει με `;` εκτελείται χωρίς να τυπώσει το αποτέλεσμά της· χωρίς το `;` τυπώνεται η τιμή της μεταβλητής. Όταν ένα αποτέλεσμα δεν ανατίθεται σε μεταβλητή, αποθηκεύεται στη μεταβλητή `ans`. Πολλές εντολές μπορούν να γραφούν στην ίδια γραμμή, χωρισμένες με κόμμα ή `;`, και μια μεγάλη εντολή συνεχίζεται στην επόμενη γραμμή με τρεις τελείες (`...`). Τα ονόματα των μεταβλητών ξεχωρίζουν πεζά από κεφαλαία (`A` και `a` είναι διαφορετικές μεταβλητές).

Παράθυρο εντολών `ses_basics.m`

```
>> x = 3 + 4*2
x = 11
>> y = 2^10;              % a semicolon suppresses the output
>> y
y = 1024
>> z = x / y
z = 0.010742
```

```
>> format long
>> z
z = 1.074218750000000e-02
>> format short
>> 1/0, -1/0, 0/0
ans = Inf
ans = -Inf
ans = NaN
>> [mod(-7, 3), rem(-7, 3), floor(-7/2), fix(-7/2)]
ans =
   2  -1  -4  -3
>> f = @(t) t.^2 - 2;     % anonymous function
>> f(1.5)
ans = 0.2500
>> f([0 1 2])
ans =
  -2  -1   2
```

Οι εντολές κάθε συνεδρίας βρίσκονται σε αρχείο με το όνομα που αναγράφεται στον τίτλο της (εδώ στο αρχείο `ses_basics.m`), ώστε να μπορούν να επαναληφθούν. Τα αποτελέσματα εμφανίζονται όπως τα τυπώνει το Octave με την επιλογή `format compact`· το MATLAB τα τυπώνει με λίγο διαφορετικά κενά. Το `format long` δείχνει 16 σημαντικά ψηφία και το `format short` (η προεπιλογή) 5. Η μορφή εμφάνισης δεν επηρεάζει την ακρίβεια των υπολογισμών, που γίνονται πάντα σε διπλή ακρίβεια (Ενότητα 1.8). Η τελευταία εντολή ορίζει μια *ανώνυμη συνάρτηση*, `f = @(t) t.^2 - 2`, που μπορεί να δοθεί ως όρισμα σε άλλες συναρτήσεις· με αυτόν τον τρόπο θα περνάμε στις μεθόδους των επόμενων κεφαλαίων τη συνάρτηση της οποίας ζητείται η ρίζα, η παράγωγος ή το ολοκλήρωμα.

Εντολή	Λειτουργία
`help sqrt`, `doc sqrt`	βοήθεια για μια εντολή ή συνάρτηση
`lookfor determinant`	αναζήτηση συναρτήσεων με λέξη-κλειδί
`who`, `whos`	οι μεταβλητές του χώρου εργασίας (και τα μεγέθη τους)
`clear x`, `clear`	διαγραφή μιας ή όλων των μεταβλητών
`clc`	καθαρισμός του παραθύρου εντολών
`format long`, `format short`	πλήθος ψηφίων στην εμφάνιση
`pwd`, `cd`, `ls`	τρέχων φάκελος, αλλαγή φακέλου, περιεχόμενα
`which addition`	από ποιο αρχείο φορτώνεται μια συνάρτηση
`tic`, `toc`	μέτρηση χρόνου εκτέλεσης
`%` (ή `#` στο Octave)	σχόλιο μέχρι το τέλος της γραμμής
`Ctrl-C`	διακοπή εντολής που εκτελείται (π.χ. ατέρμονος βρόχος)

Πίνακας 1.1: Χρήσιμες εντολές του περιβάλλοντος.

1.2 Εκτελέσιμο πρόγραμμα και συνάρτηση

Ένα *πρόγραμμα* (script) είναι ένα αρχείο `.m` με εντολές που εκτελούνται με τη σειρά, σαν να είχαν πληκτρολογηθεί στο παράθυρο εντολών· εκτελείται γράφοντας το όνομά του χωρίς την κατάληξη. Μια *συνάρτηση* δέχεται ορίσματα και επιστρέφει αποτελέσματα· οι μεταβλητές της είναι τοπικές και δεν εμφανίζονται στον χώρο εργασίας. Κάθε συνάρτηση αποθηκεύεται σε δικό της αρχείο, με όνομα *ίδιο* με το όνομα της συνάρτησης, στον τρέχοντα φάκελο ή σε φάκελο του `path`. Η πρώτη γραμμή σχολίων μετά τη δήλωση `function` εμφανίζεται με την εντολή `help`.

Παράδειγμα 1.1 (Κλήση συνάρτησης)**.** Να υπολογιστεί το άθροισμα των αριθμών 10 και 5 με χρήση συνάρτησης.

Octave/MATLAB addition.m

```
function s = addition(a, b)
% ADDITION  Returns the sum of a and b.
  s = a + b;
end
```

Octave/MATLAB main.m

```
% Calls the function addition (file addition.m in the same folder)
a = 10;
b = 5;
s = addition(a, b);
fprintf('%g + %g = %g\n', a, b, s);
```

Η εντολή `main` τυπώνει `10 + 5 = 15`. Μια συνάρτηση μπορεί να επιστρέφει περισσότερα από ένα αποτελέσματα, π.χ. `function [s, k] = myexp(x)` (Παράδειγμα 1.9), που καλείται ως `[s, k] = myexp(1)`· αν ζητηθεί μόνο το πρώτο, γράφουμε `s = myexp(1)`. Η μεταβλητή `nargin` δίνει μέσα στη συνάρτηση το πλήθος των ορισμάτων με τα οποία κλήθηκε, ώστε να μπορούν να οριστούν προεπιλεγμένες τιμές.

1.3 Βασικές πράξεις

Πράξη	Octave/MATLAB	Παράδειγμα
Πρόσθεση, αφαίρεση	`a + b`, `a - b`	`7 - 2` $\to 5$
Πολλαπλασιασμός	`a * b`	`7 * 2` $\to 14$
Διαίρεση	`a / b`	`7 / 2` $\to 3.5$
Ύψωση σε δύναμη	`a ^ b`	`2 ^ 10` $\to 1024$
Ακέραιο πηλίκο	`floor(a / b)`, `fix(a / b)`	`floor(-7/2)` $\to -4$, `fix(-7/2)` $\to -3$
Υπόλοιπο	`mod(a, b)`, `rem(a, b)`	`mod(-7,3)` $\to 2$, `rem(-7,3)` $\to -1$
Πράξη ανά στοιχείο	`.*`, `./`, `.^`	`[1 2 3].^2` $\to$ `[1 4 9]`

Πίνακας 1.2: Αριθμητικοί τελεστές.

Η προτεραιότητα των πράξεων είναι η συνήθης των μαθηματικών: πρώτα οι παρενθέσεις, μετά η ύψωση σε δύναμη, μετά το πρόσημο, ο πολλαπλασιασμός και η διαίρεση (από αριστερά προς τα δεξιά) και τέλος η πρόσθεση και η αφαίρεση. Έτσι το `-2^2` ισούται με -4 και το `2^-1` με 0.5. Τρία σημεία χρειάζονται προσοχή:

- Για **αρνητικούς αριθμούς** το υπόλοιπο ορίζεται με δύο τρόπους: το `mod(-7,3)` δίνει 2 (πρόσημο του διαιρέτη), ενώ το `rem(-7,3)` δίνει -1 (πρόσημο του διαιρετέου). Αντίστοιχα η `floor` στρογγυλεύει προς το $-\infty$ και η `fix` προς το μηδέν.
- Όταν οι τελεστές εφαρμόζονται σε **πίνακες**, τα `*`, `/` και `^` είναι πράξεις της Γραμμικής Άλγεβρας (Ενότητα 1.6). Για πράξεις στοιχείο προς στοιχείο χρησιμοποιούνται τα `.*`, `./` και `.^`.
- Όλοι οι αριθμοί είναι εξ ορισμού **πραγματικοί διπλής ακρίβειας** (`double`). Ακέραιοι τύποι, όπως `int32` και `int64`, υπάρχουν, αλλά οι πράξεις τους *στρογγυλεύουν* στον πλησιέστερο ακέραιο και, όταν το αποτέλεσμα ξεπερνά το μέγιστο, δίνουν το μέγιστο (`intmax`) αντί για σφάλμα.

1.4 Συναρτήσεις βιβλιοθήκης και ειδικές σταθερές

Οι μαθηματικές συναρτήσεις είναι ενσωματωμένες και εφαρμόζονται σε πίνακες στοιχείο προς στοιχείο: το `sqrt([4 9 16])` δίνει `[2 3 4]`.

Συνάρτηση	Octave/MATLAB
ημίτονο, συνημίτονο, εφαπτομένη	`sin cos tan` (σε μοίρες: `sind cosd tand`)
αντίστροφες τριγωνομετρικές	`asin acos atan`, `atan2(y, x)`
υπερβολικές	`sinh cosh tanh`
εκθετική, λογάριθμοι	`exp`, `log`, `log10`, `log2`
τετραγωνική ρίζα, απόλυτη τιμή	`sqrt`, `abs`
πλησιέστερος ακέραιος	`round`
προς τα πάνω / προς τα κάτω / προς το μηδέν	`ceil`, `floor`, `fix`
πρόσημο	`sign`
παραγοντικό, διωνυμικός συντελεστής	`factorial(n)`, `nchoosek(n, k)`

Πίνακας 1.3: Συνηθέστερες μαθηματικές συναρτήσεις. Οι γωνίες δίνονται σε ακτίνια.

Σταθερά	Octave/MATLAB	Τιμή
π	`pi`	$3.14159\ldots$
e	`exp(1)`	$2.71828\ldots$
έψιλον μηχανής	`eps`	2.22×10^{-16}
μέγιστος / ελάχιστος κανονικός αριθμός	`realmax`, `realmin`	1.80×10^{308}, 2.23×10^{-308}
άπειρο	`Inf`	π.χ. `1/0`
μη αριθμός	`NaN`	π.χ. `0/0`
φανταστική μονάδα	`i`, `j`, `1i`	$\sqrt{-1}$
τελευταίο αποτέλεσμα	`ans`	

Πίνακας 1.4: Ειδικές σταθερές και μεταβλητές. Τα `i` και `j` μπορούν να χρησιμοποιηθούν και ως μεταβλητές (π.χ. μετρητές βρόχων)· τότε χάνεται η φανταστική μονάδα, γι' αυτό σε υπολογισμούς με μιγαδικούς γράφουμε `1i`.

1.5 Δομές ελέγχου και επανάληψης

Οι συνθήκες σχηματίζονται με σχεσιακούς και λογικούς τελεστές (Πίνακας 1.5). Το αποτέλεσμα μιας σύγκρισης είναι `1` (αληθές) ή `0` (ψευδές)· για πίνακες η σύγκριση γίνεται στοιχείο προς στοιχείο και δίνει λογικό πίνακα.

	Octave/MATLAB	Παράδειγμα
ίσο, διάφορο	`==`, `~=`	`3 ~= 4` $\to 1$
σύγκριση	`< <= > >=`	`[1 5 2] > 2` $\to$ `[0 1 0]`
ΚΑΙ, Ή (για βαθμωτά, με βραχυκύκλωμα)	`&&`, `||`	`x > 0 && log(x) < 1`
ΚΑΙ, Ή στοιχείο προς στοιχείο	`&`, `|`	`(v > 0) & (v < 5)`
ΟΧΙ	`~`	`~(3 > 4)` $\to 1$
όλα / κάποιο αληθές	`all`, `any`	`any([0 0 1])` $\to 1$

Πίνακας 1.5: Σχεσιακοί και λογικοί τελεστές. Το `=` είναι πάντα ανάθεση τιμής, το `==` έλεγχος ισότητας.

1.5.1 Η εντολή if

Η εντολή `if` εκτελεί μια ομάδα εντολών μόνο όταν ισχύει μια συνθήκη· με τα `elseif`/`else` εξετάζονται εναλλακτικές περιπτώσεις. Το `elseif` γράφεται ως μία λέξη, και κάθε δομή κλείνει με `end`. Το παράδειγμα ελέγχει το πλήθος των πραγματικών ριζών ενός τριωνύμου.

Octave/MATLAB `ctrl_if.m`

```
% if / elseif / else: number of real roots of a*x^2 + b*x + c = 0
a = 1;  b = -3;  c = 2;
d = b^2 - 4 * a * c;
if d > 0
  fprintf('two real roots: %g, %g\n', (-b - sqrt(d)) / (2 * a), (-b + sqrt(d)) / (2 * a));
elseif d == 0
  fprintf('one double root: %g\n', -b / (2 * a));
else
  fprintf('no real roots\n');
end
```

1.5.2 Η εντολή switch

Η `switch` επιλέγει μία από πολλές ομάδες εντολών ανάλογα με την τιμή μιας έκφρασης, που μπορεί να είναι αριθμός ή συμβολοσειρά. Εκτελείται *μόνο* η περίπτωση (`case`) που ταιριάζει· αν δεν ταιριάζει καμία, εκτελείται η `otherwise`. Μια περίπτωση μπορεί να καλύπτει πολλές τιμές, π.χ. `case {'FCC', 'fcc'}`.

Παράδειγμα 1.2 (Θεωρητική πυκνότητα μετάλλου)**.** Η θεωρητική πυκνότητα ενός κρυσταλλικού στερεού είναι

$$\rho = \frac{n\,A}{V_c\,N_A},$$

όπου n το πλήθος των ατόμων ανά μοναδιαία κυψελίδα, A η ατομική μάζα, V_c ο όγκος της κυψελίδας και N_A η σταθερά Avogadro. Για τις κυβικές δομές $V_c = a^3$ και $n = 1$ (απλή κυβική, SC), $n = 2$ (χωροκεντρωμένη, BCC) ή $n = 4$ (εδροκεντρωμένη, FCC). Να γραφεί συνάρτηση που υπολογίζει την πυκνότητα και να εφαρμοστεί στα Cu (FCC, $a = 0{,}3615\,\text{nm}$, $A = 63{,}55\,\text{g/mol}$), Fe (BCC, 0,2866 nm, 55,85 g/mol) και Al (FCC, 0,4049 nm, 26,98 g/mol).

Λύση. Η επιλογή του n από τη δομή γίνεται με `switch`· για άγνωστη δομή η `error` τερματίζει την εκτέλεση με μήνυμα. Προσοχή στις μονάδες: η πλεγματική σταθερά μετατρέπεται από nm σε cm ($1\,\text{nm} = 10^{-7}\,\text{cm}$), ώστε η πυκνότητα να προκύψει σε g/cm^3.

Octave/MATLAB `density.m`

```
function rho = density(structure, a_nm, A)
% DENSITY  Theoretical density (g/cm^3) of a cubic metal.
%   structure : 'SC', 'BCC' or 'FCC'
%   a_nm      : lattice parameter in nm,   A : atomic mass in g/mol
  NA = 6.02214076e23;                    % Avogadro constant, 1/mol
  switch structure
    case 'SC'
      n = 1;
    case 'BCC'
      n = 2;
    case 'FCC'
      n = 4;
    otherwise
      error('density: unknown structure %s', structure);
  end
  a_cm = a_nm * 1e-7;                    % nm -> cm
  rho  = n * A / (a_cm^3 * NA);
end
```

Octave/MATLAB `ex_density.m`

```
% Chapter 1 example: theoretical density of Cu, Fe and Al
fprintf('Cu (FCC): %.2f g/cm^3\n', density('FCC', 0.3615, 63.55));
fprintf('Fe (BCC): %.2f g/cm^3\n', density('BCC', 0.2866, 55.85));
fprintf('Al (FCC): %.2f g/cm^3\n', density('FCC', 0.4049, 26.98));
```

Έξοδος

```
Cu (FCC): 8.94 g/cm^3
Fe (BCC): 7.88 g/cm^3
Al (FCC): 2.70 g/cm^3
```

Οι τιμές συμφωνούν με τις πειραματικές πυκνότητες (8.94, 7.87 και $2{,}70\,\mathrm{g/cm^3}$).

1.5.3 Βρόχοι for

Ο βρόχος `for k = a:h:b` επαναλαμβάνει μια ομάδα εντολών για τις τιμές $k = a, a+h, a+2h, \ldots$ που δεν ξεπερνούν το b· αν παραλειφθεί το βήμα h, είναι 1. Γενικότερα, ο μετρητής διατρέχει τις στήλες οποιουδήποτε πίνακα, π.χ. `for x = [1 10 -1]`.

Octave/MATLAB `ctrl_for.m`

```
% for loop: partial sums of 1/k^2 (the limit is pi^2/6)
n = 1000;
s = 0;
for k = 1:n
  s = s + 1 / k^2;
end
fprintf('sum = %.8f   pi^2/6 = %.8f\n', s, pi^2 / 6);

for k = 10:-2:1                          % negative step: 10, 8, 6, 4, 2
  fprintf('%d ', k);
end
fprintf('\n');
```

Έξοδος

```
sum = 1.64393457   pi^2/6 = 1.64493407
10 8 6 4 2
```

1.5.4 Βρόχοι while

Ο βρόχος `while` επαναλαμβάνεται όσο ισχύει μια συνθήκη· χρησιμοποιείται όταν το πλήθος των επαναλήψεων δεν είναι γνωστό εκ των προτέρων, όπως σε όλες τις επαναληπτικές μεθόδους των επόμενων κεφαλαίων. Η `break` τερματίζει αμέσως τον βρόχο και η `continue` προχωρά στην επόμενη επανάληψη. Το παράδειγμα υπολογίζει το *έψιλον της μηχανής*, δηλαδή την απόσταση του 1 από τον επόμενο αριθμό κινητής υποδιαστολής.

Octave/MATLAB `macheps.m`

```
% while loop: machine epsilon of double precision
e = 1;
k = 0;
while 1 + e / 2 > 1
  e = e / 2;
  k = k + 1;
end
fprintf('eps = %.6e = 2^(-%d)   built-in: %.6e\n', e, k, eps);
```

Έξοδος

```
eps = 2.220446e-16 = 2^(-52)   built-in: 2.220446e-16
```

1.6 Διανύσματα και πίνακες

Στο Octave/MATLAB κάθε μεταβλητή είναι πίνακας: ένας αριθμός είναι πίνακας 1×1, ένα διάνυσμα γραμμή πίνακας $1 \times n$ και ένα διάνυσμα στήλη πίνακας $n \times 1$. Οι πράξεις εφαρμόζονται σε ολόκληρους πίνακες, χωρίς βρόχους, και οι βασικοί υπολογισμοί της Γραμμικής Άλγεβρας γίνονται με μία εντολή. Στην ενότητα αυτή συνοψίζουμε τους ορισμούς και τις ιδιότητες που χρειάζονται στα επόμενα κεφάλαια (ιδίως στο Κεφάλαιο 2) και τις αντίστοιχες εντολές.

1.6.1 Δημιουργία διανυσμάτων και πινάκων

Τα στοιχεία ενός πίνακα γράφονται μέσα σε αγκύλες· τα στοιχεία μιας γραμμής χωρίζονται με κενό ή κόμμα και οι γραμμές με `;`. Η σύνταξη `a:h:b` δίνει διάνυσμα με σταθερό βήμα και η `linspace(a, b, n)` δίνει n ισαπέχοντα σημεία του $[a, b]$, μαζί με τα άκρα.

Παράθυρο εντολών `mat_create.m`

```
>> v = [1 2 3]               % row vector
v =
   1   2   3
>> w = [4; 5; 6]             % column vector
w =
   4
   5
   6
>> t = 0:0.25:1              % start:step:end
t =
         0    0.2500    0.5000    0.7500    1.0000
>> u = linspace(0, 1, 5)     % 5 equally spaced points in [0, 1]
u =
        0   0.2500   0.5000   0.7500   1.0000
>> A = [1 2 3; 4 5 6; 7 8 10]
A =
    1    2    3
    4    5    6
    7    8   10
>> size(A)
ans =
   3   3
>> numel(A)
ans = 9
>> Z = zeros(2, 3)
```

```
Z =
   0   0   0
   0   0   0
>> I = eye(3)
I =
Diagonal Matrix
   1   0   0
   0   1   0
   0   0   1
>> D = diag([2 -1 5])
D =
Diagonal Matrix
   2   0   0
   0  -1   0
   0   0   5
>> diag(A)'                  % main diagonal of A, as a row
ans =
    1    5   10
>> B = [A, w]                % concatenation: 3x4 matrix
B =
    1    2    3    4
    4    5    6    5
    7    8   10    6
```

Το Octave σημειώνει ότι οι `eye(3)` και `diag(...)` είναι διαγώνιοι πίνακες (Diagonal Matrix), τους οποίους αποθηκεύει οικονομικά· στους υπολογισμούς συμπεριφέρονται όπως κάθε πίνακας. Άλλες συναρτήσεις δημιουργίας είναι οι `ones(m,n)` (μονάδες), `rand(m,n)` (τυχαίοι αριθμοί στο $(0, 1)$), `magic(n)` και `pascal(n)` (πίνακες με ειδικές ιδιότητες, χρήσιμοι για δοκιμές) και `repmat`, που επαναλαμβάνει έναν πίνακα.

1.6.2 Δείκτες, υποπίνακες και λογικοί δείκτες

Το στοιχείο της γραμμής i και της στήλης j είναι το `A(i,j)`· η αρίθμηση αρχίζει από το **1**. Η λέξη `end` δηλώνει τον τελευταίο δείκτη, και η άνω-κάτω τελεία μόνη της (`:`) όλες τις γραμμές ή στήλες. Με διανύσματα δεικτών επιλέγεται ολόκληρος υποπίνακας, και με λογικό διάνυσμα (αποτέλεσμα σύγκρισης) επιλέγονται τα στοιχεία που ικανοποιούν μια συνθήκη.

Παράθυρο εντολών `mat_index.m`

```
>> A = [1 2 3; 4 5 6; 7 8 10];
>> A(2, 3)
ans = 6
>> A(3, :)                   % third row
ans =
    7    8   10
>> A(:, 2)'                  % second column, as a row
ans =
   2   5   8
>> A(end, end)
ans = 10
>> A(1:2, 2:3)               % submatrix
ans =
   2   3
   5   6
>> A(2, :) = [0 1 0]         % replace the second row
A =
    1    2    3
    0    1    0
```

```
    7    8   10
>> sum(A)                    % column sums
ans =
    8   11   13
>> sum(A, 2)'                % row sums
ans =
    6    1   25
>> v = [3 -1 4 -1 5 -9 2 6];
>> v(v < 0)                  % elements that satisfy a condition
ans =
  -1  -1  -9
>> find(v < 0)               % their positions
ans =
   2   4   6
>> [m, k] = max(v)           % largest element and its position
m = 6
k = 8
>> sort(v)
ans =
  -9  -1  -1   2   3   4   5   6
>> cumsum([1 2 3 4])
ans =
    1    3    6   10
```

Οι `sum`, `prod`, `max`, `min` και `mean` εφαρμόζονται σε πίνακα *κατά στήλες*· με δεύτερο όρισμα `2` (`sum(A, 2)`) εφαρμόζονται κατά γραμμές, και με `A(:)`, που μετατρέπει τον πίνακα σε ένα διάνυσμα στήλη, σε όλα τα στοιχεία. Μια γραμμή ή στήλη διαγράφεται με ανάθεση κενού πίνακα, π.χ. `A(2, :) = []`.

1.6.3 Πράξεις πινάκων και διανυσμάτων

Πράξεις πινάκων

- **Άθροισμα** $A + B$ και **πολλαπλασιασμός με αριθμό** cA γίνονται στοιχείο προς στοιχείο· το άθροισμα ορίζεται για πίνακες ίδιων διαστάσεων.
- **Γινόμενο** πίνακα A διαστάσεων $m \times p$ επί πίνακα B διαστάσεων $p \times n$ είναι ο πίνακας $C = AB$ διαστάσεων $m \times n$ με

$$c_{ij} = \sum_{k=1}^{p} a_{ik} b_{kj},$$

  δηλαδή το στοιχείο c_{ij} είναι το εσωτερικό γινόμενο της γραμμής i του A με τη στήλη j του B. Ορίζεται μόνο όταν το πλήθος των στηλών του A ισούται με το πλήθος των γραμμών του B. Ισχύουν $(AB)C = A(BC)$ και $A(B + C) = AB + AC$, αλλά *εν γένει* $AB \neq BA$.
- Για διανύσματα στήλες $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$: *εσωτερικό γινόμενο* $\mathbf{u}^\top \mathbf{v} = \sum_i u_i v_i$ (αριθμός), *εξωτερικό γινόμενο* $\mathbf{u}\mathbf{v}^\top$ (πίνακας $n \times n$ με στοιχεία $u_i v_j$) και, για $n = 3$, *εξωτερικό (διανυσματικό) γινόμενο* $\mathbf{u} \times \mathbf{v} = (u_2 v_3 - u_3 v_2,\ u_3 v_1 - u_1 v_3,\ u_1 v_2 - u_2 v_1)$, κάθετο και στα δύο. Η γωνία φ δύο διανυσμάτων δίνεται από τη σχέση $\mathbf{u}^\top \mathbf{v} = \|\mathbf{u}\|_2 \|\mathbf{v}\|_2 \cos\varphi$.

Παράθυρο εντολών `mat_ops.m`

```
>> A = [1 2; 3 4];  B = [0 1; 1 0];
>> A + B
ans =
   1   3
   4   4
>> A * B                      % matrix product
ans =
   2   1
   4   3
>> B * A                      % in general A*B ~= B*A
ans =
   3   4
   1   2
>> A .* B                     % element-wise product
ans =
   0   2
   3   0
>> A .^ 2                     % element-wise power
ans =
    1    4
    9   16
>> A ^ 2                      % matrix power: A*A
ans =
    7   10
   15   22
>> 2 * A - 1                  % scalars act on every element
ans =
   1   3
   5   7
>> A * [1; 1]                 % matrix-vector product
ans =
   3
   7
>> u = [1; 2; 2];  v = [3; 0; -4];
>> u' * v                     % inner (dot) product
ans = -5
>> dot(u, v)
ans = -5
>> u * v'                     % outer product: 3x3 matrix
ans =
   3   0  -4
   6   0  -8
   6   0  -8
>> cross(u, v)'               % cross product (3-vectors only)
ans =
   -8   10   -6
```

Η διαφορά των `A*B` και `A.*B`, και των `A^2` και `A.^2`, είναι η συχνότερη πηγή λαθών στο Octave. Αν οι διαστάσεις δεν είναι συμβατές, το πρόγραμμα σταματά με μήνυμα σφάλματος, π.χ. για `[1 2 3] * [4 5 6]`: `operator *: nonconformant arguments (op1 is 1x3, op2 is 1x3)`· το σωστό είναι `[1 2 3] * [4 5 6]'` (εσωτερικό γινόμενο) ή `[1 2 3] .* [4 5 6]` (γινόμενο ανά στοιχείο). Οι πράξεις ανά στοιχείο μεταξύ διανύσματος γραμμής και διανύσματος στήλης επεκτείνουν αυτόματα τις διαστάσεις (*broadcasting*): το `[1 2 3] + [10; 20]` δίνει πίνακα 2×3.

1.6.4 Ανάστροφος πίνακας

Ανάστροφος

Ο ανάστροφος του πίνακα A διαστάσεων $m \times n$ είναι ο πίνακας $A^\top$ διαστάσεων $n \times m$ με $(A^\top)_{ij} = a_{ji}$: οι γραμμές του A γίνονται στήλες. Ισχύουν

$$(A^\top)^\top = A, \qquad (A+B)^\top = A^\top + B^\top, \qquad (AB)^\top = B^\top A^\top.$$

Ο πίνακας είναι *συμμετρικός* όταν $A^\top = A$· ο $A^\top A$ είναι πάντα συμμετρικός. Ένας τετραγωνικός πίνακας Q λέγεται *ορθογώνιος* όταν $Q^\top Q = I$, δηλαδή $Q^{-1} = Q^\top$· οι στήλες του είναι κάθετα μοναδιαία διανύσματα και ο πολλαπλασιασμός με Q διατηρεί τα μήκη.

Στο Octave ο ανάστροφος γράφεται `A'`. Για μιγαδικούς πίνακες το `A'` δίνει τον *συζυγή* ανάστροφο ($\bar{a}_{ji}$), ενώ το `A.'` τον απλό ανάστροφο· για πραγματικούς πίνακες τα δύο συμπίπτουν.

Παράθυρο εντολών `mat_transpose.m`

```
>> A = [1 2 3; 4 5 6]
A =
   1   2   3
   4   5   6
>> A'
ans =
   1   4
   2   5
   3   6
>> B = [1 0; 2 1; 0 3];
>> (A * B)'
ans =
    5   14
   11   23
>> B' * A'                    % (AB)' = B'A'
ans =
    5   14
   11   23
>> S = A' * A                 % always symmetric
S =
   17   22   27
   22   29   36
   27   36   45
>> isequal(S, S')
ans = 1
>> z = [1+2i; 3-1i];
>> z'                         % conjugate transpose
ans =
   1 - 2i   3 + 1i
>> z.'                        % transpose without conjugation
ans =
   1 + 2i   3 - 1i
```

Χαρακτηριστικό παράδειγμα ορθογώνιου πίνακα είναι ο πίνακας στροφής κατά γωνία θ στο επίπεδο,

$$R(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix},$$

για τον οποίο $R(\theta)^\top = R(-\theta) = R(\theta)^{-1}$ και $\det R = 1$.

Παράθυρο εντολών `mat_rotation.m`

```
>> th = pi / 6;               % 30 degrees
>> R = [cos(th) -sin(th); sin(th) cos(th)]
R =
   0.8660  -0.5000
   0.5000   0.8660
>> norm(R' * R - eye(2))      % R'R = I, i.e. inv(R) = R'
ans = 7.4371e-18
>> det(R)
ans = 1
>> x = [2; 1];
>> [norm(R * x), norm(x)]     % lengths are preserved
ans =
   2.2361   2.2361
>> eig(R)                     % cos(th) +- i sin(th)
ans =
   0.8660 + 0.5000i
   0.8660 - 0.5000i
```

Το `norm(R' * R - eye(2))` δεν είναι ακριβώς μηδέν αλλά της τάξης του 10^{-17}, λόγω στρογγύλευσης των $\cos\theta$ και $\sin\theta$· αυτό είναι το αναμενόμενο σε κάθε υπολογισμό με αριθμούς κινητής υποδιαστολής. Οι ιδιοτιμές του R είναι $\cos\theta \pm i\sin\theta$, μιγαδικές με μέτρο 1.

1.6.5 Ορίζουσα

Ορίζουσα

Για τετραγωνικό πίνακα A διαστάσεων $n\times n$ η ορίζουσα ορίζεται αναδρομικά με το *ανάπτυγμα κατά την πρώτη γραμμή*

$$\det A = \sum_{j=1}^{n} (-1)^{1+j} a_{1j} \det M_{1j},$$

όπου M_{1j} ο πίνακας που προκύπτει από τον A αν διαγραφούν η γραμμή 1 και η στήλη j. Η ποσότητα $C_{ij} = (-1)^{i+j}\det M_{ij}$ λέγεται *συμπαράγοντας* του a_{ij}· το ανάπτυγμα μπορεί να γίνει κατά οποιαδήποτε γραμμή ή στήλη. Για $n=2$: $\det\begin{pmatrix} a & b \\ c & d \end{pmatrix} = ad - bc$.

Ιδιότητες. $\det A^{\top} = \det A$, $\det(AB) = \det A \det B$, $\det(cA) = c^n \det A$, $\det A^{-1} = 1/\det A$. Η εναλλαγή δύο γραμμών αλλάζει το πρόσημο της ορίζουσας, ενώ η πρόσθεση πολλαπλασίου μιας γραμμής σε άλλη δεν την αλλάζει. Η ορίζουσα τριγωνικού πίνακα είναι το γινόμενο των διαγώνιων στοιχείων. Ο A είναι αντιστρέψιμος αν και μόνο αν $\det A \neq 0$.

Παράδειγμα 1.3 (Ορίζουσα 3×3)**.** Να υπολογιστεί η ορίζουσα του $A = \begin{pmatrix} 2 & -1 & 0 \\ 1 & 3 & 2 \\ 0 & 1 & 4 \end{pmatrix}$ με το χέρι και να επαληθευτούν στο Octave οι ιδιότητές της.

Λύση. Με ανάπτυγμα κατά την πρώτη γραμμή:

$$\det A = 2\begin{vmatrix} 3 & 2 \\ 1 & 4 \end{vmatrix} - (-1)\begin{vmatrix} 1 & 2 \\ 0 & 4 \end{vmatrix} + 0\begin{vmatrix} 1 & 3 \\ 0 & 1 \end{vmatrix} = 2\cdot 10 + 1\cdot 4 + 0 = 24.$$

Παράθυρο εντολών `mat_det.m`

```
>> A = [2 -1 0; 1 3 2; 0 1 4];
>> det(A)
ans = 24
>> det(A')                    % det(A') = det(A)
ans = 24
>> B = [1 2 0; 0 1 1; 1 0 1];
>> [det(A*B), det(A)*det(B)]  % det(AB) = det(A) det(B)
ans =
   72   72
>> [det(3*A), 3^3*det(A)]     % det(cA) = c^n det(A) for n x n
ans =
   648   648
>> det(A([2 1 3], :))         % swapping two rows changes the sign
ans = -24
>> M = magic(4)                % singular: rank 3
M =
   16    2    3   13
    5   11   10    8
    9    7    6   12
    4   14   15    1
>> det(M)                     % rounding error instead of 0
ans = -1.4495e-12
>> rank(M)
ans = 3
>> det(0.1 * eye(20))         % tiny determinant, yet a harmless matrix
ans = 1.0000e-20
```

Δύο παρατηρήσεις είναι σημαντικές. Πρώτον, ο πίνακας `magic(4)` είναι ιδιόμορφος (η τάξη του είναι 3), αλλά η `det` δεν δίνει μηδέν: τα σφάλματα στρογγύλευσης δίνουν έναν μικρό αριθμό, του οποίου η ακριβής τιμή εξαρτάται από τη σειρά των πράξεων. Δεύτερον, ο $0.1\,I$ διαστάσεων 20×20 έχει ορίζουσα 10^{-20}, αλλά είναι ένας απολύτως «ακίνδυνος» πίνακας, που αντιστρέφεται χωρίς καμία απώλεια ακρίβειας. Η ορίζουσα επομένως *δεν* είναι μέτρο του πόσο κοντά είναι ένας πίνακας στο να είναι ιδιόμορφος· το μέτρο αυτό είναι ο δείκτης κατάστασης (Ενότητα 1.6.8 και Κεφάλαιο 2). Ο έλεγχος ιδιομορφίας γίνεται με την `rank` ή την `cond`, ποτέ με `det(A) == 0`. Η `det` δεν χρησιμοποιεί το ανάπτυγμα, που απαιτεί περίπου $n!$ πράξεις (Άσκηση 1.21), αλλά την απαλοιφή Gauss του Κεφαλαίου 2, με περίπου $\frac{2}{3}n^3$ πράξεις.

1.6.6 Αντίστροφος πίνακας

Αντίστροφος

Ο αντίστροφος ενός τετραγωνικού πίνακα A είναι ο πίνακας A^{-1} με $AA^{-1} = A^{-1}A = I$. Υπάρχει αν και μόνο αν $\det A \neq 0$ και δίνεται από τον τύπο

$$A^{-1} = \frac{1}{\det A}\,\operatorname{adj} A, \qquad (\operatorname{adj} A)_{ij} = C_{ji},$$

όπου $\operatorname{adj} A$ ο *προσαρτημένος* πίνακας, ο ανάστροφος του πίνακα των συμπαραγόντων. Για $n = 2$:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} = \frac{1}{ad - bc}\begin{pmatrix} d & -b \\ -c & a \end{pmatrix}.$$

Ιδιότητες. $(A^{-1})^{-1} = A$, $(AB)^{-1} = B^{-1}A^{-1}$, $(A^{\top})^{-1} = (A^{-1})^{\top}$. Αν A αντιστρέψιμος, το σύστημα $A\mathbf{x} = \mathbf{b}$ έχει τη μοναδική λύση $\mathbf{x} = A^{-1}\mathbf{b}$. Η *τάξη* (rank) ενός πίνακα είναι το πλήθος των γραμμικά ανεξάρτητων γραμμών (ή στηλών) του· ένας $n \times n$ πίνακας είναι αντιστρέψιμος αν και μόνο αν έχει τάξη n.

Παράδειγμα 1.4 (Αντίστροφος και επίλυση συστήματος)**.** Για τον πίνακα A του Παραδείγματος 1.3 να βρεθεί ο A^{-1} από τον προσαρτημένο πίνακα και να λυθεί το σύστημα $A\mathbf{x} = (1, 2, 3)^{\top}$.

Λύση. Οι συμπαράγοντες είναι

$$C_{11} = 10,\ C_{12} = -4,\ C_{13} = 1, \quad C_{21} = 4,\ C_{22} = 8,\ C_{23} = -2, \quad C_{31} = -2,\ C_{32} = -4,\ C_{33} = 7,$$

π.χ. $C_{12} = -\begin{vmatrix} 1 & 2 \\ 0 & 4 \end{vmatrix} = -4$ και $C_{21} = -\begin{vmatrix} -1 & 0 \\ 1 & 4 \end{vmatrix} = 4$. Άρα

$$A^{-1} = \frac{1}{24}\begin{pmatrix} 10 & 4 & -2 \\ -4 & 8 & -4 \\ 1 & -2 & 7 \end{pmatrix}, \qquad \mathbf{x} = A^{-1}\begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} = \frac{1}{24}\begin{pmatrix} 12 \\ 0 \\ 18 \end{pmatrix} = \begin{pmatrix} 0.5 \\ 0 \\ 0.75 \end{pmatrix}.$$

Παράθυρο εντολών `mat_inv.m`

```
>> A = [2 -1 0; 1 3 2; 0 1 4];
>> Ai = inv(A)
Ai =
   0.416667   0.166667  -0.083333
  -0.166667   0.333333  -0.166667
   0.041667  -0.083333   0.291667
>> 24 * Ai                    % det(A)*inv(A) = adjugate of A
ans =
   10    4   -2
   -4    8   -4
    1   -2    7
>> A * Ai                     % identity, up to rounding
ans =
   1.0000        0        0
   0.0000   1.0000        0
  -0.0000   0.0000   1.0000
>> b = [1; 2; 3];
>> x = A \ b                  % solves A*x = b (preferred to inv(A)*b)
x =
   0.5000
        0
   0.7500
```

```
>> A * x - b                  % residual
ans =
   0
   0
   0
>> rank([1 2; 2 4])
ans = 1
>> inv([1 2; 2 4])            % singular matrix: warning and Inf
warning: matrix singular to machine precision
ans =
   Inf   Inf
   Inf   Inf
```

Το σύστημα λύνεται με τον τελεστή `\` (*αριστερή διαίρεση*): `x = A \ b`. Ο τελεστής χρησιμοποιεί απαλοιφή Gauss (Κεφάλαιο 2) και είναι ταχύτερος και ακριβέστερος από το `inv(A)*b`, που υπολογίζει πρώτα ολόκληρο τον αντίστροφο. Τα στοιχεία `0.0000` και `-0.0000` του `A*Ai` είναι αριθμοί της τάξης του 10^{-17} (σφάλματα στρογγύλευσης). Για τον ιδιόμορφο πίνακα το Octave εμφανίζει την προειδοποίηση `warning: matrix singular to machine precision` και επιστρέφει `Inf`· το MATLAB δίνει αντίστοιχη προειδοποίηση.

1.6.7 Ιδιοτιμές και ιδιοδιανύσματα

Ιδιοτιμές

Ο αριθμός λ (πραγματικός ή μιγαδικός) είναι *ιδιοτιμή* του τετραγωνικού πίνακα A αν υπάρχει μη μηδενικό διάνυσμα $\mathbf{v}$, το *ιδιοδιάνυσμα*, με

$$A\mathbf{v} = \lambda \mathbf{v}.$$

Οι ιδιοτιμές είναι οι ρίζες του *χαρακτηριστικού πολυωνύμου* $p(\lambda) = \det(\lambda I - A)$, βαθμού n, και κάθε ιδιοδιάνυσμα είναι μη μηδενική λύση του ομογενούς συστήματος $(A - \lambda I)\mathbf{v} = \mathbf{0}$. Για $n = 2$: $p(\lambda) = \lambda^2 - (\operatorname{tr} A)\lambda + \det A$.
Ιδιότητες. $\sum_i \lambda_i = \operatorname{tr} A = \sum_i a_{ii}$ και $\prod_i \lambda_i = \det A$. Οι A και $A^{\top}$ έχουν τις ίδιες ιδιοτιμές. Οι ιδιοτιμές τριγωνικού πίνακα είναι τα διαγώνια στοιχεία του. Ο A^k έχει ιδιοτιμές λ_i^k και ο A^{-1} ιδιοτιμές $1/\lambda_i$. Ένας *συμμετρικός* πραγματικός πίνακας έχει πραγματικές ιδιοτιμές και ορθοκανονική βάση ιδιοδιανυσμάτων: $A = Q\Lambda Q^{\top}$ με Q ορθογώνιο και Λ διαγώνιο.

Παράδειγμα 1.5 (Ιδιοτιμές πίνακα 2×2)**.** Να βρεθούν οι ιδιοτιμές και τα ιδιοδιανύσματα του $A = \begin{pmatrix} 4 & 1 \\ 2 & 3 \end{pmatrix}$.

Λύση. Είναι $\operatorname{tr} A = 7$ και $\det A = 10$, άρα $p(\lambda) = \lambda^2 - 7\lambda + 10 = (\lambda - 5)(\lambda - 2)$ και $\lambda_1 = 5$, $\lambda_2 = 2$. Για $\lambda_1 = 5$ το σύστημα $(A - 5I)\mathbf{v} = \mathbf{0}$ δίνει $-v_1 + v_2 = 0$, άρα $\mathbf{v}_1 = (1, 1)^{\top}$· για $\lambda_2 = 2$ δίνει $2v_1 + v_2 = 0$, άρα $\mathbf{v}_2 = (1, -2)^{\top}$. Η `eig` επιστρέφει τα ιδιοδιανύσματα ως στήλες του `V`, κανονικοποιημένα σε μοναδιαίο μήκος ($\mathbf{v}_1/\sqrt{2}$ και $-\mathbf{v}_2/\sqrt{5}$), και τις ιδιοτιμές στη διαγώνιο του `D`, ώστε $AV = VD$.

Παράθυρο εντολών `mat_eig.m`

```
>> A = [4 1; 2 3];
>> lambda = eig(A)
lambda =
   5
   2
>> [V, D] = eig(A)
V =
   0.7071  -0.4472
```

```
   0.7071   0.8944
D =
Diagonal Matrix
   5   0
   0   2
>> A * V - V * D              % zero, up to rounding
ans =
   0   0
   0   0
>> [sum(lambda), trace(A)]
ans =
   7   7
>> [prod(lambda), det(A)]
ans =
   10   10
>> p = poly(A)                % lambda^2 - 7 lambda + 10
p =
    1   -7   10
>> roots(p)
ans =
   5
   2
>> S = [2 1; 1 2];             % symmetric matrix
>> [Q, L] = eig(S)
Q =
  -0.7071   0.7071
   0.7071   0.7071
L =
Diagonal Matrix
   1   0
   0   3
>> norm(Q' * Q - eye(2))      % Q'Q = I: orthonormal eigenvectors
ans = 2.4442e-16
```

Ένα ιδιοδιάνυσμα ορίζεται μόνο ως προς ένα μη μηδενικό πολλαπλασιαστή, γι' αυτό το πρόσημο ή η σειρά των στηλών που επιστρέφει η `eig` μπορεί να διαφέρει από πρόγραμμα σε πρόγραμμα· σημασία έχει ότι $A\mathbf{v} = \lambda\mathbf{v}$. Η `poly` δίνει τους συντελεστές του χαρακτηριστικού πολυωνύμου και η `roots` τις ρίζες ενός πολυωνύμου. Στην πράξη οι ιδιοτιμές *δεν* υπολογίζονται ως ρίζες του χαρακτηριστικού πολυωνύμου, που είναι πολύ ευαίσθητες σε σφάλματα των συντελεστών (Κεφάλαιο 4), αλλά με επαναληπτικές μεθόδους, όπως η μέθοδος των δυνάμεων του Κεφαλαίου 2.

Παράδειγμα 1.6 (Κύριες τάσεις)**.** Η εντατική κατάσταση σε ένα σημείο ενός στερεού περιγράφεται από τον συμμετρικό τανυστή των τάσεων

$$\sigma = \begin{pmatrix} 80 & 30 & -20 \\ 30 & 40 & 10 \\ -20 & 10 & -30 \end{pmatrix} \text{ MPa}.$$

Οι *κύριες τάσεις* $\sigma_1 \geq \sigma_2 \geq \sigma_3$ είναι οι ιδιοτιμές του σ και οι *κύριες διευθύνσεις* τα αντίστοιχα ιδιοδιανύσματα. Να υπολογιστούν, μαζί με τις αναλλοίωτες $I_1 = \operatorname{tr}\sigma$, $I_2 = \tfrac{1}{2}\big[(\operatorname{tr}\sigma)^2 - \operatorname{tr}(\sigma^2)\big]$, $I_3 = \det\sigma$, τη μέγιστη διατμητική τάση $\tau_{\max} = (\sigma_1 - \sigma_3)/2$ και την ισοδύναμη τάση von Mises

$$\sigma_{\mathrm{VM}} = \sqrt{\tfrac{1}{2}\big[(\sigma_1-\sigma_2)^2 + (\sigma_2-\sigma_3)^2 + (\sigma_3-\sigma_1)^2\big]}.$$

Λύση. Οι κύριες τάσεις είναι οι ρίζες της χαρακτηριστικής εξίσωσης $\sigma^3 - I_1\sigma^2 + I_2\sigma - I_3 = 0$. Επειδή ο τανυστής είναι συμμετρικός, οι ιδιοτιμές είναι πραγματικές και οι κύριες διευθύνσεις κάθετες μεταξύ τους. Το πρόγραμμα ταξινομεί τις ιδιοτιμές κατά φθίνουσα σειρά με τη `sort` και αναδιατάσσει

ανάλογα τις στήλες του `V`· το πρόσημο κάθε διεύθυνσης επιλέγεται ώστε η μεγαλύτερη συνιστώσα της να είναι θετική.

Octave/MATLAB `ex_stress.m`

```
% Principal stresses of a three-dimensional stress state (MPa)
sigma = [80 30 -20; 30 40 10; -20 10 -30];

I1 = trace(sigma);                          % stress invariants
I2 = (I1^2 - trace(sigma^2)) / 2;
I3 = det(sigma);
fprintf('I1 = %.2f, I2 = %.2f, I3 = %.2f\n', I1, I2, I3);

[V, D] = eig(sigma);                        % sigma*V = V*D
[s, idx] = sort(diag(D), 'descend');        % s(1) >= s(2) >= s(3)
V = V(:, idx);
for k = 1:3                                 % sign convention for the
  [~, m] = max(abs(V(:, k)));               % directions: largest
  V(:, k) = V(:, k) * sign(V(m, k));        % component positive
end
fprintf('principal stresses: %.2f  %.2f  %.2f MPa\n', s);
fprintf('principal directions (columns):\n');
fprintf('  %8.4f %8.4f %8.4f\n', V');
fprintf('roots of the characteristic polynomial: %.2f  %.2f  %.2f\n', ...
        sort(roots([1 -I1 I2 -I3]), 'descend'));
fprintf('check: sum = %.2f, product = %.2f\n', sum(s), prod(s));

tau_max = (s(1) - s(3)) / 2;
vm = sqrt(((s(1) - s(2))^2 + (s(2) - s(3))^2 + (s(3) - s(1))^2) / 2);
fprintf('max shear stress = %.2f MPa, von Mises stress = %.2f MPa\n', tau_max, vm);
```

Έξοδος

```
I1 = 90.00, I2 = -1800.00, I3 = -105000.00
principal stresses: 97.41  29.33  -36.75 MPa
principal directions (columns):
    0.8888  -0.4036   0.2171
    0.4462   0.8702  -0.2091
   -0.1045   0.2827   0.9535
roots of the characteristic polynomial: 97.41  29.33  -36.75
check: sum = 90.00, product = -105000.00
max shear stress = 67.08 MPa, von Mises stress = 116.19 MPa
```

Το άθροισμα και το γινόμενο των κύριων τάσεων ισούνται με I_1 και I_3, όπως προβλέπουν οι ιδιότητες των ιδιοτιμών. Η τάση von Mises $116{,}2\,\mathrm{MPa}$ συγκρίνεται με το όριο διαρροής του υλικού: για έναν μαλακό χάλυβα με όριο διαρροής $250\,\mathrm{MPa}$ ο συντελεστής ασφαλείας είναι περίπου 2.15.

1.6.8 Νόρμες διανυσμάτων και πινάκων

Νόρμες

Μια *νόρμα* μετρά το «μέγεθος» ενός διανύσματος ή πίνακα· ικανοποιεί $\|\mathbf{x}\| \geq 0$ (με ισότητα μόνο για $\mathbf{x} = \mathbf{0}$), $\|c\mathbf{x}\| = |c|\,\|\mathbf{x}\|$ και την τριγωνική ανισότητα $\|\mathbf{x}+\mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$.

- **Διανύσματα:** $\|\mathbf{x}\|_1 = \sum_i |x_i|$, $\|\mathbf{x}\|_2 = \left(\sum_i x_i^2\right)^{1/2}$ (ευκλείδεια), $\|\mathbf{x}\|_\infty = \max_i |x_i|$, και γενικά $\|\mathbf{x}\|_p = \left(\sum_i |x_i|^p\right)^{1/p}$.
- **Πίνακες** (επαγόμενες νόρμες, $\|A\| = \max_{\mathbf{x}\neq\mathbf{0}} \|A\mathbf{x}\|/\|\mathbf{x}\|$):

$$\|A\|_1 = \max_j \sum_i |a_{ij}|, \qquad \|A\|_\infty = \max_i \sum_j |a_{ij}|, \qquad \|A\|_2 = \sqrt{\lambda_{\max}(A^\top A)},$$

  δηλαδή το μέγιστο άθροισμα στήλης, το μέγιστο άθροισμα γραμμής και η μεγαλύτερη *ιδιάζουσα τιμή*. Η νόρμα Frobenius $\|A\|_F = \left(\sum_{i,j} a_{ij}^2\right)^{1/2}$ δεν είναι επαγόμενη. Για κάθε επαγόμενη νόρμα $\|A\mathbf{x}\| \leq \|A\|\,\|\mathbf{x}\|$ και $\|AB\| \leq \|A\|\,\|B\|$.
- **Δείκτης κατάστασης:** $\kappa(A) = \|A\|\,\|A^{-1}\| \geq 1$. Όσο μεγαλύτερος είναι, τόσο πιο ευαίσθητη είναι η λύση του $A\mathbf{x} = \mathbf{b}$ σε σφάλματα των δεδομένων (Κεφάλαιο 2).

Παράδειγμα 1.7 (Νόρμες)**.** Να υπολογιστούν οι νόρμες 1, 2 και ∞ του $\mathbf{x} = (3, -4, 12)^\top$ και οι νόρμες 1, ∞, 2, Frobenius και ο δείκτης κατάστασης του $A = \begin{pmatrix} 1 & -2 \\ 3 & 4 \end{pmatrix}$.

Λύση. $\|\mathbf{x}\|_1 = 3+4+12 = 19$, $\|\mathbf{x}\|_2 = \sqrt{9+16+144} = 13$, $\|\mathbf{x}\|_\infty = 12$. Για τον A: αθροίσματα στηλών 4 και 6, άρα $\|A\|_1 = 6$· αθροίσματα γραμμών 3 και 7, άρα $\|A\|_\infty = 7$· $\|A\|_F = \sqrt{1+4+9+16} = \sqrt{30} = 5.4772$. Επίσης $A^\top A = \begin{pmatrix} 10 & 10 \\ 10 & 20 \end{pmatrix}$ με ιδιοτιμές $15 \pm \sqrt{125}$, άρα $\|A\|_2 = \sqrt{15+\sqrt{125}} = 5.1167$. Τέλος $A^{-1} = \frac{1}{10}\begin{pmatrix} 4 & 2 \\ -3 & 1 \end{pmatrix}$ με $\|A^{-1}\|_2 = \sqrt{15+\sqrt{125}}/10$, οπότε $\kappa_2(A) = (15+\sqrt{125})/10 = 2.6180$.

Παράθυρο εντολών `mat_norms.m`

```
>> x = [3; -4; 12];
>> [norm(x, 1), norm(x), norm(x, Inf)]
ans =
   19   13   12
>> A = [1 -2; 3 4];
>> norm(A, 1)                % largest column sum
ans = 6
>> norm(A, Inf)              % largest row sum
ans = 7
>> norm(A)                   % 2-norm
ans = 5.1167
>> sqrt(max(eig(A' * A)))    % the same: largest singular value
ans = 5.1167
>> norm(A, 'fro')            % Frobenius norm
ans = 5.4772
>> sqrt(sum(A(:).^2))
ans = 5.4772
>> cond(A)                   % condition number norm(A)*norm(inv(A))
ans = 2.6180
>> norm(A) * norm(inv(A))
ans = 2.6180
```

Συνάρτηση	Αποτέλεσμα
`size(A)`, `numel(A)`, `length(v)`	διαστάσεις, πλήθος στοιχείων, μήκος διανύσματος
`zeros`, `ones`, `eye`, `rand`, `diag`	ειδικοί πίνακες
`A'`, `A.'`	(συζυγής) ανάστροφος
`A * B`, `A .* B`, `A ^ k`, `A .^ k`	γινόμενο και δύναμη πινάκων ή ανά στοιχείο
`dot(u,v)`, `cross(u,v)`, `kron(A,B)`	εσωτερικό, διανυσματικό, Kronecker γινόμενο
`det(A)`, `inv(A)`, `rank(A)`, `trace(A)`	ορίζουσα, αντίστροφος, τάξη, ίχνος
`A \ b`	λύση του $A\mathbf{x} = \mathbf{b}$
`eig(A)`, `[V, D] = eig(A)`	ιδιοτιμές, ιδιοδιανύσματα
`poly(A)`, `roots(p)`	χαρακτηριστικό πολυώνυμο, ρίζες πολυωνύμου
`norm(x, p)`, `norm(A, p)`, `cond(A)`	νόρμες ($p = 1, 2,$`Inf`,`'fro'`), δείκτης κατάστασης
`triu(A)`, `tril(A)`	άνω και κάτω τριγωνικό μέρος
`sum`, `prod`, `cumsum`, `max`, `min`, `mean`, `sort`, `find`	στατιστικά και αναζήτηση
`reshape(A, m, n)`, `A(:)`, `repmat(A, m, n)`	αναδιάταξη, διάνυσμα στήλη, επανάληψη

Πίνακας 1.6: Συναρτήσεις για πίνακες και διανύσματα.

1.7 Είσοδος, έξοδος και γραφικές παραστάσεις

Η απλή εμφάνιση μιας τιμής γίνεται με τη `disp(x)` και η μορφοποιημένη με τη `fprintf`, που δέχεται μια συμβολοσειρά μορφής και τις τιμές. Αν δοθούν διανύσματα, η μορφή επαναλαμβάνεται μέχρι να εξαντληθούν τα στοιχεία· η `sprintf` δίνει το ίδιο αποτέλεσμα ως συμβολοσειρά.

Προσδιοριστής	Σημασία	Παράδειγμα για την τιμή π ή 42
`%d`	ακέραιος	`42`
`%8.4f`	σταθερή υποδιαστολή, πλάτος 8, 4 δεκαδικά	`  3.1416`
`%.3e`	εκθετική μορφή, 3 δεκαδικά	`3.142e+00`
`%g`	η συντομότερη από τις δύο	`3.14159`
`%s`	συμβολοσειρά	
`\n`	αλλαγή γραμμής	

Για αρχεία: η `fid = fopen('data.dat', 'w')` ανοίγει αρχείο για εγγραφή, η `fprintf(fid, ...)` γράφει σε αυτό και η `fclose(fid)` το κλείνει. Ένας πίνακας αριθμών αποθηκεύεται και διαβάζεται απλούστερα με τις `save('data.dat', 'A', '-ascii')` και `A = load('data.dat')`. Η `input` ζητά τιμή από τον χρήστη.

Παράδειγμα 1.8 (Συντελεστής διάχυσης: πίνακας τιμών, αρχείο και γράφημα)**.** Ο συντελεστής διάχυσης του άνθρακα στον ωστενίτη ακολουθεί τον νόμο Arrhenius

$$D(T) = D_0 \exp\!\Big(-\frac{Q}{RT}\Big), \qquad D_0 = 2{,}3 \times 10^{-5}\,\mathrm{m^2/s}, \quad Q = 148\,\mathrm{kJ/mol}, \quad R = 8{,}314\,\mathrm{J/(mol\,K)}.$$

Να υπολογιστεί ο D για $T = 900, 950, \ldots, 1100\,^\circ\mathrm{C}$, να τυπωθεί πίνακας τιμών, να αποθηκευτούν τα ζεύγη (T, D) σε αρχείο και να βρεθεί η πρώτη θερμοκρασία του πίνακα στην οποία ο D είναι τουλάχιστον διπλάσιος από την τιμή του στους $900\,^\circ\mathrm{C}$.

Λύση. Η θερμοκρασία στον νόμο Arrhenius είναι *απόλυτη*, $T(\mathrm{K}) = T(^\circ\mathrm{C}) + 273.15$. Ο υπολογισμός γίνεται για όλες τις θερμοκρασίες μαζί, με πράξεις ανά στοιχείο (`./`)· η `find(..., 1)` δίνει τη θέση του πρώτου στοιχείου που ικανοποιεί τη συνθήκη.

Octave/MATLAB `arrhenius.m`

```
% Chapter 1 example: diffusivity of carbon in austenite, D = D0*exp(-Q/(R*T))
R  = 8.314462618;                       % J/(mol K)
D0 = 2.3e-5;                            % m^2/s
Q  = 148e3;                             % J/mol

TC = 900:50:1100;                       % vector of temperatures, deg C
TK = TC + 273.15;                       % element-wise operations
D  = D0 * exp(-Q ./ (R * TK));

fid = fopen('arrhenius.dat', 'w');
for i = 1:numel(TC)
  fprintf('%6.0f  %8.2f  %.4e\n', TC(i), TK(i), D(i));
  fprintf(fid, '%.2f %.6e\n', TK(i), D(i));
end
fclose(fid);
fprintf('D doubles between 900 and %d C\n', TC(find(D >= 2 * D(1), 1)));
```

Έξοδος

```
   900   1173.15  5.9175e-12
   950   1223.15  1.1003e-11
  1000   1273.15  1.9486e-11
  1050   1323.15  3.3050e-11
  1100   1373.15  5.3941e-11
D doubles between 900 and 1000 C
```

Αύξηση της θερμοκρασίας κατά μόλις 100 °C υπερτριπλασιάζει τον συντελεστή διάχυσης. Επειδή $\ln D = \ln D_0 - (Q/R)\,(1/T)$, η γραφική παράσταση του $\ln D$ συναρτήσει του $1/T$ είναι ευθεία γραμμή με κλίση $-Q/R$ (Σχήμα 1.1). Το γράφημα σχεδιάζεται από το αρχείο δεδομένων με το παρακάτω πρόγραμμα. Η εντολή `plot(x, y)` σχεδιάζει σε γραμμικούς άξονες, ενώ οι εντολές `semilogy`, `semilogx` και `loglog` σε λογαριθμικούς. Με την εντολή `hold on` προστίθενται περισσότερες καμπύλες στο ίδιο γράφημα, και με την `print` το γράφημα αποθηκεύεται σε αρχείο.

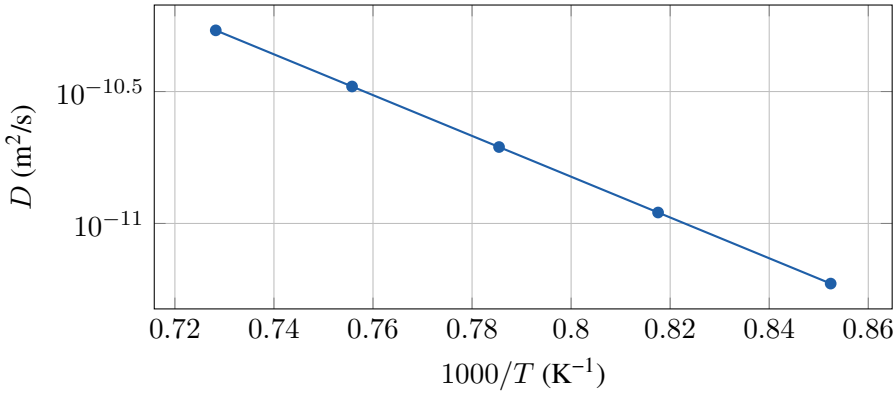
Octave/MATLAB `plot_arrhenius.m`

```
% Plot of the data written by the Arrhenius example: ln D versus 1/T
data = load('arrhenius.dat');           % columns: T (K), D (m^2/s)
T = data(:, 1);  D = data(:, 2);

figure;
semilogy(1000 ./ T, D, 'o-', 'LineWidth', 1.5);
xlabel('1000/T (1/K)');
ylabel('D (m^2/s)');
title('Diffusivity of C in austenite');
grid on;
print('-dpng', 'arrhenius.png');       % save the figure to a file
```

1.8 Αριθμητική κινητής υποδιαστολής

Οι πραγματικοί αριθμοί αποθηκεύονται στον υπολογιστή με πεπερασμένο πλήθος δυαδικών ψηφίων. Το Octave/MATLAB χρησιμοποιεί εξ ορισμού τον τύπο *διπλής ακρίβειας* (`double`) του προτύπου IEEE 754.



Σχήμα 1.1: Διάγραμμα Arrhenius για τη διάχυση του C στον ωστενίτη.

Αριθμοί διπλής ακρίβειας (IEEE 754)

Κάθε αριθμός έχει τη μορφή $x = \pm(1.b_1 b_2 \ldots b_{52})_2 \times 2^e$ με $-1022 \le e \le 1023$ και καταλαμβάνει 64 bits (1 πρόσημο, 11 εκθέτης, 52 κλασματικό μέρος).

- **Έψιλον μηχανής:** $\epsilon_{\text{mach}} = 2^{-52} \approx 2.22 \times 10^{-16}$, δηλαδή περίπου 16 σημαντικά δεκαδικά ψηφία.
- **Εύρος:** από 2.2×10^{-308} έως 1.8×10^{308}. Αποτέλεσμα μεγαλύτερο κατ' απόλυτη τιμή δίνει `Inf` (*υπερχείλιση*), πολύ μικρότερο δίνει 0 (*υποχείλιση*).
- **Στρογγύλευση:** το αποτέλεσμα κάθε πράξης είναι ο πλησιέστερος αριθμός μηχανής, δηλαδή $\text{fl}(x \circ y) = (x \circ y)(1+\delta)$ με $|\delta| \le \epsilon_{\text{mach}}/2$.

Στην απλή ακρίβεια (`single`) το κλασματικό μέρος έχει 23 bits και $\epsilon_{\text{mach}} = 2^{-23} \approx 1.19 \times 10^{-7}$.

Οι συνέπειες φαίνονται στο παρακάτω πρόγραμμα.

Octave/MATLAB `floats.m`

```
% Floating-point surprises (double precision)
fprintf('0.1 + 0.2          = %.17f\n', 0.1 + 0.2);
fprintf('0.1 + 0.2 == 0.3 ? = %d\n', (0.1 + 0.2) == 0.3);
s = 0;
for k = 1:10
  s = s + 0.1;
end
fprintf('ten times 0.1      = %.17f\n', s);
fprintf('(1e16 + 1) - 1e16  = %g\n', (1e16 + 1) - 1e16);
fprintf('largest double     = %.6e\n', realmax);
fprintf('smallest normal    = %.6e\n', realmin);
```

Έξοδος

```
0.1 + 0.2          = 0.30000000000000004
0.1 + 0.2 == 0.3 ? = 0
ten times 0.1      = 0.99999999999999989
(1e16 + 1) - 1e16  = 0
largest double     = 1.797693e+308
smallest normal    = 2.225074e-308
```

Από εδώ προκύπτουν τρεις πρακτικοί κανόνες, που θα συναντήσουμε σε όλα τα επόμενα κεφάλαια:

1. **Δεν συγκρίνουμε πραγματικούς αριθμούς με ==.** Αντί για `x == y` ελέγχουμε $|x - y| \le \varepsilon$ ή $|x - y| \le \varepsilon\,|y|$.
2. **Αποφεύγουμε την αφαίρεση σχεδόν ίσων αριθμών.** Η διαφορά τους έχει πολύ λιγότερα σωστά ψηφία από τους ίδιους (*καταστροφική απαλοιφή*).
3. **Η σειρά των πράξεων έχει σημασία.** Η πρόσθεση δεν είναι προσεταιριστική: $(a + b) + c \neq a + (b + c)$ εν γένει.

Παράδειγμα 1.9 (Σειρά Taylor της εκθετικής συνάρτησης)**.** Να γραφεί συνάρτηση που υπολογίζει το e^x από τη σειρά $e^x = \sum_{k=0}^{\infty} x^k/k!$, προσθέτοντας όρους μέχρι ο τελευταίος να μην αλλάζει πλέον το άθροισμα. Να ελεγχθεί η ακρίβεια για $x = 1, 10, -1, -10, -20, -30$.

Λύση. Κάθε όρος υπολογίζεται από τον προηγούμενο, $t_k = t_{k-1}\,x/k$, ώστε να αποφεύγονται τα παραγοντικά και οι μεγάλες δυνάμεις. Ο βρόχος `while` σταματά όταν $|t_k| \le \epsilon_{\text{mach}}|s|$. Η έξοδος δίνει κατά σειρά το x, το πλήθος των όρων, το άθροισμα, το σχετικό σφάλμα του, και το σχετικό σφάλμα της «ασφαλούς» εκδοχής που περιγράφεται παρακάτω.

Octave/MATLAB `myexp.m`

```
function [s, k] = myexp(x)
% MYEXP  exp(x) from the Taylor series; k = number of terms used.
  s = 1;  term = 1;  k = 0;
  while abs(term) > eps * abs(s)
    k = k + 1;
    term = term * x / k;                % x^k/k! from the previous term
    s = s + term;
  end
end
```

Octave/MATLAB `ex_myexp.m`

```
% Chapter 1 example: Taylor series of exp(x) and loss of accuracy for x < 0
for x = [1 10 -1 -10 -20 -30]
  [s, k] = myexp(x);
  if x < 0
    safe = 1 / myexp(-x);               % exp(x) = 1/exp(-x): no cancellation
  else
    safe = s;
  end
  fprintf('%4d  %3d  %14.6e  %9.2e  %9.2e\n', x, k, s, ...
          abs(s - exp(x)) / exp(x), abs(safe - exp(x)) / exp(x));
end
```

Έξοδος

```
   1   18    2.718282e+00   1.63e-16   1.63e-16
  10   46    2.202647e+04   3.30e-16   3.30e-16
  -1   19    3.678794e-01   3.02e-16   1.51e-16
 -10   58    4.539993e-05   3.07e-09   1.49e-16
 -20   95    5.621884e-09   1.73e+00   2.01e-16
 -30  118   -3.066812e-05   3.28e+08   4.05e-16
```

Για $x > 0$ το αποτέλεσμα είναι ακριβές σε όλα τα ψηφία. Για $x < 0$ όμως η ακρίβεια χάνεται: για $x = -20$ το αποτέλεσμα δεν έχει *κανένα* σωστό ψηφίο και για $x = -30$ είναι αρνητικό, ενώ μαθηματικά η σειρά συγκλίνει για κάθε x. Η αιτία είναι η καταστροφική απαλοιφή: οι όροι εναλλάσσουν πρόσημο και φτάνουν σε μέγεθος το $20^{20}/20! \approx 4 \times 10^7$, ενώ το άθροισμά τους είναι μόλις

$e^{-20} \approx 2 \times 10^{-9}$. Το σφάλμα στρογγύλευσης των μεγάλων όρων, περίπου $4 \times 10^{7} \cdot \epsilon_{\text{mach}} \approx 10^{-8}$, είναι μεγαλύτερο από το ίδιο το αποτέλεσμα. Η θεραπεία είναι απλή: για $x < 0$ υπολογίζουμε $e^{x} = 1/e^{-x}$, όπου όλοι οι όροι είναι θετικοί και δεν υπάρχει απαλοιφή (τελευταία στήλη).

Το παράδειγμα δείχνει τη βασική ιδέα της Αριθμητικής Ανάλυσης: ένας μαθηματικά σωστός τύπος δεν είναι απαραίτητα και καλός αλγόριθμος. Η ανάλυση ολοκληρώνεται στην Άσκηση 1.30.

1.9 Συνηθισμένα λάθη

- **Αρίθμηση από το 1.** Το `v(0)` δίνει σφάλμα· το πρώτο στοιχείο είναι το `v(1)` και το τελευταίο το `v(end)`. Ο μαθηματικός τύπος $\sum_{i=0}^{n} c_i x^i$ γράφεται με δείκτες `c(i+1)`.
- **Πράξεις πινάκων αντί για πράξεις ανά στοιχείο.** Το `x^2` και το `x*y` για διανύσματα είναι πράξεις πινάκων· ανά στοιχείο γράφουμε `x.^2`, `x.*y`, `1./x`.
- **Διάνυσμα γραμμή ή στήλη.** Το `[1 2 3]` είναι γραμμή και το `[1; 2; 3]` στήλη· στο γινόμενο πίνακα επί διάνυσμα χρειάζεται στήλη.
- **`=` αντί για `==`** μέσα σε συνθήκη.
- **Όνομα αρχείου διαφορετικό από το όνομα της συνάρτησης**, ή αρχείο εκτός του τρέχοντος φακέλου («undefined function»).
- **Μεταβλητή με όνομα ενσωματωμένης συνάρτησης**, π.χ. `sum = 0`· μετά από αυτό η `sum(v)` δεν λειτουργεί μέχρι να δοθεί `clear sum`. Το ίδιο ισχύει για τα `i`, `j`, `eps`, `pi`.
- **Αντιστροφή πίνακα για επίλυση συστήματος.** Γράφουμε `A \ b` και όχι `inv(A)*b`.
- **Μοίρες αντί για ακτίνια** στις τριγωνομετρικές συναρτήσεις.
- **Ατέρμονος βρόχος `while`**, όταν η συνθήκη δεν γίνεται ποτέ ψευδής· προσθέτουμε πάντα έναν μετρητή με μέγιστο πλήθος επαναλήψεων.

1.10 Ασκήσεις

Α. Βασικές ασκήσεις

Άσκηση 1.1 ★ (προτεραιότητα πράξεων)**.** Υπολογίστε πρώτα με το χέρι και μετά στο Octave τις παραστάσεις

$$2 + 3 \cdot 4^2/8, \qquad -2^2, \qquad (-2)^2, \qquad 7/2, \qquad \lfloor -7/2 \rfloor, \qquad -7 \bmod 3.$$

Ποιες συναρτήσεις δίνουν το ακέραιο πηλίκο και το υπόλοιπο, και σε τι διαφέρουν οι `floor` και `fix`, και οι `mod` και `rem`, για αρνητικούς αριθμούς;

Άσκηση 1.2 ★ (μετατροπή θερμοκρασιών)**.** Γράψτε συναρτήσεις που μετατρέπουν θερμοκρασία από βαθμούς Κελσίου σε Kelvin και σε Fahrenheit, με $F = \frac{9}{5}C + 32$. Τυπώστε πίνακα για τις τιμές $C = -40, -5, 30, 65, 100$. Σε ποια θερμοκρασία οι κλίμακες Κελσίου και Fahrenheit δίνουν τον ίδιο αριθμό;

Άσκηση 1.3 ★ (ρίζες τριωνύμου)**.** Γράψτε συνάρτηση που δέχεται τους συντελεστές a, b, c και τυπώνει τις ρίζες της $ax^2 + bx + c = 0$, διακρίνοντας τις περιπτώσεις δύο πραγματικών ριζών, διπλής ρίζας και συζυγών μιγαδικών ριζών. Δοκιμάστε τη στα $x^2 - 3x + 2$, $x^2 + 2x + 1$ και $x^2 + x + 1$. Τι πρέπει να γίνει όταν $a = 0$; Συγκρίνετε με τη `roots([a b c])`.

Άσκηση 1.4 ★ (βρόχοι `for` και `while`)**.** (α) Υπολογίστε με βρόχο `for` το άθροισμα $\sum_{k=1}^{n} k^2$ για $n = 10, 100, 1000$. Επαληθεύστε με τον τύπο $n(n+1)(2n+1)/6$, καθώς και με την εντολή `sum((1:n).^2)`, που δεν χρειάζεται βρόχο. (β) Βρείτε με βρόχο `while` το μικρότερο n για το οποίο το άθροισμα ξεπερνά το 10^6.

Άσκηση 1.5 ★ (παραγοντικό και υπερχείλιση)**.** Υπολογίστε το $n!$ με βρόχο. Ποιο είναι το μεγαλύτερο n για το οποίο το $n!$ χωρά (α) σε ακέραιο 32 bit, (β) σε ακέραιο 64 bit, (γ) σε αριθμό διπλής ακρίβειας; Τι δίνει το Octave για το $25!$ όταν ο βρόχος υπολογίζει το γινόμενο στον τύπο `int64` (με αρχική τιμή `f = int64(1)`) και τι με `double`;

Άσκηση 1.6 ★ (σειρά Leibniz)**.** Η σειρά $\pi = 4\sum_{k=0}^{\infty} \dfrac{(-1)^k}{2k+1}$ συγκλίνει πολύ αργά. Προσθέστε όρους με βρόχο `while` μέχρι ο επόμενος όρος να γίνει κατ' απόλυτη τιμή μικρότερος από 10^{-6}. Πόσοι όροι χρειάστηκαν και ποιο είναι το πραγματικό σφάλμα;

Άσκηση 1.7 ★ (switch: μετατροπή μονάδων τάσης)**.** Γράψτε συνάρτηση `to_mpa(value, unit)` που μετατρέπει μια τάση σε MPa από τις μονάδες Pa, kPa, MPa, GPa, psi και ksi ($1\,\text{psi} = 6894{,}757\,\text{Pa}$), χρησιμοποιώντας `switch`. Μετατρέψτε τα 30 ksi, 0,2 GPa και 14,7 psi. Τι πρέπει να κάνει η συνάρτηση όταν η μονάδα είναι άγνωστη;

Άσκηση 1.8 ★ (διανύσματα)**.** Δημιουργήστε το διάνυσμα $x_i = 0, 0.1, \dots, 1$ και το $y_i = x_i^2$. Υπολογίστε τη μέση και τη μέγιστη τιμή των y_i και το άθροισμα $\sum_i \frac{h}{2}(y_i + y_{i+1})$ με $h = 0.1$, που προσεγγίζει το $\int_0^1 x^2\,\mathrm{d}x$ (κανόνας τραπεζίου, Κεφάλαιο 6), χωρίς βρόχους.

Άσκηση 1.9 ★ (πράξεις πινάκων με το χέρι)**.** Έστω

$$A = \begin{pmatrix} 1 & 2 \\ 0 & -1 \\ 3 & 1 \end{pmatrix}, \qquad B = \begin{pmatrix} 2 & -1 & 1 \\ 0 & 1 & 4 \end{pmatrix}, \qquad \mathbf{u} = \begin{pmatrix} 1 \\ -1 \\ 2 \end{pmatrix}, \qquad \mathbf{v} = \begin{pmatrix} 2 \\ 0 \\ 1 \end{pmatrix}.$$

(α) Ποιες από τις πράξεις AB, BA, $A + B$, $A + B^{\top}$, $A\mathbf{u}$, $B\mathbf{u}$ ορίζονται και ποιες είναι οι διαστάσεις του αποτελέσματος; Υπολογίστε τα AB και BA. (β) Υπολογίστε τα $\mathbf{u}^{\top}\mathbf{v}$, $\mathbf{u}\mathbf{v}^{\top}$ και $\mathbf{u} \times \mathbf{v}$ και ελέγξτε ότι το $\mathbf{u} \times \mathbf{v}$ είναι κάθετο στα $\mathbf{u}$ και $\mathbf{v}$. (γ) Επαληθεύστε στο Octave τα αποτελέσματα και τη σχέση $(AB)^{\top} = B^{\top}A^{\top}$. Τι δίνουν τα `A .* B` και `A .* B'`;

Άσκηση 1.10 ★ (ορίζουσα και αντίστροφος)**.** Για τον πίνακα $C = \begin{pmatrix} 1 & 2 & 0 \\ 2 & 5 & 1 \\ 0 & 1 & 3 \end{pmatrix}$: (α) υπολογίστε την ορίζουσα με ανάπτυγμα κατά την πρώτη γραμμή και κατά την τρίτη στήλη· (β) υπολογίστε τον αντίστροφο από τον προσαρτημένο πίνακα· (γ) λύστε το σύστημα $C\mathbf{x} = (1, 1, 1)^{\top}$ και επαληθεύστε με τις `det`, `inv` και `C \ b`. Γιατί ο C^{-1} είναι συμμετρικός;

Άσκηση 1.11 ★ (ιδιοτιμές με το χέρι)**.** Βρείτε τις ιδιοτιμές (και, όπου ζητείται, τα ιδιοδιανύσματα) των πινάκων

$$\text{(α)}\ \begin{pmatrix} 2 & 0 & 0 \\ 1 & 3 & 0 \\ 4 & -1 & 5 \end{pmatrix}, \qquad \text{(β)}\ \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix}\ \text{(και ιδιοδιανύσματα)}, \qquad \text{(γ)}\ \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}.$$

Ελέγξτε σε κάθε περίπτωση ότι το άθροισμα και το γινόμενο των ιδιοτιμών ισούνται με το ίχνος και την ορίζουσα, και επαληθεύστε με την `eig`. Ποια είναι η γεωμετρική σημασία του πίνακα (γ) και γιατί δεν έχει πραγματικές ιδιοτιμές;

Άσκηση 1.12 ★ (νόρμες)**.** (α) Υπολογίστε τις νόρμες 1, 2 και ∞ του $\mathbf{x} = (1, -2, 2, -4)^{\top}$. (β) Για τον $M = \begin{pmatrix} 2 & -1 \\ -3 & 4 \end{pmatrix}$ υπολογίστε με το χέρι τις $\|M\|_1$, $\|M\|_{\infty}$, $\|M\|_F$ και την $\|M\|_2$ από τις ιδιοτιμές του $M^{\top}M$, και επαληθεύστε με τη `norm`. (γ) Ελέγξτε τις ανισότητες $\|M\|_2 \le \|M\|_F$ και $\|M\|_2 \le \sqrt{\|M\|_1 \|M\|_{\infty}}$.

Β. Προγραμματιστικές ασκήσεις

Άσκηση 1.13 ★★ (σχήμα Horner)**.** Το πολυώνυμο $p(x) = c_0 x^n + c_1 x^{n-1} + \cdots + c_n$ υπολογίζεται με μόνο n πολλαπλασιασμούς ως $p(x) = (\cdots((c_0 x + c_1)x + c_2)x + \cdots)x + c_n$. Γράψτε συνάρτηση `horner(c, x)` και υπολογίστε τα $p(1.5)$ και $p(-2)$ για $p(x) = 2x^4 - 3x^3 + 5x - 7$· συγκρίνετε με την ενσωματωμένη `polyval`, που χρησιμοποιεί την ίδια σειρά συντελεστών. Πόσους πολλαπλασιασμούς απαιτεί ο «αφελής» υπολογισμός όρο προς όρο; Επεκτείνετε τη συνάρτηση ώστε να επιστρέφει και την $p'(x)$.

Άσκηση 1.14 ★★ (ημίτονο από τη σειρά Taylor)**.** Κατά το Παράδειγμα 1.9, γράψτε συνάρτηση που υπολογίζει το $\sin x$ από τη σειρά Taylor. Δείξτε πειραματικά ότι για $x = 30$ η ακρίβεια χάνεται και διορθώστε το πρόβλημα ανάγοντας πρώτα το όρισμα στο $[-\pi, \pi]$.

Άσκηση 1.15 ★★ (αλγόριθμος του Ευκλείδη)**.** Γράψτε συνάρτηση που υπολογίζει τον μέγιστο κοινό διαιρέτη δύο θετικών ακεραίων με τον αλγόριθμο του Ευκλείδη (βρόχος `while` και υπόλοιπο διαίρεσης) και από αυτόν το ελάχιστο κοινό πολλαπλάσιο. Εφαρμόστε τη στους 1071 και 462 και συγκρίνετε με τις ενσωματωμένες `gcd` και `lcm`.

Άσκηση 1.16 ★★ (γινόμενο πινάκων)**.** Γράψτε συνάρτηση που υπολογίζει το γινόμενο $C = AB$ δύο πινάκων $n \times n$ με τρεις φωλιασμένους βρόχους, $c_{ij} = \sum_{k} a_{ik} b_{kj}$. Ελέγξτε τη σε πίνακες 3×3 και μετρήστε με τις `tic`/`toc` τον χρόνο για $n = 50, 100, 200$. Συγκρίνετε με τον ενσωματωμένο πολλαπλασιασμό `A*B` και εξηγήστε τη διαφορά. Πώς αυξάνει ο χρόνος με το n;

Άσκηση 1.17 ★★ (ανάγνωση αρχείου)**.** Το αρχείο `arrhenius.dat` του Παραδείγματος 1.8 περιέχει ζεύγη (T, D). Γράψτε πρόγραμμα που το διαβάζει *χωρίς να γνωρίζει εκ των προτέρων το πλήθος των γραμμών* και τυπώνει το πλήθος των σημείων, καθώς και την ελάχιστη, τη μέγιστη και τη μέση τιμή του D: (α) με τη `load`, (β) γραμμή προς γραμμή με τις `fopen`, `fgetl` και `sscanf`.

Άσκηση 1.18 ★★ (αριθμοί Fibonacci)**.** Υπολογίστε τους αριθμούς Fibonacci $F_1 = F_2 = 1$, $F_{n+1} = F_n + F_{n-1}$. (α) Δείξτε αριθμητικά ότι $F_{n+1}/F_n \to \varphi = (1 + \sqrt{5})/2$ και καταγράψτε το σφάλμα για $n = 10, 20, 30, 40$. (β) Ποιος είναι ο μεγαλύτερος F_n που χωρά σε ακέραιο 32 και 64 bit; Τι συμβαίνει όταν οι F_n υπολογίζονται με τον τύπο `int32` και το όριο ξεπεραστεί;

Άσκηση 1.19 ★★ (μέθοδος Monte Carlo)**.** Εκτιμήστε το π ρίχνοντας N τυχαία σημεία στο τετράγωνο $[0,1]^2$ και μετρώντας όσα πέφτουν μέσα στο τεταρτοκύκλιο $x^2 + y^2 \le 1$. Καταγράψτε το σφάλμα για $N = 10^2, 10^3, \ldots, 10^7$ και δείξτε σε διάγραμμα λογαριθμικών αξόνων ότι μειώνεται ως $N^{-1/2}$. Γράψτε τον υπολογισμό χωρίς βρόχο, με τη `rand(N, 1)` και λογικό δείκτη.

Άσκηση 1.20 ★★ (γραφική παράσταση)**.** Σχεδιάστε τη συνάρτηση $f(x) = \sin x / x$ στο $[-10, 10]$ με 401 σημεία. Τι τιμή παίρνει το πρόγραμμά σας στο $x = 0$ και πώς διορθώνεται; Προσθέστε τίτλο, ονομασίες αξόνων και πλέγμα και αποθηκεύστε το γράφημα σε αρχείο PNG.

Άσκηση 1.21 ★★ (ορίζουσα με ανάπτυγμα)**.** Γράψτε *αναδρομική* συνάρτηση `[d, calls] = detcof(A)` που υπολογίζει την ορίζουσα με ανάπτυγμα κατά την πρώτη γραμμή και μετρά πόσες φορές καλείται. (α) Ελέγξτε τη στον πίνακα του Παραδείγματος 1.3 και στον `magic(4)`. Γιατί η `detcof` δίνει ακριβώς 0 για τον `magic(4)`, ενώ η `det` όχι; (β) Για τους πίνακες `pascal(n)`, που έχουν ορίζουσα 1, καταγράψτε το πλήθος των κλήσεων για $n = 4, \ldots, 9$ και συγκρίνετε με το $n!$. Δείξτε ότι το πλήθος c_n ικανοποιεί $c_n = 1 + n\,c_{n-1}$ και εκτιμήστε τον χρόνο για $n = 20$, αν μία κλήση διαρκεί $1\,\mu\text{s}$.

Άσκηση 1.22 ★★ (ιδιότητες πράξεων πινάκων)**.** Για τους πίνακες

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

γράψτε πρόγραμμα που ελέγχει αριθμητικά τις σχέσεις $AB \neq BA$, $(AB)^{\top} = B^{\top} A^{\top}$, $\det(AB) =$

$\det A \det B$, $(AB)^{-1} = B^{-1}A^{-1}$, $\operatorname{tr}(AB) = \operatorname{tr}(BA)$, ότι οι A και $A^{\top}$ έχουν τις ίδιες ιδιοτιμές και ότι $\sum \lambda_i = \operatorname{tr} A$. Πότε μπορεί ο έλεγχος να γίνει με `isequal` ή `==` και πότε χρειάζεται ανοχή; Γιατί ο B^{-1} έχει ακέραια στοιχεία;

Άσκηση 1.23 ★★ (γωνίες, προβολές, εμβαδά και όγκοι)**.** Για τα διανύσματα $\mathbf{u} = (1, 2, 2)$, $\mathbf{v} = (3, 0, 4)$ και $\mathbf{w} = (0, 1, -1)$ υπολογίστε: (α) τη γωνία των $\mathbf{u}$ και $\mathbf{v}$, (β) την προβολή του $\mathbf{v}$ στη διεύθυνση του $\mathbf{u}$, $\frac{\mathbf{u}^{\top}\mathbf{v}}{\mathbf{u}^{\top}\mathbf{u}}\,\mathbf{u}$, (γ) το εμβαδόν του παραλληλογράμμου με πλευρές $\mathbf{u}, \mathbf{v}$, $\|\mathbf{u} \times \mathbf{v}\|_2$, και (δ) τον όγκο του παραλληλεπιπέδου με ακμές $\mathbf{u}, \mathbf{v}, \mathbf{w}$, που ισούται με $|\det[\mathbf{u}\ \mathbf{v}\ \mathbf{w}]|$. Γράψτε τα αποτελέσματα ως συναρτήσεις που δέχονται οποιαδήποτε διανύσματα του $\mathbb{R}^3$.

Γ. Προχωρημένα προβλήματα: αριθμητική κινητής υποδιαστολής

Άσκηση 1.24 ★★★ (έψιλον μηχανής)**.** (α) Εξηγήστε γιατί ο βρόχος του προγράμματος `macheps` τερματίζει ακριβώς στο 2^{-52}, αν και $1 + 2^{-53} > 1$ στους πραγματικούς αριθμούς. (β) Επαναλάβετε τον υπολογισμό σε απλή ακρίβεια. (γ) Βρείτε με βρόχο τον μικρότερο θετικό αριθμό διπλής ακρίβειας και εξηγήστε γιατί είναι πολύ μικρότερος από το 2.2×10^{-308} του πίνακα.

Άσκηση 1.25 ★★★ (ο αριθμός 0.1)**.** (α) Δείξτε ότι το 0.1 δεν έχει πεπερασμένη δυαδική αναπαράσταση και βρείτε το δυαδικό του ανάπτυγμα. (β) Σε μια προσομοίωση ο χρόνος ενημερώνεται ως `t = t + 0.1` για 10^6 βήματα. Πόσο διαφέρει το τελικό `t` από το 10^5; Συγκρίνετε με το `t = k*0.1` και εξηγήστε γιατί το σφάλμα είναι πολύ μεγαλύτερο από 10^6 φορές το σφάλμα αναπαράστασης του 0.1.

Άσκηση 1.26 ★★★ (ευσταθής επίλυση τριωνύμου)**.** Η εξίσωση $x^2 - 10^8 x + 1 = 0$ έχει ρίζες $x_1 \approx 10^8$ και $x_2 \approx 10^{-8}$. (α) Υπολογίστε τη μικρή ρίζα με τον γνωστό τύπο. Πόσα ψηφία είναι σωστά; (β) Εξηγήστε το φαινόμενο και δείξτε ότι οι τύποι

$$q = -\tfrac{1}{2}\big(b + \operatorname{sgn}(b)\sqrt{b^2 - 4ac}\big), \qquad x_1 = \frac{q}{a}, \qquad x_2 = \frac{c}{q}$$

δεν περιέχουν αφαίρεση σχεδόν ίσων αριθμών. (γ) Υλοποιήστε τους και ενσωματώστε τους στη συνάρτηση της Άσκησης 1.3.

Άσκηση 1.27 ★★★ (απαλοιφή σε συνάρτηση)**.** Η $f(x) = (1 - \cos x)/x^2$ τείνει στο $1/2$ όταν $x \to 0$. Υπολογίστε την για $x = 10^{-1}, 10^{-2}, \dots, 10^{-8}$. Εξηγήστε τα αποτελέσματα, εκτιμήστε το σχετικό σφάλμα ως συνάρτηση του x και γράψτε την f σε ισοδύναμη μορφή που υπολογίζεται με πλήρη ακρίβεια.

Άσκηση 1.28 ★★★ (ασταθής αναδρομή)**.** Έστω $I_n = \displaystyle\int_0^1 x^n e^{x-1}\,\mathrm{d}x$, $n = 0, 1, 2, \dots$

(α′) Δείξτε ότι $I_0 = 1 - 1/e$, $I_n = 1 - n\,I_{n-1}$ και $0 < I_n < 1/(n+1)$.

(β′) Υπολογίστε τα $I_1, \dots, I_{20}$ με την αναδρομή. Από ποιο n τα αποτελέσματα είναι προφανώς λανθασμένα;

(γ′) Δείξτε ότι ένα αρχικό σφάλμα δ_0 στο I_0 γίνεται $n!\,\delta_0$ στο I_n.

(δ′) Δείξτε ότι η αναδρομή προς τα πίσω, $I_{n-1} = (1 - I_n)/n$, είναι ευσταθής, και υπολογίστε το I_{20} ξεκινώντας από την πρόχειρη τιμή $I_{40} = 0$.

Άσκηση 1.29 ★★★ (σειρά άθροισης)**.** Υπολογίστε σε *απλή* ακρίβεια το $S_N = \sum_{k=1}^{N} 1/k$ για $N = 10^6$ (α) προσθέτοντας από $k = 1$ προς N, (β) από $k = N$ προς 1, και (γ) με την αντισταθμισμένη άθροιση του Kahan:

```
y = x - c; t = s + y; c = (t - s) - y; s = t;
```

Συγκρίνετε με το αποτέλεσμα διπλής ακρίβειας. Εξηγήστε γιατί η σειρά άθροισης έχει σημασία και τι αντιπροσωπεύει η μεταβλητή `c`.

Άσκηση 1.30 ★★★ (ανάλυση του Παραδείγματος 1.9)**.** (α) Δείξτε ότι ο μεγαλύτερος όρος της σειράς του e^x εμφανίζεται για $k \approx |x|$ και εκτιμήστε το μέγεθός του με τον τύπο του Stirling. (β) Συμπεράνετε ότι για $x < 0$ το σχετικό σφάλμα του αθροίσματος είναι περίπου $\epsilon_{\text{mach}}\, e^{2|x|}/\sqrt{2\pi|x|}$ και συγκρίνετε με τα αποτελέσματα του παραδείγματος. (γ) Για ποιο $x < 0$ χάνονται όλα τα σημαντικά ψηφία;

Άσκηση 1.31 ★★★ (πολυώνυμο σε ανεπτυγμένη μορφή)**.** Υπολογίστε το $p(x) = (x-1)^6$ σε 201 ισαπέχοντα σημεία του $[0.995, 1.005]$ με δύο τρόπους: απευθείας και από την ανεπτυγμένη μορφή $x^6 - 6x^5 + 15x^4 - 20x^3 + 15x^2 - 6x + 1$ (με το σχήμα Horner της Άσκησης 1.13). Σχεδιάστε τα δύο αποτελέσματα. Εκτιμήστε θεωρητικά το μέγεθος του «θορύβου» και εξηγήστε τι συνέπειες έχει για την εύρεση της ρίζας $x = 1$ με τις μεθόδους του Κεφαλαίου 4.

1.11 Απαντήσεις επιλεγμένων ασκήσεων

1.1 $8, -4, 4, 3.5, -4, 2\cdot$ `floor(-7/2)` $= -4$, `fix(-7/2)` $= -3$, `mod(-7,3)` $= 2$, `rem(-7,3)` $= -1$.

1.2 Στους -40 βαθμούς.

1.4 $385, 338350, 333833500\cdot$ $n = 144$.

1.5 (α) $12!$, (β) $20!$, (γ) $170! \approx 7.26 \times 10^{306}\cdot$ με `int64` το $25!$ «κολλά» στο $2^{63} - 1 \approx 9.22 \times 10^{18}$.

1.6 2×10^6 όροι, σφάλμα 5×10^{-7}.

1.7 206.84, 200 και $0{,}1014$ MPa.

1.8 Μέση τιμή 0.35, μέγιστη 1, τραπέζιο 0.335.

1.9 $AB = \begin{pmatrix} 2 & 1 & 9 \\ 0 & -1 & -4 \\ 6 & -2 & 7 \end{pmatrix}$, $BA = \begin{pmatrix} 5 & 6 \\ 12 & 3 \end{pmatrix}$, $\mathbf{u}^{\top}\mathbf{v} = 4$, $\mathbf{u} \times \mathbf{v} = (-1, 3, 2)$.

1.10 $\det C = 2$, $C^{-1} = \begin{pmatrix} 7 & -3 & 1 \\ -3 & 1.5 & -0.5 \\ 1 & -0.5 & 0.5 \end{pmatrix}$, $\mathbf{x} = (5, -2, 1)^{\top}$.

1.11 (α) $2, 3, 5\cdot$ (β) 3 με $(1, 1)$ και -1 με $(1, -1)\cdot$ (γ) $\pm i$.

1.12 (α) $9, 5, 4\cdot$ (β) $5, 7$, $\sqrt{30} = 5.4772$, $\sqrt{15 + \sqrt{200}} = 5.3983$.

1.13 $p(1.5) = 0.5$, $p(-2) = 39$.

1.15 ΜΚΔ $= 21$, ΕΚΠ $= 23562$.

1.18 F_{46} και F_{92}.

1.21 $c_4 = 41$, $c_9 = 623530\cdot$ για $n = 20$ περίπου $e \cdot 20!\,\mu\text{s} \approx 2 \times 10^5$ χρόνια.

1.23 $42.83°$, $(1.2222, 2.4444, 2.4444)$, $\sqrt{104} = 10.198$, 8.

1.26 Ο γνωστός τύπος δίνει 7.45×10^{-9} (σφάλμα 25%).

1.28 $I_{20} = 0.0455448841$.

1.12 Πλήρεις λύσεις

Τα προγράμματα των λύσεων έχουν πρόθεμα `sol_`. Κάθε πρόγραμμα καλύπτει μια ομάδα ασκήσεων, παρατίθεται μετά τις αντίστοιχες λύσεις και χρησιμοποιεί συναρτήσεις που βρίσκονται σε δικά τους αρχεία στον ίδιο φάκελο. Για τις Ασκήσεις 1.14, 1.16, 1.17, 1.19, 1.20 και 1.31 δίνονται υποδείξεις στο τέλος.

1.12.1 Ομάδα Α

Λύση της Άσκησης 1.1. Με βάση την προτεραιότητα των πράξεων (δύναμη, μετά πολλαπλασιασμός και διαίρεση από αριστερά προς τα δεξιά, μετά πρόσθεση):

$$2 + 3 \cdot 4^2/8 = 2 + 3 \cdot 16/8 = 2 + 48/8 = 2 + 6 = 8.$$

Στο -2^2 η δύναμη προηγείται του προσήμου, άρα $-(2^2) = -4$, ενώ $(-2)^2 = 4$. Η διαίρεση `7/2` δίνει 3.5, αφού όλοι οι αριθμοί είναι πραγματικοί. Το ακέραιο πηλίκο και το υπόλοιπο δίνονται από δύο ζεύγη συναρτήσεων:

	στρογγύλευση προς το $-\infty$	στρογγύλευση προς το μηδέν
πηλίκο	`floor(-7/2)` $= -4$	`fix(-7/2)` $= -3$
υπόλοιπο	`mod(-7,3)` $= 2$ (πρόσημο διαιρέτη)	`rem(-7,3)` $= -1$ (πρόσημο διαιρετέου)

Ισχύουν οι ταυτότητες $a = b \cdot \texttt{floor}(a/b) + \texttt{mod}(a,b)$ και $a = b \cdot \texttt{fix}(a/b) + \texttt{rem}(a,b)$. Για θετικούς αριθμούς τα δύο ζεύγη συμπίπτουν. Η `mod` είναι συνήθως η κατάλληλη για «κυκλικούς» δείκτες, π.χ. `mod(k-1, n) + 1`, που δίνει πάντα τιμή στο $1, \ldots, n$.

Λύση της Άσκησης 1.2. $K = C + 273.15$ και $F = \frac{9}{5}C + 32$. Οι δύο κλίμακες συμπίπτουν όταν $C = \frac{9}{5}C + 32$, δηλαδή $C = -40$, όπως επιβεβαιώνει η πρώτη γραμμή του πίνακα. Οι συναρτήσεις μπορούν να δεχτούν και διάνυσμα θερμοκρασιών, αφού όλες οι πράξεις τους ορίζονται ανά στοιχείο.

Λύση της Άσκησης 1.3. Με $d = b^2 - 4ac$: αν $d > 0$ οι ρίζες είναι $(-b \pm \sqrt{d})/(2a)$· αν $d = 0$ η διπλή ρίζα είναι $-b/(2a)$· αν $d < 0$ οι ρίζες είναι $-b/(2a) \pm i\sqrt{-d}/(2a)$. Για τα τρία τριώνυμα: 1 και 2· -1 (διπλή)· $-0.5 \pm 0.866\,i$. Όταν $a = 0$ η εξίσωση είναι πρώτου βαθμού ($x = -c/b$ αν $b \neq 0$) και ο τύπος θα οδηγούσε σε διαίρεση με το μηδέν· η περίπτωση πρέπει να ελέγχεται χωριστά. Ο έλεγχος `d == 0` είναι αξιόπιστος μόνο όταν οι συντελεστές είναι μικροί ακέραιοι· γενικά, λόγω στρογγύλευσης, η διπλή ρίζα αναγνωρίζεται με ανοχή. Η `roots([1 1 1])` δίνει τις ίδιες μιγαδικές ρίζες (η `sqrt` αρνητικού αριθμού δίνει στο Octave μιγαδικό αποτέλεσμα). Για την ακρίβεια του τύπου βλ. Άσκηση 1.26.

Λύση της Άσκησης 1.4. (α) 385, 338350 και 333833500, σε συμφωνία με τον τύπο και με το `sum((1:n).^2)`. (β) Ο βρόχος `while` σταματά στο $n = 144$, με άθροισμα 1005720. Από τον τύπο, $n^3/3 \approx 10^6$ δίνει $n \approx 144.2$.

Λύση της Άσκησης 1.5. (α) $12! = 479001600 < 2^{31} - 1 = 2147483647 < 13!$. (β) $20! \approx 2.43 \times 10^{18} < 2^{63} - 1 \approx 9.22 \times 10^{18} < 21!$. (γ) $170! \approx 7.26 \times 10^{306}$, ενώ $171! > 1.8 \times 10^{308}$ δίνει `Inf`. Προσοχή όμως: ήδη από το 23! το αποτέλεσμα σε διπλή ακρίβεια είναι *προσεγγιστικό*, αφού έχει περισσότερα από 53 σημαντικά δυαδικά ψηφία. Για το 25!: με `double` τυπώνεται 1.551121×10^{25} (σωστά τα πρώτα 16 ψηφία), ενώ με `int64` το γινόμενο «κολλά» στο `intmax('int64')` $= 9223372036854775807$ από το 21! και μετά, *χωρίς μήνυμα σφάλματος*. Οι ακέραιοι τύποι του Octave/MATLAB κάνουν πράξεις *κορεσμού* (saturation).

Λύση της Άσκησης 1.6. Ο όρος $4/(2k+1)$ γίνεται μικρότερος από 10^{-6} όταν $k > 2 \times 10^6$, άρα προστίθενται 2×10^6 όροι. Το σφάλμα είναι 5.0×10^{-7}, δηλαδή το *μισό* του πρώτου όρου που παραλείπεται, όπως αναμένεται για εναλλασσόμενη σειρά με αργά φθίνοντες όρους. Για κάθε επιπλέον δεκαδικό ψηφίο απαιτούνται δεκαπλάσιοι όροι: η σειρά είναι ακατάλληλη για πρακτικό υπολογισμό του π.

Λύση της Άσκησης 1.7. Η επιλογή γίνεται απευθείας πάνω στη συμβολοσειρά της μονάδας (`case 'ksi'`). Αποτελέσματα: 30 ksi $=$ 206,84 MPa, 0,2 GPa $=$ 200 MPa, 14,7 psi $=$ 0,1014 MPa (η ατμοσφαιρική πίεση). Για άγνωστη μονάδα η συνάρτηση *δεν* πρέπει να επιστρέφει μια αυθαίρετη τιμή

(π.χ. μηδέν), που θα περνούσε απαρατήρητη στους επόμενους υπολογισμούς, αλλά να σταματά με μήνυμα: `otherwise, error(...)`.

Λύση της Άσκησης 1.8. Μέση τιμή $\frac{1}{11}\sum x_i^2 = 0.35$, μέγιστη 1 και άθροισμα τραπεζίων 0.335, έναντι της ακριβούς τιμής $\int_0^1 x^2\,\mathrm{d}x = 1/3$. Το άθροισμα γράφεται χωρίς βρόχο με τα υποδιανύσματα `y(1:end-1)` και `y(2:end)`. Το διάνυσμα δημιουργήθηκε ως `0.1*(0:10)` και όχι με διαδοχικές προσθέσεις του 0.1, για τον λόγο που εξηγείται στην Άσκηση 1.25.

Octave/MATLAB quadroots.m

```
function s = quadroots(a, b, c)
% QUADROOTS  Roots of a*x^2 + b*x + c = 0 as a text string.
  d = b * b - 4 * a * c;
  if d > 0
    s = sprintf('%g and %g', (-b - sqrt(d)) / (2 * a), (-b + sqrt(d)) / (2 * a));
  elseif d == 0
    s = sprintf('%g (double)', -b / (2 * a));
  else
    s = sprintf('%g +- %gi', -b / (2 * a), sqrt(-d) / (2 * a));
  end
end
```

Octave/MATLAB to_mpa.m

```
function v = to_mpa(value, unit)
% TO_MPA  Converts a stress from the given unit to MPa.
  switch unit
    case 'Pa',  v = value * 1e-6;
    case 'kPa', v = value * 1e-3;
    case 'MPa', v = value;
    case 'GPa', v = value * 1e3;
    case 'psi', v = value * 6.894757e-3;
    case 'ksi', v = value * 6.894757;
    otherwise,  error('to_mpa: unknown unit %s', unit);
  end
end
```

Octave/MATLAB sol_basic.m

```
% Solutions 1.1-1.8 (group A)
% uses the functions quadroots.m and to_mpa.m

fprintf('--- 1.1\n');
fprintf('2 + 3*4^2/8 = %g\n', 2 + 3 * 4^2 / 8);
fprintf('-2^2 = %d,  (-2)^2 = %d\n', -2^2, (-2)^2);
fprintf('7/2 = %g,  floor(-7/2) = %d,  fix(-7/2) = %d\n', 7 / 2, floor(-7 / 2), fix(-7 /
    2));
fprintf('mod(-7,3) = %d,  rem(-7,3) = %d\n', mod(-7, 3), rem(-7, 3));

fprintf('--- 1.2\n');
for c = -40:35:100
  fprintf('%5d C = %7.2f K = %6.1f F\n', c, c + 273.15, 9 * c / 5 + 32);
end

fprintf('--- 1.3\n');
coef = [1 -3 2; 1 2 1; 1 1 1];
for i = 1:3
```

```
  fprintf('%dx^2 + (%d)x + %d: %s\n', coef(i, 1), coef(i, 2), coef(i, 3), ...
          quadroots(coef(i, 1), coef(i, 2), coef(i, 3)));
end

fprintf('--- 1.4\n');
for n = [10 100 1000]
  s = 0;
  for k = 1:n
    s = s + k * k;
  end
  fprintf('n = %4d: sum = %d, formula = %d\n', n, s, n * (n + 1) * (2 * n + 1) / 6);
end
s = 0;  n = 0;
while s <= 1e6
  n = n + 1;
  s = s + n * n;
end
fprintf('smallest n with sum > 1e6: %d (sum = %d)\n', n, s);

fprintf('--- 1.5\n');
f = 1;  n = 0;  n32 = 0;  n64 = 0;
while f * (n + 1) < Inf
  n = n + 1;
  f = f * n;
  if f <= 2^31 - 1, n32 = n; end
  if f <= 2^63 - 1, n64 = n; end
end
fprintf('int32: %d!,  int64: %d!,  double: %d! = %.4e\n', n32, n64, n, f);
f64 = int64(1);  fd = 1;
for k = 1:25
  f64 = f64 * k;                        % saturates at intmax('int64')
  fd = fd * k;
end
fprintf('25!: int64 %d,  double %.6e\n', f64, fd);

fprintf('--- 1.6\n');
s = 0;  k = 0;  term = 4;
while abs(term) > 1e-6
  s = s + term;
  k = k + 1;
  term = 4 * (-1)^k / (2 * k + 1);
end
fprintf('terms = %d, sum = %.8f, error = %.2e\n', k, s, abs(s - pi));

fprintf('--- 1.7\n');
fprintf('30 ksi = %.2f MPa,  0.2 GPa = %.2f MPa,  14.7 psi = %.4f MPa\n', ...
        to_mpa(30, 'ksi'), to_mpa(0.2, 'GPa'), to_mpa(14.7, 'psi'));

fprintf('--- 1.8\n');
x = 0.1 * (0:10);                       % vector 0, 0.1, ..., 1
y = x.^2;                               % element-wise power
trap = sum(0.1 * (y(1:end-1) + y(2:end)) / 2);
fprintf('mean = %.4f, max = %.4f, trapezoid = %.4f\n', mean(y), max(y), trap);
```

Έξοδος

```
--- 1.1
2 + 3*4^2/8 = 8
-2^2 = -4,  (-2)^2 = 4
7/2 = 3.5,  floor(-7/2) = -4,  fix(-7/2) = -3
mod(-7,3) = 2,  rem(-7,3) = -1
--- 1.2
  -40 C =  233.15 K =  -40.0 F
   -5 C =  268.15 K =   23.0 F
   30 C =  303.15 K =   86.0 F
   65 C =  338.15 K =  149.0 F
  100 C =  373.15 K =  212.0 F
--- 1.3
1x^2 + (-3)x + 2: 1 and 2
1x^2 + (2)x + 1: -1 (double)
1x^2 + (1)x + 1: -0.5 +- 0.866025i
--- 1.4
n =   10: sum = 385, formula = 385
n =  100: sum = 338350, formula = 338350
n = 1000: sum = 333833500, formula = 333833500
smallest n with sum > 1e6: 144 (sum = 1005720)
--- 1.5
int32: 12!,  int64: 20!,  double: 170! = 7.2574e+306
25!: int64 9223372036854775807,  double 1.551121e+25
--- 1.6
terms = 2000000, sum = 3.14159215, error = 5.00e-07
--- 1.7
30 ksi = 206.84 MPa,  0.2 GPa = 200.00 MPa,  14.7 psi = 0.1014 MPa
--- 1.8
mean = 0.3500, max = 1.0000, trapezoid = 0.3350
```

Λύση της Άσκησης 1.9. (α) Ο A είναι 3×2 και ο B 2×3. Ορίζονται: AB (3×3), BA (2×2), $A+B^{\top}$ (3×2) και $B\mathbf{u}$ (2×1). Δεν ορίζονται το $A+B$ (διαφορετικές διαστάσεις) και το $A\mathbf{u}$ (ο A έχει 2 στήλες, το $\mathbf{u}$ 3 στοιχεία). Π.χ. το στοιχείο $(1,3)$ του AB είναι η γραμμή $(1,2)$ του A επί τη στήλη $(1,4)$ του B, δηλαδή $1+8=9$:

$$AB=\begin{pmatrix}2 & 1 & 9\\ 0 & -1 & -4\\ 6 & -2 & 7\end{pmatrix},\qquad BA=\begin{pmatrix}5 & 6\\ 12 & 3\end{pmatrix}.$$

Τα δύο γινόμενα δεν έχουν καν τις ίδιες διαστάσεις.

(β) $\mathbf{u}^{\top}\mathbf{v}=2+0+2=4$,

$$\mathbf{u}\mathbf{v}^{\top}=\begin{pmatrix}2 & 0 & 1\\ -2 & 0 & -1\\ 4 & 0 & 2\end{pmatrix},\qquad \mathbf{u}\times\mathbf{v}=\big((-1)(1)-(2)(0),\,(2)(2)-(1)(1),\,(1)(0)-(-1)(2)\big)=(-1,3,2),$$

και $(\mathbf{u}\times\mathbf{v})^{\top}\mathbf{u}=-1-3+4=0$, $(\mathbf{u}\times\mathbf{v})^{\top}\mathbf{v}=-2+0+2=0$. Ο πίνακας $\mathbf{u}\mathbf{v}^{\top}$ έχει τάξη 1: όλες οι γραμμές του είναι πολλαπλάσια του $\mathbf{v}^{\top}$.

(γ) Το `A .* B` δίνει σφάλμα διαστάσεων (3×2 με 2×3), ενώ το `A .* B'` δίνει τον 3×2 πίνακα $\begin{pmatrix}2 & 0\\ 0 & -1\\ 3 & 4\end{pmatrix}$.

Λύση της Άσκησης 1.10. (α) Κατά την πρώτη γραμμή: $\det C=1(15-1)-2(6-0)+0=2$. Κατά την τρίτη στήλη, με πρόσημα $+,-,+$: $\det C=0\cdot(2-0)-1\cdot(1\cdot 1-2\cdot 0)+3\cdot(5-4)=-1+3=2$.

(β) Οι συμπαράγοντες δίνουν

$$\operatorname{adj} C = \begin{pmatrix} 14 & -6 & 2 \\ -6 & 3 & -1 \\ 2 & -1 & 1 \end{pmatrix}, \qquad C^{-1} = \tfrac{1}{2}\operatorname{adj} C = \begin{pmatrix} 7 & -3 & 1 \\ -3 & 1.5 & -0.5 \\ 1 & -0.5 & 0.5 \end{pmatrix}.$$

Ο C είναι συμμετρικός, άρα $(C^{-1})^\top = (C^\top)^{-1} = C^{-1}$: και ο αντίστροφος είναι συμμετρικός.

(γ) $\mathbf{x} = C^{-1}(1,1,1)^\top = (7-3+1,\ -3+1.5-0.5,\ 1-0.5+0.5)^\top = (5,-2,1)^\top$, και πράγματι $C\mathbf{x} = (5-4,\ 10-10+1,\ -2+3)^\top = (1,1,1)^\top$.

Λύση της Άσκησης 1.11. (α) Ο πίνακας είναι κάτω τριγωνικός, άρα $\det(\lambda I - A) = (\lambda-2)(\lambda-3)(\lambda-5)$ και οι ιδιοτιμές είναι τα διαγώνια στοιχεία $2, 3, 5$· άθροισμα $10 = \operatorname{tr} A$, γινόμενο $30 = \det A$.

(β) $p(\lambda) = \lambda^2 - 2\lambda - 3 = (\lambda-3)(\lambda+1)$. Για $\lambda = 3$: $-2v_1 + 2v_2 = 0$, άρα $(1,1)$· για $\lambda = -1$: $2v_1 + 2v_2 = 0$, άρα $(1,-1)$. Τα δύο ιδιοδιανύσματα είναι κάθετα, όπως αναμένεται για συμμετρικό πίνακα· άθροισμα 2, γινόμενο -3.

(γ) $p(\lambda) = \lambda^2 + 1$, άρα $\lambda = \pm i$, με άθροισμα $0 = \operatorname{tr} A$ και γινόμενο $1 = \det A$. Ο πίνακας είναι ο πίνακας στροφής κατά $90°$: κανένα μη μηδενικό πραγματικό διάνυσμα δεν μένει στην ίδια διεύθυνση μετά από στροφή $90°$, άρα δεν υπάρχουν πραγματικές ιδιοτιμές.

Λύση της Άσκησης 1.12. (α) $\|\mathbf{x}\|_1 = 1+2+2+4 = 9$, $\|\mathbf{x}\|_2 = \sqrt{1+4+4+16} = 5$, $\|\mathbf{x}\|_\infty = 4$.

(β) Αθροίσματα στηλών 5 και 5, άρα $\|M\|_1 = 5$· αθροίσματα γραμμών 3 και 7, άρα $\|M\|_\infty = 7$· $\|M\|_F = \sqrt{4+1+9+16} = \sqrt{30} = 5.4772$. Επίσης $M^\top M = \begin{pmatrix} 13 & -14 \\ -14 & 17 \end{pmatrix}$ με $\lambda = 15 \pm \sqrt{4+196} = 15 \pm \sqrt{200}$, άρα $\|M\|_2 = \sqrt{15+\sqrt{200}} = 5.3983$.

(γ) $5.3983 \le 5.4772$ και $5.3983 \le \sqrt{35} = 5.9161$. Γενικά $\|M\|_2 \le \|M\|_F \le \sqrt{n}\,\|M\|_2$ και $\|M\|_2^2 \le \|M\|_1 \|M\|_\infty$· οι νόρμες 1 και ∞, που υπολογίζονται πολύ εύκολα, δίνουν επομένως ένα φράγμα για τη νόρμα 2.

Octave/MATLAB `sol_matrix.m`

```
% Solutions 1.9–1.12 and 1.21–1.23 (matrices and vectors)
format compact
fprintf('--- 1.9\n');
A = [1 2; 0 -1; 3 1];  B = [2 -1 1; 0 1 4];
AB = A * B, BA = B * A
fprintf('(AB)'' = B''A'': %d\n', isequal((A * B)', B' * A'));
u = [1; -1; 2];  v = [2; 0; 1];
fprintf('u''v = %g,  u x v = (%g, %g, %g)\n', u' * v, cross(u, v));
uvT = u * v'

fprintf('--- 1.10\n');
C = [1 2 0; 2 5 1; 0 1 3];
fprintf('det(C) = %g\n', det(C));
adjC = det(C) * inv(C)
x = C \ [1; 1; 1]

fprintf('--- 1.11\n');
fprintf('eig: %s | %s | %s\n', mat2str(eig([2 0 0; 1 3 0; 4 -1 5])'), ...
        mat2str(eig([1 2; 2 1])'), mat2str(eig([0 -1; 1 0])'));

fprintf('--- 1.12\n');
x = [1; -2; 2; -4];
fprintf('x: %g %g %g\n', norm(x, 1), norm(x), norm(x, Inf));
M = [2 -1; -3 4];
fprintf('M: 1-norm %g, inf-norm %g, 2-norm %.4f, Frobenius %.4f\n', ...
        norm(M, 1), norm(M, Inf), norm(M), norm(M, 'fro'));
```

```
27
28 fprintf('--- 1.21\n');
29 A = [2 -1 0; 1 3 2; 0 1 4];
30 fprintf('A: detcof = %g, det = %g\n', detcof(A), det(A));
31 fprintf('magic(4): detcof = %g, det = %.4e\n', detcof(magic(4)), det(magic(4)));
32 for n = 4:9
33   [d, calls] = detcof(pascal(n));
34   fprintf('n = %d: detcof(pascal) = %g, calls = %6d, n! = %6d\n', n, d, calls,
       factorial(n));
35 end
36
37 fprintf('--- 1.22\n');
38 A = [4 -2 1; 3 6 -4; 2 1 8];  B = [1 2 3; 0 1 4; 5 6 0];
39 tol = 1e-12;
40 fprintf('det(B) = %g, inv(B) =\n', det(B));  disp(round(inv(B)));
41 fprintf('AB == BA                : %d\n', isequal(A * B, B * A));
42 fprintf('(AB)'' == B''A''           : %d\n', isequal((A * B)', B' * A'));
43 fprintf('det(AB) = det(A) det(B) : %d\n', abs(det(A * B) - det(A) * det(B)) < tol *
       abs(det(A * B)));
44 fprintf('inv(AB) = inv(B) inv(A) : %d\n', norm(inv(A * B) - inv(B) * inv(A)) < tol);
45 fprintf('trace(AB) = trace(BA)   : %d\n', trace(A * B) == trace(B * A));
46 fprintf('eig(A'') = eig(A)        : %d\n', norm(sort(eig(A')) - sort(eig(A))) < tol);
47 fprintf('sum(eig(A)) = trace(A)  : %d\n', abs(sum(eig(A)) - trace(A)) < tol);
48
49 fprintf('--- 1.23\n');
50 u = [1; 2; 2];  v = [3; 0; 4];  w = [0; 1; -1];
51 phi = acos(dot(u, v) / (norm(u) * norm(v)));
52 p = dot(u, v) / dot(u, u) * u;
53 fprintf('angle = %.2f deg, projection = (%.4f, %.4f, %.4f)\n', phi * 180 / pi, p);
54 fprintf('area = %.4f, volume = %g\n', norm(cross(u, v)), abs(det([u v w])));
```

Το πρόγραμμα καλύπτει και τις Ασκήσεις 1.21–1.23 της ομάδας Β, που λύνονται παρακάτω.

Έξοδος

```
--- 1.9
AB =
   2   1   9
   0  -1  -4
   6  -2   7
BA =
    5    6
   12    3
(AB)' = B'A': 1
u'v = 4,  u x v = (-1, 3, 2)
uvT =
   2   0   1
  -2   0  -1
   4   0   2
--- 1.10
det(C) = 2
adjC =
   14   -6    2
   -6    3   -1
    2   -1    1
x =
   5
  -2
   1
```

```
--- 1.11
eig: [5 3 2] | [-1 3] | [0-1i 0+1i]
--- 1.12
x: 9 5 4
M: 1-norm 5, inf-norm 7, 2-norm 5.3983, Frobenius 5.4772
--- 1.21
A: detcof = 24, det = 24
magic(4): detcof = 0, det = -1.4495e-12
n = 4: detcof(pascal) = 1, calls =     41, n! =     24
n = 5: detcof(pascal) = 1, calls =    206, n! =    120
n = 6: detcof(pascal) = 1, calls =   1237, n! =    720
n = 7: detcof(pascal) = 1, calls =   8660, n! =   5040
n = 8: detcof(pascal) = 1, calls =  69281, n! =  40320
n = 9: detcof(pascal) = 1, calls = 623530, n! = 362880
--- 1.22
det(B) = 1, inv(B) =
  -24   18    5
   20  -15   -4
   -5    4    1
AB == BA                : 0
(AB)' == B'A'           : 1
det(AB) = det(A) det(B) : 1
inv(AB) = inv(B) inv(A) : 1
trace(AB) = trace(BA)   : 1
eig(A') = eig(A)        : 1
sum(eig(A)) = trace(A)  : 1
--- 1.23
angle = 42.83 deg, projection = (1.2222, 2.4444, 2.4444)
area = 10.1980, volume = 8
```

1.12.2 Ομάδα Β

Λύση της Άσκησης 1.13. Ο αλγόριθμος είναι: $p \leftarrow c_0$ και, για $k = 1, \ldots, n$, $p \leftarrow p\,x + c_k$. Για το $p(x) = 2x^4 - 3x^3 + 5x - 7$ οι συντελεστές είναι $(2, -3, 0, 5, -7)$ (ο μηδενικός συντελεστής του x^2 δεν παραλείπεται) και για $x = 1.5$:

$$2 \;\to\; 2(1.5) - 3 = 0 \;\to\; 0(1.5) + 0 = 0 \;\to\; 0(1.5) + 5 = 5 \;\to\; 5(1.5) - 7 = 0.5.$$

Ομοίως $p(-2) = 39$. Η `polyval(c, x)` δίνει τα ίδια αποτελέσματα. Το σχήμα Horner απαιτεί $n = 4$ πολλαπλασιασμούς, ενώ ο υπολογισμός όρο προς όρο με επαναλαμβανόμενους πολλαπλασιασμούς απαιτεί $n(n+1)/2 = 10$. Για την παράγωγο αρκεί μια δεύτερη μεταβλητή που ενημερώνεται *πριν* από το p: $q \leftarrow q\,x + p$, $p \leftarrow p\,x + c_k$, με αρχική τιμή $q = 0$· στο τέλος $q = p'(x)$. Για $x = 1.5$ προκύπτει $p'(1.5) = 8(1.5)^3 - 9(1.5)^2 + 5 = 11.75$. Έτσι η Newton–Raphson για πολυώνυμα (Κεφάλαιο 4) κοστίζει μόνο $2n$ πολλαπλασιασμούς ανά βήμα.

Λύση της Άσκησης 1.15. Ο αλγόριθμος βασίζεται στην ιδιότητα $\text{ΜΚΔ}(a, b) = \text{ΜΚΔ}(b,\ a \bmod b)$ και τερματίζει όταν το υπόλοιπο μηδενιστεί:

$$(1071, 462) \to (462, 147) \to (147, 21) \to (21, 0),$$

άρα $\text{ΜΚΔ} = 21$ και $\text{ΕΚΠ} = 1071 \cdot 462/21 = 23562$, όπως δίνουν και οι `gcd` και `lcm`. Το πλήθος των βημάτων είναι το πολύ ανάλογο του πλήθους των ψηφίων του μικρότερου αριθμού.

Λύση της Άσκησης 1.18. (α) Το σφάλμα του λόγου μειώνεται γεωμετρικά, με παράγοντα περίπου $\varphi^{-2} \approx 0.38$ ανά βήμα: 1.5×10^{-4} ($n = 10$), 9.8×10^{-9} ($n = 20$), 6.5×10^{-13} ($n = 30$), και για

$n = 40$ έχει φτάσει στην ακρίβεια της μηχανής. (β) Ο μεγαλύτερος αριθμός Fibonacci που χωρά σε προσημασμένο ακέραιο 32 bit είναι ο $F_{46} = 1836311903$ και σε 64 bit ο $F_{92} \approx 7.54 \times 10^{18}$. Στη λύση οι F_n αποθηκεύονται ως `double`, ώστε να μπορούν να συγκριθούν με τα όρια χωρίς να τα ξεπεράσουν. Αν υπολογιστούν σε `int32`, οι F_{47} και F_{48} «κολλούν» στο $2^{31} - 1 = 2147483647$ (κορεσμός), χωρίς μήνυμα σφάλματος, και από εκεί και πέρα όλοι οι όροι είναι λανθασμένοι.

Octave/MATLAB `horner.m`

```
function [p, dp] = horner(c, x)
% HORNER  Value of c(1)*x^n + c(2)*x^(n-1) + ... + c(n+1) with n multiplications;
%   dp is the derivative p'(x).
  p = c(1);  dp = 0;
  for k = 2:numel(c)
    dp = dp * x + p;                    % update the derivative first
    p = p * x + c(k);
  end
end
```

Octave/MATLAB `mygcd.m`

```
function a = mygcd(a, b)
% MYGCD  Greatest common divisor by the Euclidean algorithm.
  while b ~= 0
    t = mod(a, b);  a = b;  b = t;
  end
end
```

Octave/MATLAB `sol_prog.m`

```
% Solutions 1.13, 1.15, 1.18 (group B)
% uses the functions horner.m and mygcd.m

fprintf('--- 1.13\n');
c = [2 -3 0 5 -7];
[p, dp] = horner(c, 1.5);
fprintf('p(1.5) = %g,  p(-2) = %g,  p''(1.5) = %g,  polyval: %g\n', ...
        p, horner(c, -2), dp, polyval(c, 1.5));

fprintf('--- 1.15\n');
g = mygcd(1071, 462);
fprintf('gcd(1071, 462) = %d,  lcm = %d,  built-in: %d %d\n', g, 1071 * 462 / g, ...
        gcd(1071, 462), lcm(1071, 462));

fprintf('--- 1.18\n');
phi = (1 + sqrt(5)) / 2;
a = 1;  b = 1;  n = 2;  n32 = 0;  n64 = 0;       % F_1, F_2 as doubles
while n < 100
  if b <= 2^31 - 1, n32 = n; end
  if b <= 2^63 - 1, n64 = n; end
  if any(n == [10 20 30 40])
    fprintf('n = %d: F(n+1)/F(n) - phi = %.3e\n', n, (a + b) / b - phi);
  end
  t = a + b;  a = b;  b = t;
  n = n + 1;
end
fprintf('largest n with F(n) in int32: %d,  in int64: %d\n', n32, n64);
a = int32(1);  b = int32(1);
```

```
for n = 3:48
  t = a + b;  a = b;  b = t;            % int32 arithmetic saturates
end
fprintf('F(47), F(48) computed in int32: %d %d\n', a, b);
```

Έξοδος

```
--- 1.13
p(1.5) = 0.5,  p(-2) = 39,  p'(1.5) = 11.75,  polyval: 0.5
--- 1.15
gcd(1071, 462) = 21,  lcm = 23562,  built-in: 21 23562
--- 1.18
n = 10: F(n+1)/F(n) - phi = 1.478e-04
n = 20: F(n+1)/F(n) - phi = 9.772e-09
n = 30: F(n+1)/F(n) - phi = 6.459e-13
n = 40: F(n+1)/F(n) - phi = 0.000e+00
largest n with F(n) in int32: 46,  in int64: 92
F(47), F(48) computed in int32: 2147483647 2147483647
```

Λύση της Άσκησης 1.21. (α) Η `detcof` δίνει 24 για τον πίνακα του Παραδείγματος 1.3 και ακριβώς 0 για τον `magic(4)`. Ο πίνακας έχει ακέραια στοιχεία και το ανάπτυγμα περιέχει μόνο πολλαπλασιασμούς, προσθέσεις και αφαιρέσεις ακεραίων, που γίνονται *χωρίς* σφάλμα στρογγύλευσης (όλα τα ενδιάμεσα αποτελέσματα είναι μικροί ακέραιοι, που αναπαρίστανται ακριβώς). Η `det` χρησιμοποιεί απαλοιφή Gauss, με διαιρέσεις, και δίνει -1.4495×10^{-12}.

(β) Κάθε κλήση για πίνακα $n \times n$ καλεί n φορές τον εαυτό της για πίνακες $(n-1)\times(n-1)$, άρα $c_n = 1 + n\,c_{n-1}$ με $c_1 = 1$. Διαιρώντας με $n!$: $c_n/n! = c_{n-1}/(n-1)! + 1/n!$, οπότε

$$c_n = n! \sum_{k=0}^{n-1} \frac{1}{k!} \approx e\, n!,$$

σε συμφωνία με τους αριθμούς της εξόδου ($c_9 = 623530 \approx 2.718 \cdot 362880$). Για $n = 20$: $e \cdot 20! \approx 6.6 \times 10^{18}$ κλήσεις, δηλαδή περίπου 6.6×10^{12} s ή 2×10^5 χρόνια. Η απαλοιφή Gauss απαιτεί για $n = 20$ περίπου 5×10^3 πράξεις. Ο ορισμός της ορίζουσας είναι χρήσιμος θεωρητικά, αλλά άχρηστος ως αλγόριθμος.

Octave/MATLAB `detcof.m`

```
function [d, calls] = detcof(A)
% DETCOF  Determinant by cofactor (Laplace) expansion along the first row.
%   [d, calls] = detcof(A) also returns the number of recursive calls.
  n = size(A, 1);
  calls = 1;
  if n == 1
    d = A(1, 1);
    return
  end
  d = 0;
  for j = 1:n
    M = A(2:n, [1:j-1, j+1:n]);             % minor: delete row 1, column j
    [dm, c] = detcof(M);
    calls = calls + c;
    d = d + (-1)^(1 + j) * A(1, j) * dm;
  end
end
```

Λύση της Άσκησης 1.22. Η έξοδος του `sol_matrix` (μετά την Άσκηση 1.12) επιβεβαιώνει όλες τις σχέσεις, εκτός βέβαια από την αντιμεταθετικότητα. Οι $(AB)^\top$ και $B^\top A^\top$ υπολογίζονται με ακριβώς τις ίδιες πράξεις ακεραίων, άρα συγκρίνονται με `isequal`. Οι αντίστροφοι, οι ορίζουσες και οι ιδιοτιμές όμως υπολογίζονται με στρογγύλευση, οπότε ο έλεγχος γίνεται με ανοχή, π.χ. `norm(X - Y) < 1e-12`, ή με σχετική ανοχή όταν οι ποσότητες είναι μεγάλες. Ο B έχει $\det B = 1$, άρα $B^{-1} = \operatorname{adj} B$, που έχει ακέραια στοιχεία αφού όλοι οι συμπαράγοντες είναι ορίζουσες ακέραιων πινάκων. Στην έξοδο χρησιμοποιήθηκε η `round`, γιατί η `inv(B)` δίνει τους ακεραίους με σφάλμα της τάξης του 10^{-15}.

Λύση της Άσκησης 1.23. (α) $\mathbf{u}^\top\mathbf{v} = 11$, $\|\mathbf{u}\| = 3$, $\|\mathbf{v}\| = 5$, άρα $\cos\varphi = 11/15$ και $\varphi = 42.83^\circ$. (β) $\frac{11}{9}(1,2,2) = (1.2222,\ 2.4444,\ 2.4444)$. (γ) $\mathbf{u}\times\mathbf{v} = (8,2,-6)$ και εμβαδόν $\sqrt{104} = 10.198$. (δ) $\det[\mathbf{u}\ \mathbf{v}\ \mathbf{w}] = \mathbf{u}^\top(\mathbf{v}\times\mathbf{w}) = (1,2,2)^\top(-4,3,3) = 8$. Για γωνίες κοντά στο 0 ή στο π ο τύπος του arccos χάνει ακρίβεια· ακριβέστερος είναι ο $\varphi = \operatorname{atan2}(\|\mathbf{u}\times\mathbf{v}\|,\ \mathbf{u}^\top\mathbf{v})$.

1.12.3 Ομάδα Γ

Οι αριθμητικοί έλεγχοι των Ασκήσεων 1.24–1.29 γίνονται από το πρόγραμμα `sol_float`, που δίνεται μετά τη λύση της Άσκησης 1.29.

Λύση της Άσκησης 1.24. (α) Στο διάστημα $[1,2)$ οι αριθμοί μηχανής απέχουν 2^{-52}. Ο πραγματικός αριθμός $1+2^{-53}$ βρίσκεται ακριβώς στο μέσο μεταξύ των 1 και $1+2^{-52}$· στην ισοπαλία το πρότυπο IEEE στρογγυλεύει προς τον αριθμό με άρτιο τελευταίο ψηφίο, δηλαδή προς το 1. Έτσι για $e = 2^{-52}$ ο έλεγχος $1 + e/2 > 1$ είναι ψευδής και ο βρόχος σταματά με $e = 2^{-52} = \epsilon_{\text{mach}}$, μετά από 52 υποδιπλασιασμούς.

(β) Σε απλή ακρίβεια προκύπτει $2^{-23} = 1.19\times10^{-7}$. Αρκεί η αρχική τιμή να είναι `single(1)`: κάθε πράξη μεταξύ `single` και `double` δίνει `single`, άρα και όλες οι ενδιάμεσες τιμές στρογγυλεύονται σε απλή ακρίβεια. Η ενσωματωμένη `eps('single')` δίνει την ίδια τιμή.

(γ) Ο βρόχος δίνει $2^{-1074} \approx 4.9\times10^{-324}$. Κάτω από το $2^{-1022} \approx 2.2\times10^{-308}$ το πρότυπο προβλέπει τους *υποκανονικούς* αριθμούς, της μορφής $(0.b_1\ldots b_{52})_2 \times 2^{-1022}$, που γεμίζουν ομοιόμορφα το κενό μέχρι το μηδέν με βήμα 2^{-1074} (*βαθμιαία υποχείλιση*). Έχουν όμως λιγότερα σημαντικά ψηφία όσο μικραίνουν.

Λύση της Άσκησης 1.25. (α) Ένα κλάσμα έχει πεπερασμένο δυαδικό ανάπτυγμα μόνο αν ο παρονομαστής του, σε ανάγωγη μορφή, είναι δύναμη του 2. Το $1/10$ έχει παρονομαστή $2\cdot5$, άρα το ανάπτυγμά του είναι περιοδικό:

$$0.1 = (0.0\,\overline{0011})_2 = (0.000110011001100\ldots)_2.$$

Ο αποθηκευμένος αριθμός είναι ο πλησιέστερος με 53 σημαντικά ψηφία, $0.1 + 5.55\times10^{-18}$.

(β) Μετά από 10^6 προσθέσεις, $t = 100000.0000013329$: το σφάλμα είναι 1.33×10^{-6}, ενώ το `1000000*0.1` δίνει ακριβώς 10^5. Το σφάλμα αναπαράστασης του 0.1 θα εξηγούσε μόνο $10^6\cdot5.55\times10^{-18} \approx 5.6\times10^{-12}$. Η κύρια αιτία είναι ότι *κάθε πρόσθεση στρογγυλεύεται*: όταν το t είναι της τάξης του 10^5, οι αριθμοί μηχανής απέχουν $2^{-36} \approx 1.5\times10^{-11}$ και κάθε πρόσθεση εισάγει σφάλμα έως το μισό αυτής της απόστασης, συστηματικά προς την ίδια κατεύθυνση για πολλά διαδοχικά βήματα. Γι' αυτό ο χρόνος υπολογίζεται πάντα ως $t_k = k\,h$ (μία στρογγύλευση) και όχι με συσσώρευση.

Λύση της Άσκησης 1.26. (α) Εδώ $b^2 - 4ac = 10^{16} - 4$ και $\sqrt{b^2-4ac} = 10^8 - 2\times10^{-8}$. Κοντά στο 10^8 όμως οι αριθμοί μηχανής απέχουν 1.49×10^{-8}, οπότε η ρίζα αποθηκεύεται ως $10^8 - 1.49\times10^{-8}$ και ο τύπος δίνει

$$x_2 = \frac{-b - \sqrt{b^2-4ac}}{2a} = \frac{1.49\times10^{-8}}{2} = 7.45\times10^{-9},$$

με σχετικό σφάλμα 25%: κανένα σωστό ψηφίο.

(β) Στον αριθμητή αφαιρούνται δύο αριθμοί που συμφωνούν σε 16 ψηφία. Το απόλυτο σφάλμα της ρίζας ($\sim 10^{-8}$) είναι αμελητέο σε σχέση με το 10^8, αλλά συγκρίσιμο με τη διαφορά. Στην ποσότητα

$q = -\frac{1}{2}\big(b+\mathrm{sgn}(b)\sqrt{b^2-4ac}\big)$ οι δύο όροι έχουν *το ίδιο πρόσημο*, άρα προστίθενται κατ' απόλυτη τιμή χωρίς απαλοιφή. Η $x_1 = q/a$ είναι η μεγαλύτερη κατ' απόλυτη τιμή ρίζα και η δεύτερη προκύπτει από τον τύπο του Vieta $x_1 x_2 = c/a$, δηλαδή $x_2 = c/q$, με μόνο μία διαίρεση.

(γ) Οι ευσταθείς τύποι δίνουν $x_2 = 1.0000000000 \times 10^{-8}$, σωστό σε όλα τα ψηφία.

Λύση της Άσκησης 1.27. Για μικρό x είναι $\cos x \approx 1 - x^2/2$, πολύ κοντά στο 1. Η υπολογισμένη τιμή του $\cos x$ έχει σφάλμα έως $\epsilon_{\text{mach}}/2$, το οποίο μεταφέρεται αυτούσιο στη διαφορά $1-\cos x \approx x^2/2$. Το σχετικό σφάλμα της f είναι επομένως περίπου

$$\frac{\epsilon_{\text{mach}}/2}{x^2/2} = \frac{\epsilon_{\text{mach}}}{x^2},$$

δηλαδή 10^{-12} για $x = 10^{-2}$, 10^{-8} για $x = 10^{-4}$, 10^{-4} για $x = 10^{-6}$, ενώ για $x \lesssim 10^{-8}$ το $\cos x$ στρογγυλεύεται ακριβώς στο 1 και η f μηδενίζεται. Αυτό ακριβώς δείχνει η πρώτη στήλη της εξόδου. Με την ταυτότητα $1 - \cos x = 2\sin^2(x/2)$,

$$f(x) = \frac{2\sin^2(x/2)}{x^2} = \frac{1}{2}\Big(\frac{\sin(x/2)}{x/2}\Big)^2,$$

δεν υπάρχει αφαίρεση και το αποτέλεσμα έχει πλήρη ακρίβεια (δεύτερη στήλη).

Λύση της Άσκησης 1.28. (α) $I_0 = \int_0^1 e^{x-1}\mathrm{d}x = 1 - e^{-1}$. Με ολοκλήρωση κατά παράγοντες:

$$I_n = \big[x^n e^{x-1}\big]_0^1 - n\int_0^1 x^{n-1}e^{x-1}\,\mathrm{d}x = 1 - n\,I_{n-1}.$$

Επειδή $0 < e^{x-1} < 1$ στο $(0,1)$, είναι $0 < I_n < \int_0^1 x^n \mathrm{d}x = 1/(n+1)$.

(β) Η αναδρομή προς τα εμπρός δίνει σωστά αποτελέσματα περίπου μέχρι το $n = 14$· για $n = 15$ μόνο δύο ψηφία είναι σωστά, για $n = 18$ το αποτέλεσμα είναι αρνητικό (-0.029), πράγμα αδύνατο, και για $n = 20$ ισούται με -30.2.

(γ) Αν η υπολογισμένη τιμή είναι $\tilde{I}_n = I_n + \delta_n$, τότε $\delta_n = -n\,\delta_{n-1}$, άρα $|\delta_n| = n!\,|\delta_0|$. Το I_0 έχει αναπόφευκτα σφάλμα $|\delta_0| \approx 10^{-17}$ έως 10^{-16}, και επειδή $18! \approx 6.4 \times 10^{15}$ το σφάλμα γίνεται της τάξης της μονάδας κοντά στο $n = 18$, όπως παρατηρήθηκε. Ο αλγόριθμος είναι *ασταθής*, αν και κάθε βήμα του είναι μαθηματικά ακριβές και εισάγει ελάχιστο σφάλμα στρογγύλευσης.

(δ) Προς τα πίσω το σφάλμα *διαιρείται* με n σε κάθε βήμα: $\delta_{n-1} = -\delta_n/n$. Ξεκινώντας από $I_{40} \approx 0$, με σφάλμα το πολύ $1/41$, το σφάλμα στο I_{20} είναι το πολύ $\frac{1}{41}\cdot\frac{20!}{40!} \approx 10^{-31}$: η πρόχειρη αρχική τιμή δεν παίζει κανένα ρόλο. Προκύπτει $I_{20} = 0.0455448841$, που ικανοποιεί και το φράγμα $I_{20} < 1/21 = 0.0476$.

Λύση της Άσκησης 1.29.

μέθοδος (απλή ακρίβεια)	S_N	σφάλμα
από $k = 1$ προς N	14.3573580	3.5×10^{-2}
από $k = N$ προς 1	14.3926516	7.5×10^{-5}
άθροιση Kahan	14.3927269	1.8×10^{-7}
διπλή ακρίβεια	14.3927267	

Όταν το μερικό άθροισμα είναι περίπου 14, οι αριθμοί απλής ακρίβειας απέχουν $2^{-20} \approx 9.5 \times 10^{-7}$. Προσθέτοντας από τους μεγάλους όρους προς τους μικρούς, κάθε όρος $1/k$ στρογγυλεύεται σε πολλαπλάσιο αυτής της απόστασης: για $k \approx 10^6$ ο όρος 10^{-6} προστίθεται ως 9.5×10^{-7}, με σχετικό σφάλμα 5%, και για ακόμη μεγαλύτερο N οι όροι χάνονται εντελώς. Προσθέτοντας από τους μικρούς προς τους μεγάλους, οι μικροί όροι αθροίζονται μεταξύ τους όσο το άθροισμα είναι ακόμη μικρό και οι αριθμοί μηχανής πυκνοί. Γενικός κανόνας: *αθροίζουμε κατά αύξον μέγεθος*.

Στην άθροιση Kahan η `c` κρατά το σφάλμα στρογγύλευσης της τελευταίας πρόσθεσης: η παράσταση `(t - s) - y` ισούται με το μέρος του `y` που *δεν* χώρεσε στο `t` (με αντίθετο πρόσημο), και αφαιρείται από τον επόμενο όρο. Το σφάλμα του αποτελέσματος είναι της τάξης του ϵ_{mach}, ανεξάρτητα από το N, με κόστος τέσσερις πράξεις ανά όρο. Η μέθοδος προϋποθέτει ότι οι πράξεις εκτελούνται ακριβώς με τη σειρά που γράφονται· μαθηματικά η `c` είναι πάντα μηδέν, και μια «απλοποίηση» της παράστασης θα κατέστρεφε τη διόρθωση.

Octave/MATLAB `sol_float.m`

```
% Solutions 1.24–1.29 (group C, floating-point arithmetic)
fprintf('--- 1.24\n');
e = single(1);  k = 0;
while single(1) + e / 2 > single(1)
  e = e / 2;  k = k + 1;
end
fprintf('single: eps = %.6e = 2^(-%d)\n', e, k);
x = 1;  k = 0;
while x / 2 > 0
  x = x / 2;  k = k + 1;
end
fprintf('smallest positive double = %.4e = 2^(-%d)\n', x, k);

fprintf('--- 1.25\n');
t = 0;
for i = 1:1000000
  t = t + 0.1;
end
fprintf('t = %.10f, error = %.3e, by product: %.3e\n', t, abs(t - 1e5), abs(1000000 * 0.1
    - 1e5));

fprintf('--- 1.26\n');
a = 1;  b = -1e8;  c = 1;
d = sqrt(b * b - 4 * a * c);
naive = (-b - d) / (2 * a);
q = -(b - d) / 2;                       % b < 0: -(b + sign(b) d)/2
stable = c / q;
fprintf('naive = %.10e, stable = %.10e, rel. error naive = %.2f\n', naive, stable,
    abs(naive - 1e-8) / 1e-8);

fprintf('--- 1.27\n');
for p = 1:8
  x = 10^(-p);
  fprintf('x = 1e-%d: %.12f  %.12f\n', p, (1 - cos(x)) / x^2, 2 * sin(x / 2)^2 / x^2);
end

fprintf('--- 1.28\n');
N = 20;
fwd = zeros(1, N + 1);                  % fwd(n+1) = I_n  (indices start at 1)
fwd(1) = 1 - exp(-1);
for n = 1:N
  fwd(n + 1) = 1 - n * fwd(n);
end
bwd = zeros(1, 41);
for n = 40:-1:1
  bwd(n) = (1 - bwd(n + 1)) / n;
end
for n = [5 10 15 18 20]
  fprintf('n = %2d: forward %14.6e   backward %.10f\n', n, fwd(n + 1), bwd(n + 1));
```

```
end

fprintf('--- 1.29\n');
N = 1000000;  one = single(1);
fw = single(0);
for k = 1:N
  fw = fw + one / single(k);
end
bw = single(0);
for k = N:-1:1
  bw = bw + one / single(k);
end
s = single(0);  comp = single(0);        % Kahan compensated summation
for k = 1:N
  y = one / single(k) - comp;
  tt = s + y;
  comp = (tt - s) - y;
  s = tt;
end
dbl = 0;
for k = N:-1:1
  dbl = dbl + 1 / k;
end
fprintf('double   %.7f\n', dbl);
fprintf('forward  %.7f  error %.1e\n', fw, abs(double(fw) - dbl));
fprintf('backward %.7f  error %.1e\n', bw, abs(double(bw) - dbl));
fprintf('Kahan    %.7f  error %.1e\n', s, abs(double(s) - dbl));
```

Έξοδος

```
--- 1.24
single: eps = 1.192093e-07 = 2^(-23)
smallest positive double = 4.9407e-324 = 2^(-1074)
--- 1.25
t = 100000.0000013329, error = 1.333e-06, by product: 0.000e+00
--- 1.26
naive = 7.4505805969e-09, stable = 1.0000000000e-08, rel. error naive = 0.25
--- 1.27
x = 1e-1: 0.499583472197  0.499583472197
x = 1e-2: 0.499995833347  0.499995833347
x = 1e-3: 0.499999958326  0.499999958333
x = 1e-4: 0.499999996961  0.499999999583
x = 1e-5: 0.500000041370  0.499999999996
x = 1e-6: 0.500044450291  0.500000000000
x = 1e-7: 0.499600361081  0.500000000000
x = 1e-8: 0.000000000000  0.500000000000
--- 1.28
n =  5: forward   1.455329e-01   backward 0.1455329406
n = 10: forward   8.387707e-02   backward 0.0838770701
n = 15: forward   5.903379e-02   backward 0.0590175409
n = 18: forward  -2.945367e-02   backward 0.0501198550
n = 20: forward  -3.019239e+01   backward 0.0455448841
--- 1.29
double   14.3927267
forward  14.3573580  error 3.5e-02
backward 14.3926516  error 7.5e-05
Kahan    14.3927269  error 1.8e-07
```

Λύση της Άσκησης 1.30. (α) Ο λόγος διαδοχικών όρων είναι $|t_k/t_{k-1}| = |x|/k$, μεγαλύτερος του 1 όσο $k < |x|$ και μικρότερος μετά· άρα οι όροι αυξάνουν μέχρι $k \approx |x|$. Με τον τύπο του Stirling $k! \approx \sqrt{2\pi k}\,(k/e)^k$, για $k = |x|$:

$$\max_k |t_k| \approx \frac{|x|^{|x|}}{|x|!} \approx \frac{e^{|x|}}{\sqrt{2\pi|x|}}.$$

(β) Κάθε όρος υπολογίζεται με σχετικό σφάλμα της τάξης του ϵ_{mach}, άρα το απόλυτο σφάλμα του αθροίσματος είναι τουλάχιστον $\epsilon_{\text{mach}} \max|t_k|$. Για $x < 0$ το αποτέλεσμα είναι $e^{-|x|}$, οπότε

$$\text{σχετικό σφάλμα} \approx \frac{\epsilon_{\text{mach}}\, e^{|x|}/\sqrt{2\pi|x|}}{e^{-|x|}} = \epsilon_{\text{mach}}\,\frac{e^{2|x|}}{\sqrt{2\pi|x|}}.$$

Για $x = -10$ ο τύπος δίνει 1.4×10^{-8} (παρατηρήθηκε 3×10^{-9}), για $x = -20$ δίνει 4.7 (παρατηρήθηκε 1.7) και για $x = -30$ δίνει 1.8×10^{9} (παρατηρήθηκε 3×10^{8}): σωστή τάξη μεγέθους σε όλες τις περιπτώσεις. Για $x > 0$ όλοι οι όροι είναι θετικοί, το αποτέλεσμα είναι μεγαλύτερο από κάθε όρο και το σχετικό σφάλμα μένει της τάξης του ϵ_{mach}.

(γ) Όλα τα ψηφία χάνονται όταν το σχετικό σφάλμα φτάσει τη μονάδα: $e^{2|x|} \approx \sqrt{2\pi|x|}/\epsilon_{\text{mach}}$, δηλαδή $|x| \approx \frac{1}{2}\ln(5 \times 10^{16}) \approx 19$. Στην περιοχή αυτή το αποτέλεσμα εξαρτάται ακόμη και από τη σειρά με την οποία γράφονται οι πράξεις: αν ο όρος ενημερωθεί ως $t \cdot (x/k)$ αντί για $(t \cdot x)/k$, προκύπτουν εντελώς διαφορετικοί (και εξίσου λανθασμένοι) αριθμοί.

1.12.4 Υποδείξεις για τις υπόλοιπες ασκήσεις

1.14 Ο όρος ενημερώνεται ως $t_k = -t_{k-1}\,x^2/\big((2k)(2k+1)\big)$ με $t_0 = x$. Για $x = 30$ ο μεγαλύτερος όρος είναι περίπου 8×10^{11} και χάνονται 12 ψηφία. Η αναγωγή γίνεται με $x \leftarrow x - 2\pi\,\text{round}(x/2\pi)$.

1.16 Ο χρόνος των τριών βρόχων αυξάνει ως n^3 (οκταπλασιάζεται όταν διπλασιάζεται το n). Ο ενσωματωμένος πολλαπλασιασμός καλεί βελτιστοποιημένη βιβλιοθήκη (BLAS) σε μεταγλωττισμένο κώδικα, ενώ οι βρόχοι διερμηνεύονται εντολή προς εντολή· η διαφορά είναι δύο έως τρεις τάξεις μεγέθους. Γενικός κανόνας: στο Octave/MATLAB προτιμάμε τις πράξεις πινάκων και τις ενσωματωμένες συναρτήσεις από τους βρόχους, όπου αυτό είναι δυνατό.

1.17 (α) `data = load('arrhenius.dat'); D = data(:, 2);` και `numel(D)`, `min(D)`, `max(D)`, `mean(D)`. (β) Μετά το `fid = fopen(...)`, ο βρόχος `while true` διαβάζει `line = fgetl(fid)` και σταματά με `break` όταν `~ischar(line)` (τέλος αρχείου)· η `v = sscanf(line, '%f')` δίνει τους δύο αριθμούς της γραμμής. Το ελάχιστο και το μέγιστο αρχικοποιούνται με `Inf` και `-Inf`, όχι με το μηδέν.

1.19 Η τυπική απόκλιση της εκτίμησης είναι $4\sqrt{p(1-p)/N}$ με $p = \pi/4$, δηλαδή περίπου $1.64/\sqrt{N}$. Χωρίς βρόχο: `x = rand(N, 1); y = rand(N, 1); p = 4 * sum(x.^2 + y.^2 <= 1) / N`. Για επαναλήψιμα αποτελέσματα η γεννήτρια αρχικοποιείται με `rand('seed', 1)` (Octave) ή `rng(1)` (MATLAB).

1.20 Στο $x = 0$ προκύπτει $0/0 =$ `NaN` και το σημείο λείπει από το γράφημα. Αντικαταστήστε την τιμή με το όριο 1, π.χ. `y(x == 0) = 1`, ή χρησιμοποιήστε τη `sinc(x/pi)`. Οι εντολές `title`, `xlabel`, `ylabel`, `grid on` και `print('-dpng', 'sinc.png')` ολοκληρώνουν το γράφημα.

1.31 Οι όροι της ανεπτυγμένης μορφής είναι της τάξης του 20 κοντά στο $x = 1$, άρα ο θόρυβος είναι περίπου $20\,\epsilon_{\text{mach}} \approx 4 \times 10^{-15}$, ενώ $(x-1)^6 \le 1.6 \times 10^{-14}$ σε όλο το διάστημα. Η υπολογισμένη p αλλάζει πρόσημο ακανόνιστα για $|x-1| \lesssim (4\times10^{-15})^{1/6} \approx 0.004$, οπότε η διχοτόμηση μπορεί να «βρει» ρίζα οπουδήποτε σε αυτή την περιοχή (βλ. και Άσκηση 4.17).