Πληροφορική

Ενότητα 5: Α. Λογικές εκφράσεις και δείκτες (Β΄ μέρος- Διαχείριση πινάκων).
Β. Αριθμητικές μέθοδοι επίλυσης προβλημάτων

Κωνσταντίνος Καρατζάς
Τμήμα Μηχανολόγων Μηχανικών

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ ΚΑΙ ΘΡΗΣΚΕΥΜΑΤΩΝ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.

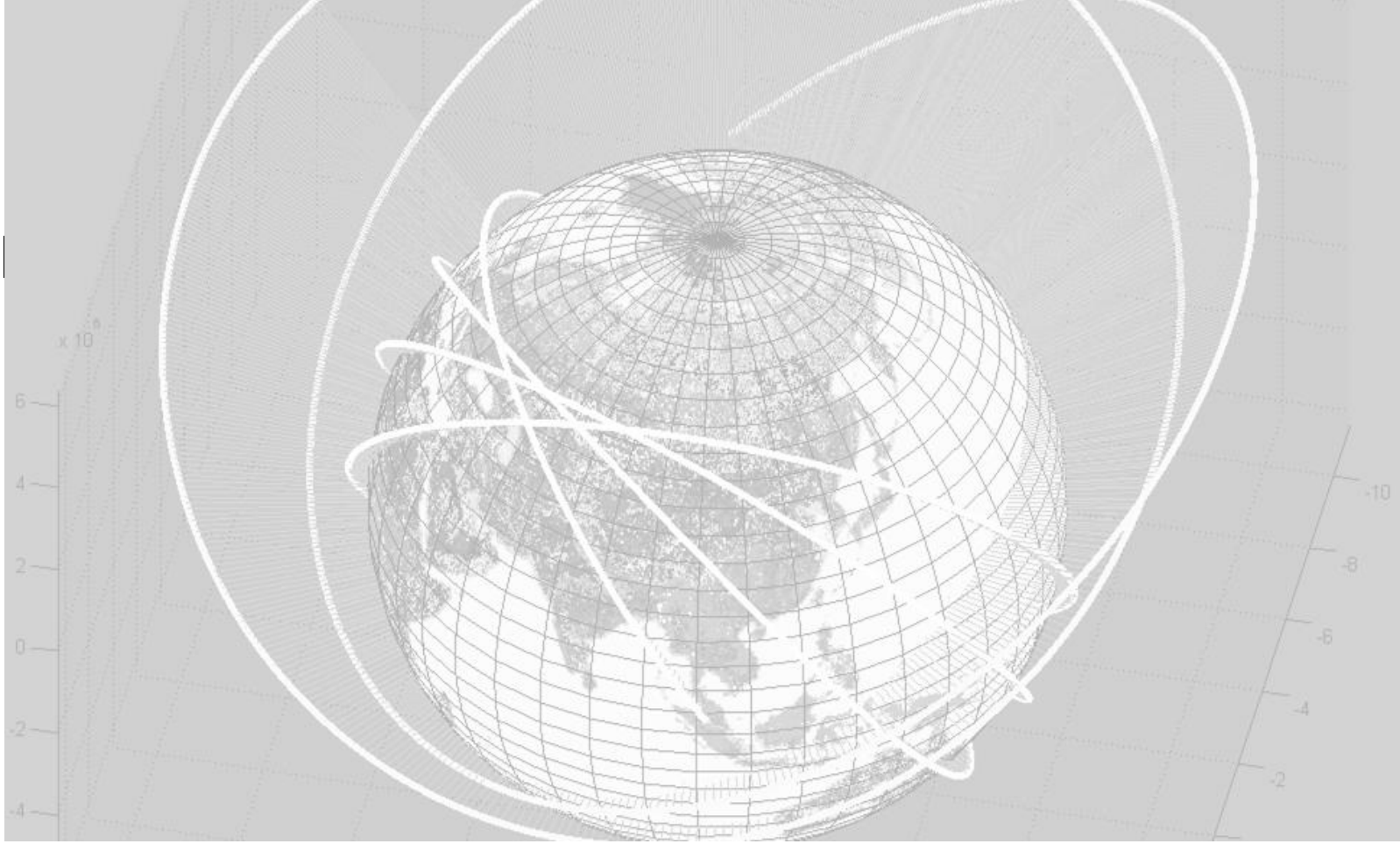
Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «Ανοικτά Ακαδημαϊκά Μαθήματα στο Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης» έχει χρηματοδοτήσει μόνο την αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Δ.#8 ΠΛΗΡΟΦΟΡΙΚΗ:
ΛΟΓΙΚΕΣ ΕΚΦΡΑΣΕΙΣ & ΔΕΙΚΤΕΣ – Β ΜΕΡΟΣ: ΔΙΑΧΕΙΡΙΣΗ ΠΙΝΑΚΩΝ

Κώστας Καρατζάς
Τμήμα Μηχανολόγων Μηχανικών, ΑΠΘ

24.03.2015



Λογικές Εκφράσεις - Συνέχεια

Λογικοί τελεστές

- Σύγκριση στοιχείο προς στοιχείο (οι τελεστέοι αποτελούν στοιχεία πινάκων)
  - **and, or (&, |)**
  - Κάθε στοιχείο κρίνεται ως προς το εάν είναι
    - Μηδέν -> **false**
    - Όχι μηδέν -> **true**
    - **Το αποτέλεσμα είναι ένας πίνακας λογικών τιμών (0,1) ίδιων διαστάσεων με τα συγκρινόμενα διανύσματα - πίνακες**
- Σύγκριση δύο λογικών τιμών (οι τελεστέοι αποτελούν «βαθμωτά» μεγέθη)
  - **βραχυκυκλωμένο and, or (&&, ||)**
  - Κάθε τιμή κρίνεται ως προς το εάν είναι
    - Μηδέν -> **false**
    - Όχι μηδέν -> **true**
    - **Το αποτέλεσμα είναι μία λογική τιμή (0,1)**

Συνοψίζοντας από την προηγούμενη Τρίτη..

- Σχεσιακοί τελεστές: συγκρίνουν δύο ποσότητες. Το αποτέλεσμα είναι true/false
  - Σύγκριση στοιχείο προς στοιχείο πάντα!
- Λογικοί τελεστές:ελέγχουν μία παράσταση. Το αποτέλεσμα είναι true/false
  - Σύγκριση στοιχείο προς στοιχείο
  - Σύγκριση λογικών τιμών (βραχυκυκλ.)
- Λογικές συναρτήσεις: **`είναι_κάτι, βρές, κλπ`**

Προτεραιότητα

Μοναδιαίοι τελεστές

Precedence level	Operator
1 (highest)	Parentheses ()
2	Transpose (.'), power (.^), complex conjugate transpose ('), matrix power (^)
3	Unary plus (+), unary minus (−), logical negation (~)
4	Multiplication (.*), right division (./), left division (.\), matrix multiplication (*), matrix right division (/), matrix left division (\)
5	Addition (+), subtraction (−)
6	Colon operator (:)
7	Less than (<), less than or equal to (<=), greater than (>), greater than or equal to (>=), equal to (==), not equal to (~=)
8	Logical and (&)
9	Logical or (|)
10	Logical short-circuit and (&&)
11 (lowest)	Logical short-circuit or (||)

Για "ισοδύναμους" τελεστές, προτεραιότητα έχουν οι πράξεις από αριστερά

Λογικός (logical) τύπος δεδομένων

- Πρόκειται για έναν ακόμη τύπο δεδομένων στο MATLAB; λαμβάνει τιμές 1 (true) ή 0 (false).

Είσοδος	Έξοδος: ans =	Data type
4 < 5	**1**	Logical array
4 <= 5	**1**	Logical array
4 > 5	**0**	Logical array
4 >= 5	**0**	Logical array
4 ~= 5	**1**	Logical array

Ενδιαφέρουσες δοκιμές

```
>> 3 > 5 & 0
Τί είναι αυτό? Έχει νόημα?
Ναί!!! (σκεφτείτε την προτεραιότητα πράξεων)

Έστω τώρα r= 0.6
0<r<1. Μαθηματικά είναι σωστό αλλά…
Αποτέλεσμα?
Γιατί?
0<r TRUE, δηλ 1 (λογικό)
Οπότε
1<1 FALSE, δηλ 0 (λογικό)
Το σωστό είναι:
0<r & r<1 (προτεραιότητες ή παρενθέσεις!)
```

Είδαμε ότι

- Οι λογικές εκφράσεις αποτελούν ιδανικό τρόπο για τον έλεγχο τιμών
- Τί μπορούν να κάνουν για εμάς σε σχέση με τη διαχείριση πινάκων?

Παράδειγμα 1: Αποφυγή απείρου

Έστω ότιθέλω να σχεδιάσω την τριγ. συν. tan

Αρχικά ορίζω ένα διάστημα τιμών του χ

```
>> x=-3*pi:pi/100:3/2*pi;
```

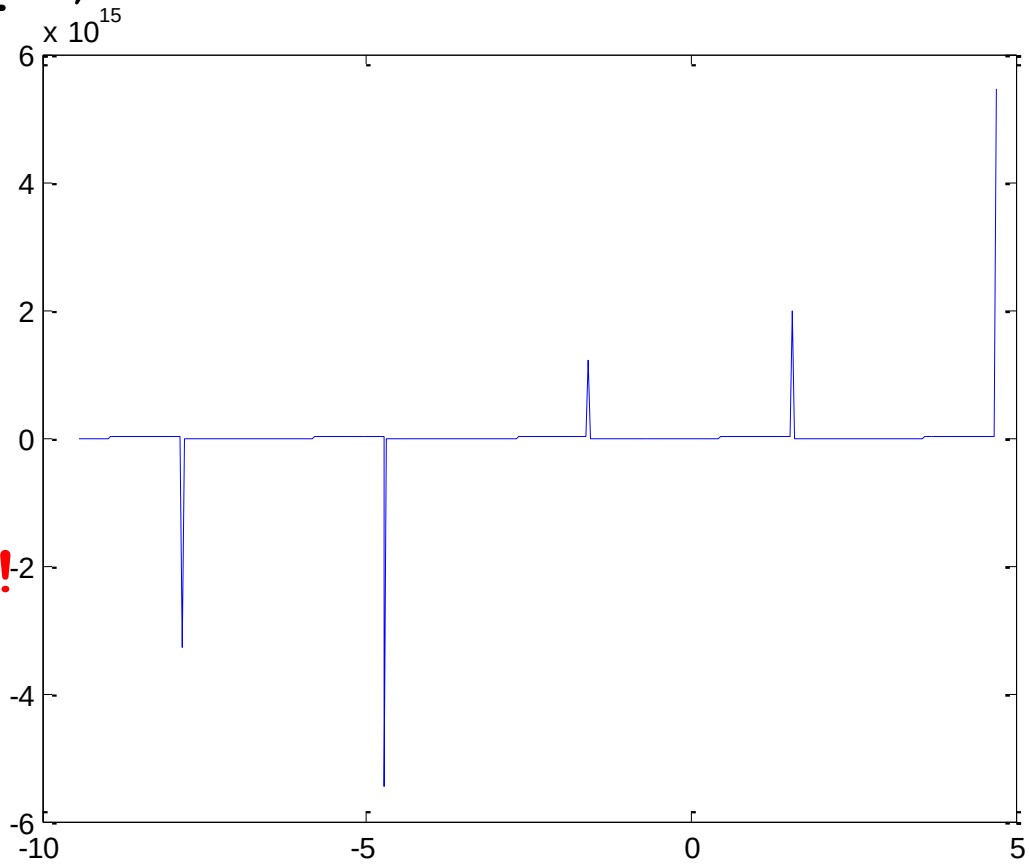
Ορίζω τη συνάρτηση

```
>> y=tan(x)
```

Σχεδιάζω

```
>> plot(x,y)
```

Και προκύπτει πρόβλημα!



Τι τις χρειαζόμαστε λοιπόν?

Κάτι δεν πάει καλά!

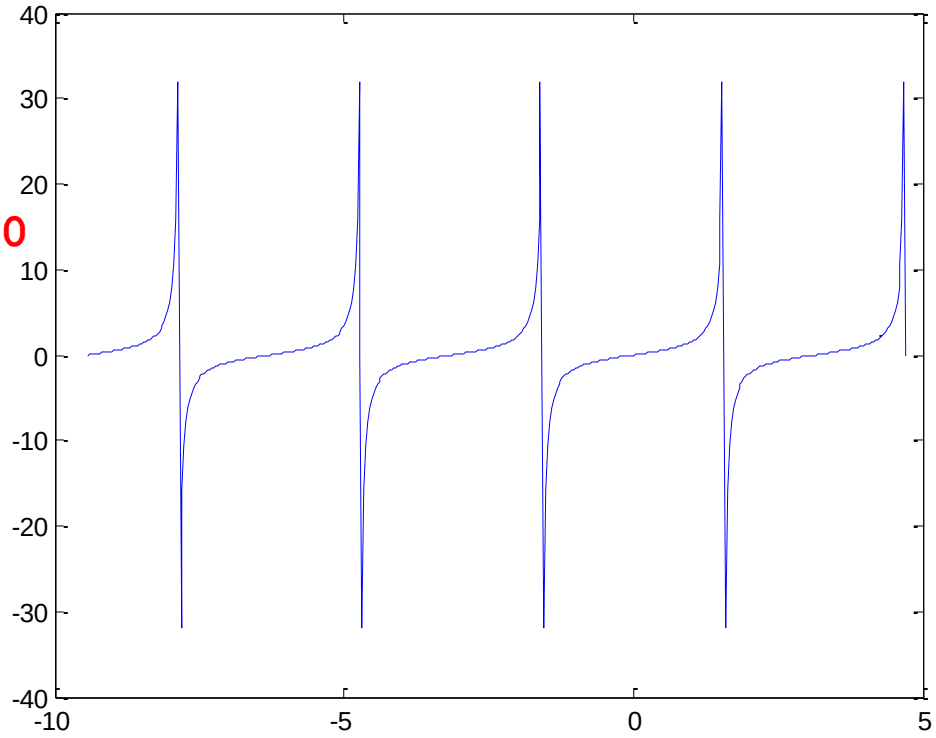
Η tan(x)τείνει στο ± ∞ για περιττά πολλαπλάσια του π/2!

Πρέπει να "αφαιρέσω" από το διάνυσμα τιμών y αυτές τις τιμές

Θέλω τα στοιχεία του y που είναι < 10^{10} (απόλυτη τιμή)

Λογικός έλεγχος: abs(y)<1e10

```
>> y=y.*(abs(y)<1e10);
>> plot(x,y)
```



Παράδειγμα 2: Αποφυγή του μηδενός

Θέλω το sin(x)/x από -4π έως 4π

```
x = -4*pi : pi / 20 : 4*pi;
```

x(81)=0

Γιατί;

Και μετά.. `y = sin(x) ./ x;`

Οπότε το sin(x(81))/x(81) δεν ορίζεται!

Αντικαθιστώ όσα στοιχεία του x είναι μηδέν με κάτι πολύ κοντά στο μηδέν (γιατί?)

```
x = x + (x == 0)*eps;
```

$$\lim_{x \to 0} \frac{sin(x)}{x} = 1$$

`sin(eps)/eps=1` (γιατί; δείτε εδώ και διαβαστε και τον κανονα του L'Hôpital)

```
plot(x,y)
```

Οι λογικές εκφράσεις επιτελούν καίριους ελέγχους και είναι απαραίτητες στην αποφυγή συχνών λογικών λαθών

Η εντολή find (ξανά..)

Μία ιδιαίτερα χρήσιμη εντολή του Matlab είναι η εντολή find, η οποία μας δίνει τη δυνατότητα να βρούμε στοιχεία ενός πίνακα που πληρούν μία λογική συνθήκη. Η σύνταξή της είναι:

find(λογική έκφραση)

και επιστρέφει μία λίστα από θέσεις του πίνακα που πληρούν την λογική έκφραση.

```
>> A = magic(4)% θέλω όσα >8 να γίνουν 100
A =

    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1

i = find(A > 8); δείτε το αποτέλεσμα!

A(i) = 100

Εναλλακτικά:

A(A > 8) = 100;
```

déja vú : δομές επανάληψης

`for`: όταν γνωρίζω αριθμό επαναλήψεων

`while` : όταν δεν γνωρίζω αριθμό επαναλήψεων αλλά γνωρίζω συνθήκη ελέγχου

Τα **`while`** χρησιμοποιούν συνθήκες ελέγχου: συνεχίσουν έως ότου αυτές γίνουν ψευδείς

Τα **`for loops`** μπορούν να μετατραπούν σε **`while loops`**

Από for σε while

Μπορώ να μετατρέψω ένα for loop σε ένα

while loop μετατρέποντας την επανάληψη σε λογική συνθήκη

for -- > while

```
for i=1:2:10
    disp(i^2)

end
```

```
i=1;
while i<=10
  disp(i^2)
  i+i+2;
end
```

for -- > while

```
for i=[2 4 1 6]
  disp(i^2)
end
```

```
i=[2 4 1 6]
while i<=noe
    disp(i^2)
    noe=noe+1;
end
```

Περί κατηγοριών δεδομένων

- Αριθμοί, για αριθμητικού τύπου δεδομένα

```
students_present_today = 89;
% ένας ακέραιος
average_number_of_students_in_class_  =  111,3;
% ένας πραγματικός
```

- Λογικοί αριθμοί – Boolean (true ή false) , για λογικού τύπου δεδομένα

```
Simera_exei_ilio = true;
```

- Χαρακτήρες, για αλφαριθμητικού τύπου δεδομένα

```
onoma_miteras_foititi = 'eleni';
```

- Πίνακες, που περιλαμβάνουν λίστες δεδομένων ίδιου τύπου (ενός από τους προαναφερόμενους)

```
student_grades = [97, 78, 88, 93, 89];
```

Ένα νέο είδος δεδομένων: `structs`

- Πολλές φορές όμως θέλουμε να «ομαδοποιήσουμε» δεδομένα διαφορετικών τύπων σε έναν ενιαίο, σύνθετο τύπο δεδομένων, που να μας επιτρέπει να περιγράψουμε μία «οντότητα» ή «αντικείμενο» πλήρως, στα πλαίσια των αναγκών μας

Ένα νέο είδος δεδομένων: **`structs`**

- Structures (δομές) = δομές δεδομένων στο MATLAB που χρησιμοποιούνται για να αποθηκεύουν δεδομένα διαφορετικού τύπου σε ένα «αντικείμενο»
- Κάθε structure αποτελείται από πεδία
- Κάθε πεδίο αποτελείται από έναν πίνακα κάποιου συγκεκριμένου τύπου

Ένα νέο είδος δεδομένων: **structs**

- Πολλές φορές όμως θέλουμε να «ομαδοποιήσουμε» δεδομένα διαφορετικών τύπων σε έναν ενιαίο, σύνθετο τύπο δεδομένων, που να μας επιτρέπει να περιγράψουμε μία «οντότητα» ή «αντικείμενο» πλήρως, στα πλαίσια των αναγκών μας

```
student_name = 'Xanapestos Kanenas';
student_recordID =1234;
student_semester = 3;
student_new_in_class = true;
student_exercises_grade = 7.2
```

- το όνομα της δομής (μεταβλητής) είναι, student, η «περίοδος» είναι 1 και τα ονόματα των πεδίων είναι name, recordID, new_in_class, exercises_grade. Επισημαίνεται ότι το κάθε πεδίο έχει τον δικό του τύπο δεδομένων.

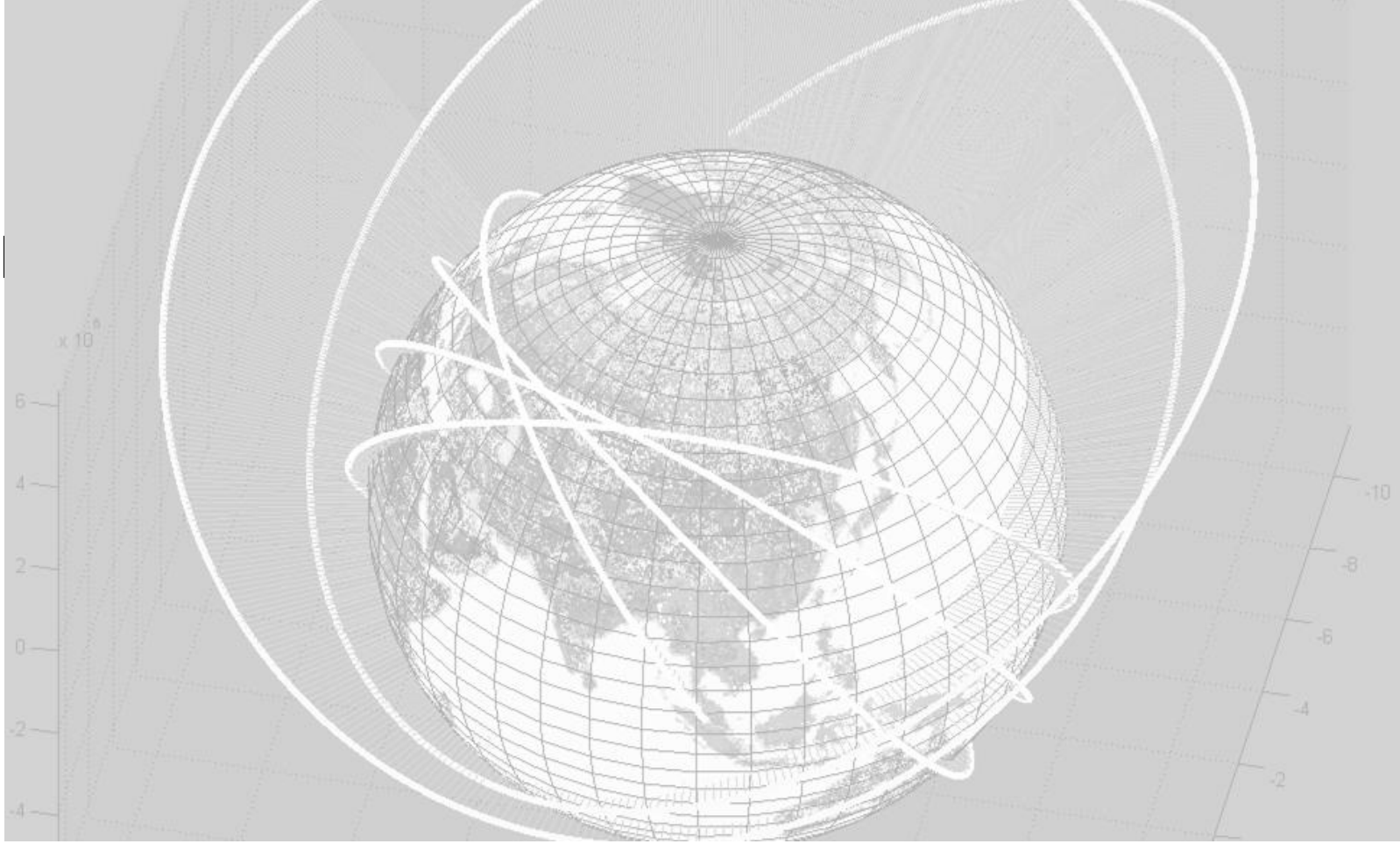
```
student(1).name = 'Xanapestos Kanenas';
student(1).recordID =1234;
student(1).semester = 3;
student(1).new_in_class = true;
student(1).exercises_grade = 7.2
```

```
student(1).name = 'Xanapestos Kanenas';
student(1).recordID =1234;
student(1).semester = 3;
student(1).new_in_class = true;
student(1).exercises_grade = 7.2
```

```
student(2).name = 'Xanapestos Kanenas';
student(2).recordID =6666;
student(2).semester = 2;
student(2).new_in_class = true;
student(2).exercises_grade = 9.2
```

```
>> myfiles = dir;  % όλα τα αρχεία του τρέχοντος καταλόγου του Matlab
>> myfiles(4)      % κάθε στοιχείο είναι μία δομή (struct)
ans =
       name: 'Figure-1a.tif'
       date: '04-Μαρ-2009 13:47:32'
      bytes: 290348
      isdir: 0
    datenum: 7.3384e+005

>> for i=4:8       % Τα ονόματα των αρχείων 4 ως 8
   disp(myfiles(i).name)
end
Figure-1a.tif
Figure-1b.tif
Figure-2.tif
PM-Timeseries.fig
PM-Timeseries.tif
```



Τέλος

Κώστας Καρατζάς

Τμήμα Μηχανολόγων Μηχανικών, ΑΠΘ

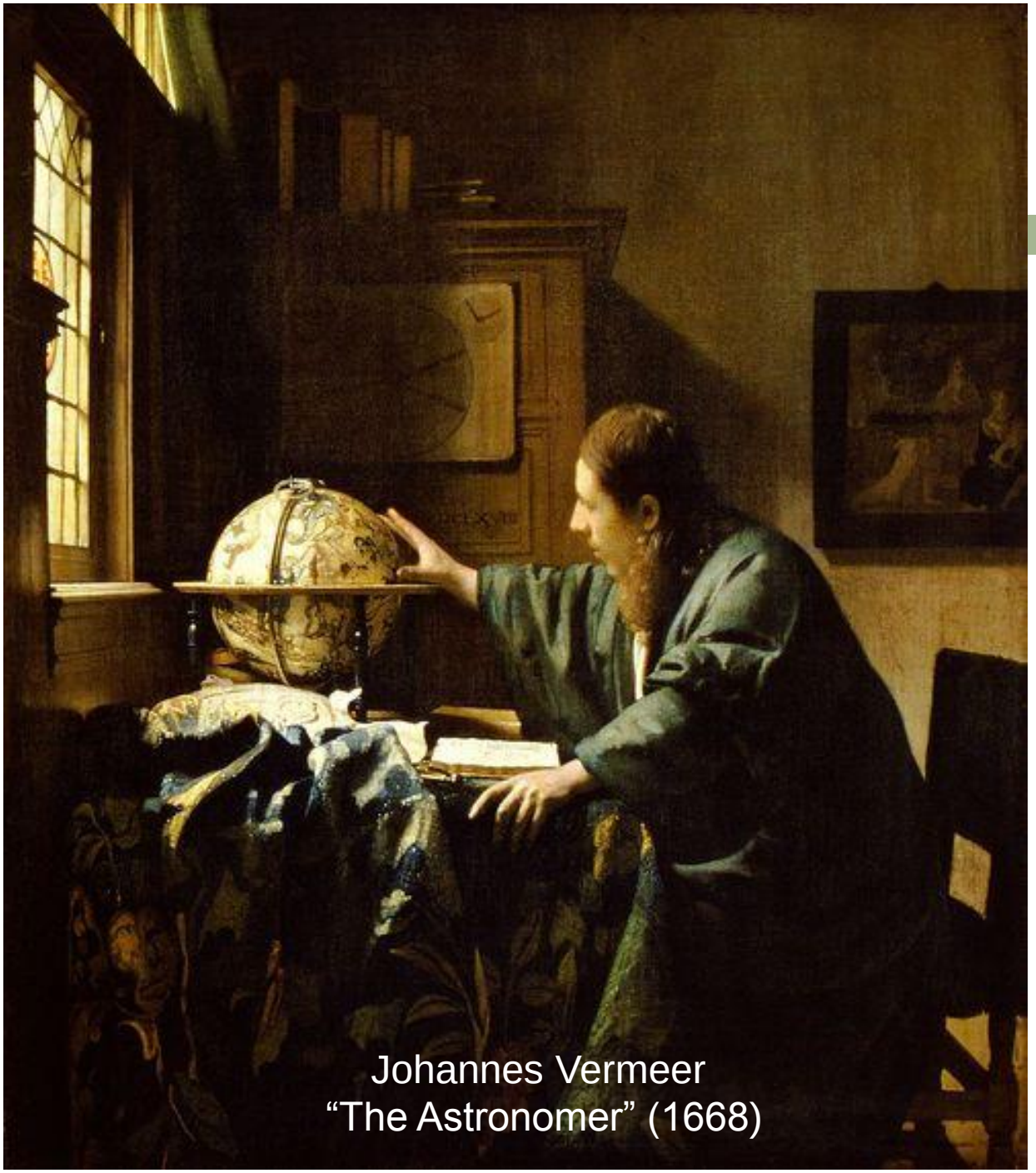
Δ.#9: ΠΛΗΡΟΦΟΡΙΚΗ

ΑΡΙΘΜΗΤΙΚΕς ΜΕΘΟΔΟΙ ΕΠΙΛΥΣΗΣ ΠΡΟΒΛΗΜΑΤΩΝ

Κωνσταντίνος Καρατζάς
Τμήμα Μηχανολόγων Μηχανικών ΑΠΘ

24.03.2015

Αριθμητικές (και όχι μόνο) μέθοδοι επίλυσης προβλημάτων



Μέθοδοι επίλυσης προβλημάτων

Οι αριθμητικές μέθοδοι:

- Λειτουργούν υπολογίζοντας αριθμητικές ποσότητες βάσει σχέσεων
- Αποτελούν αντικείμενο του μαθήματος της **Αριθμητικής Ανάλυσης**

Προθέρμανση…εξαντλητική αναζήτηση

- Έστω 1000 αριθμοί, ο καθένας μεταξύ 1 και 10^6, διατεταγμένοι από τον μικρότερο στο μεγαλύτερο
- Ο αριθμός 123456 περιλαμβάνεται σε αυτούς?

Exhaustive enumeration –brute force

- Ελέγχουμε έναν προς έναν όλους τους αριθμούς

```
Επανέλαβε όσο ελεγχ.αριθμός διαφορετικός ζητούμενου

        Ελεγχόμενος αριθμός = πρώτος αριθμός λίστας

        Ελεγχόμενος αριθμός = ζητούμενο -> βρέθηκε λύση

        Ελεγχόμενος αριθμός = επόμενος αριθμός λίστας
```

2. Η μέθοδος της διχοτόμησης

- Χωρίζουμε τη λίστα σε δύο ίσα τμήματα
- Από τον μικρότερο και μεγαλύτερο αριθμό κάθε τμήματος επιλέγουμε το τμήμα που πιθανόν περιλαμβάνει τον ζητούμενο αριθμό
- Επαναλαμβάνουμε
  - Πόσες επαναλήψεις χρειαζόμαστε?

Και σε προβλήματα επίλυσης εξισώσεων-εύρεσης τιμών

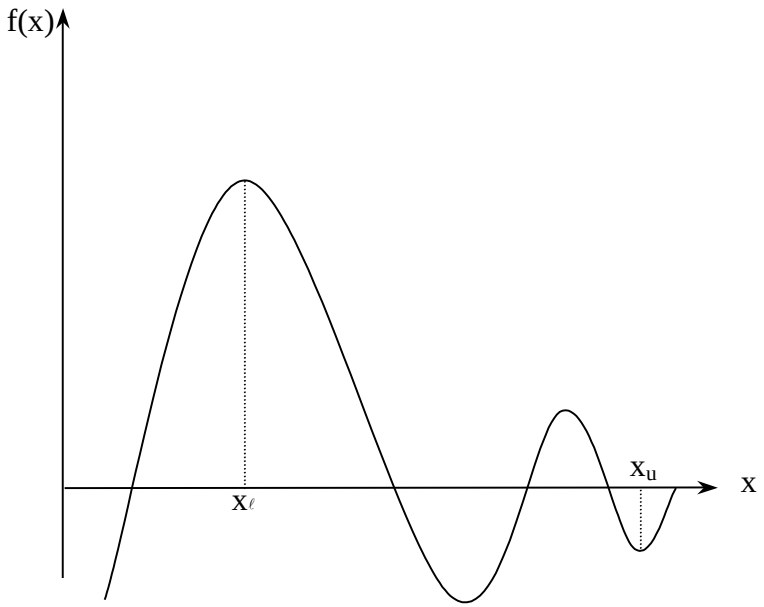
Μέθοδος της διχοτόμησης

Σύμφωνα με το θεώρημα **Bolzano** αν για μία συνεχή συνάρτηση, f, στο [a,b] ισχύει $f(a)\cdot f(b)<0$

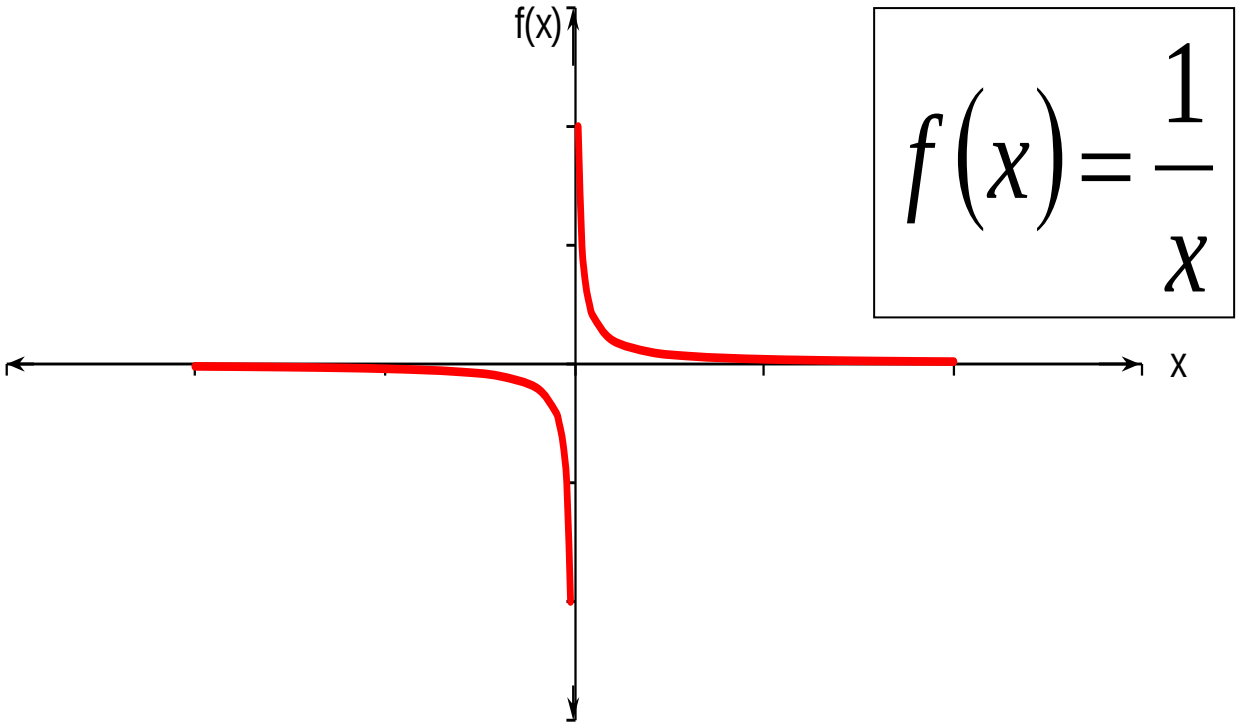
τότε η εξίσωση

$f(x)=0$

έχει μία τουλάχιστον λύση εντός του [a,b]



Εάν η συνάρτηση δεν είναι συνεχής..



$$f(x)=\frac{1}{x}$$

Ο αλγόριθμος του προβλήματος

Η μέθοδος της διχοτόμησης, κάνει χρήση του θεωρήματος Bolzano για να υπολογίσει τη λύση μίας εξίσωσης. Η λύση προσεγγίζεται με διαδοχικές διχοτομήσεις του διαστήματος.

```
f(a)*f(b)<0
x1=a, x2=b
root=(x1 + x2)/2

Επανέλαβε μέχρι να έχουμε λύση

f(root)=0  έχουμε λύση
f(a)*f(root)<0  η λύση είναι στο [a, root]  -> x2=root
f(root)*f(b)<0  η λύση είναι στο [root, b]  -> x1=root
```

Παράδειγμα

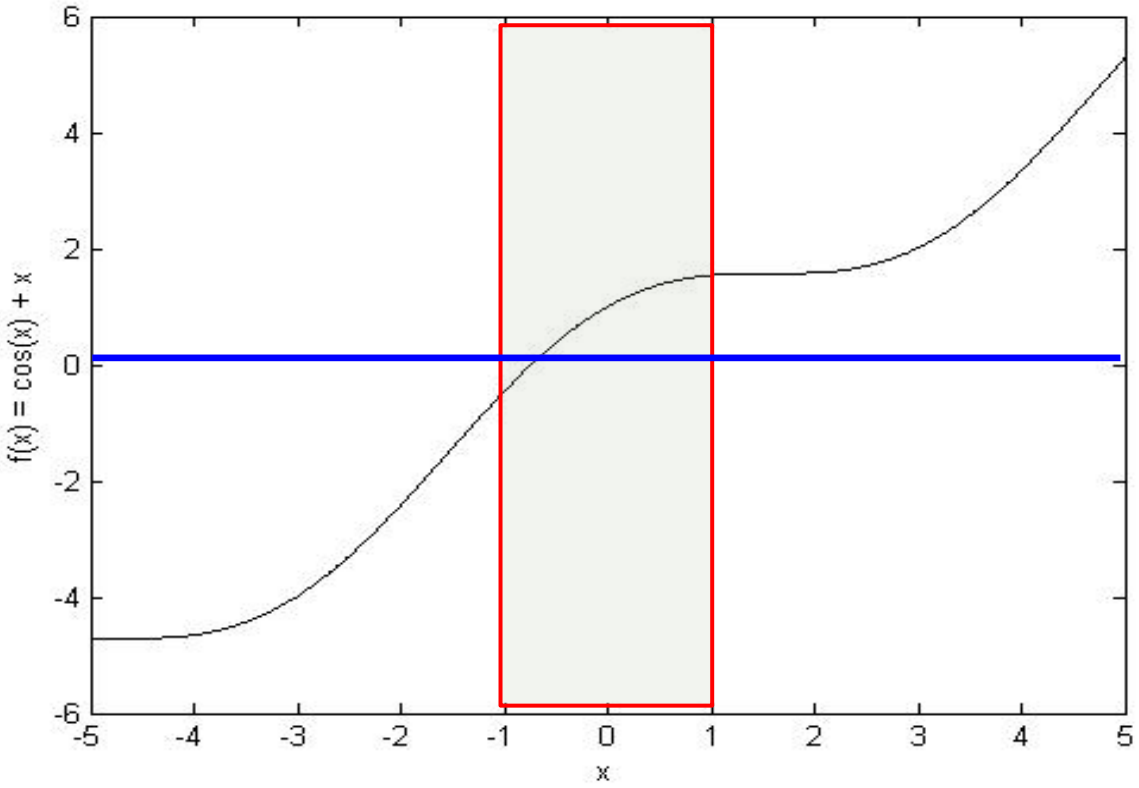
- $x^3 + 4x^2 - 1 = 0$

a	*b*	$x_c = (a+b)/2$	*f*(*a*)	*f*(*b*)	*f*(x_c)
0	1	0.5	**-1**	4	**0.125**
0	0.5	0.25	-1	**0.125**	**- 0.73438**
0.25	0.5	0.375	-0.73438	**0.125**	**- 0.38477**
0.375	0.5	0.4375	-0.38477	**0.125**	**- 0.15063**
0.4375	0.5	0.46875	-0.15063	**0.125**	**- 0.0181**
0.46875	0.5	0.484375	**-0.0181**	0.125	**0.05212**
0.46875	0.484375	0.476563	**- 0.0181**	0.05212	**0.01668**

Και τελικά… X=0.472834.

Άκρίβεια: η διαφορά μεταξύ δύο τιμών

Άσκηση εργαστηρίου



- $f(x) = cos(x) + x$

Βασικά στοιχεία

- Πόσες και ποιες μεταβλητές?
- Αρχικοποιήσεις
- Από τον αλγόριθμο στον κώδικα
  - Ποιού τύπου επανάληψη θα επιλέξουμε?
    - Μέγιστος αριθμός επαναλήψεων?
    - Αποδεκτή απόκλιση λύσης
  - Αποδεκτή απόκλιση λύσης?

Μεταβλητές-αρχικοποιήσεις

```
a = -1;
b = 1;
root = a;
acc = abs(root - (a+b)/2)= 1;
tol = 1e-6;
```

```
function f = myFunc(x)
f = x + cos(x);
end
```

Χρήση `while`

```
while (acc>tol)

    acc = abs(root - (a+b)/2);
    root = (a + b)/2;

    if(myFunc(a)*myFunc(root)<0)
        b = root;
    elseif(myFunc(b)*myFunc(root)<0)
        a = root;
    end


end
```

Πόσες φορές θα τρέξει?

```
while (acc>tol)

    acc = abs(root - (a+b)/2);
    root = (a + b)/2;

    if(myFunc(a)*myFunc(root)<0)
        b = root;
    elseif(myFunc(b)*myFunc(root)<0)
        a = root;
    end
    i=i+1
end

disp(root)
disp(i)
```

Πόσες φορές θα τρέξει?

- Για **`tol = 1e-5, i=18`**
- Για **`tol = 1e-6, i=21`**
- Για **`tol = 1e-7, i=25`**
- Για **`tol = 1e-8, i=28`**
- Για **`tol = 1e-9, i=31`**

Χρήση **for**

```
for i=1:maxIter %να οριστεί από την αρχή

    acc = abs(root - (a+b)/2);
    root = (a + b)/2;

    if (acc<tol)
        break;
    elseif(myFunc(a)*myFunc(root)<0)
        b = root;
    elseif(myFunc(b)*myFunc(root)<0)
        a = root;
    end

end

disp(root)
```

```
a = -1;
b = 1;
root = a;
acc = 1;
tol = 1e-6;
maxIter=1000
```

Απευθείας υπολογισμός

Εντολή **`fzero`**

`fzero`(χειριστήριο συνάρτησης, διάστημα λύσης)

`fzero` (@myFunc,[-1,1])

Απευθείας υπολογισμός

Εντολή **fzero**

fzero(χειριστήριο συνάρτησης, διάστημα λύσης)

```
fzero (@myFunc,[-1,1])
```

Χειριστήρια συνάρτησης @

Μία συνάρτηση μπορεί να κληθεί με συνοπτικό τρόπο με τη βοήθεια ενός χειριστηρίου συνάρτησης (function handle)

Ένα χειριστήριο συνάρτησης κατασκευάζεται προσθέτοντας το σύμβολο at, @, πριν από το όνομα της συνάρτησης.

Παραδείγματα

$F(X)=X^3-2X^2+5$

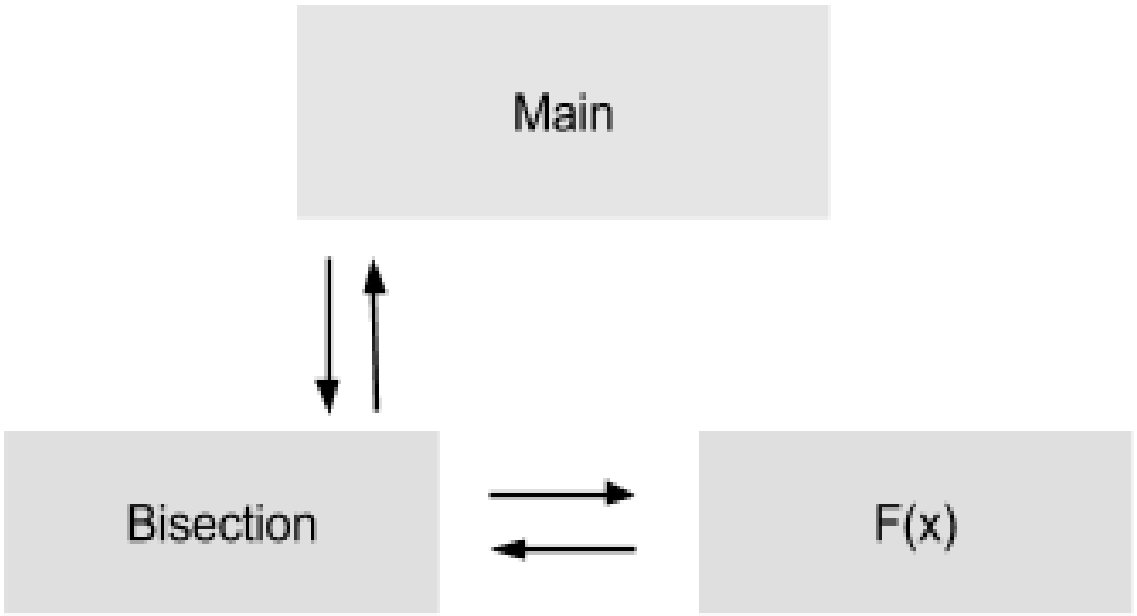
```
onoma_handle=@(x)x.^3 - 2*x – 5
Για F(91):
onoma_nandle(91)
  ans =  753394
```

Ρίζα της F(X)=0 κοντά στο μηδέν

```
fzero(onoma_handle, 0)
ans = -2.09455146754517
```

```
onoma_handle(ans)
ans =   1.562284284162227e-007
```

Και πως το μετατρέπω σε συνάρτηση;



Παράμετρος Εισόδου: Όρια Διαστήματος (a, b)
Παράμετρος Εξόδου: Λύση της Εξίσωσης

Παράμετρος Εισόδου: Αριθμός, x
Παράμετρος Εξόδου: Τιμή Συνάρτηση στο σημείο x

Και πως το μετατρέπω σε συνάρτηση;

Στο εργαστήριο!

Μία 2η περίπτωση: αριθμητική ολοκλήρωση

Τι είναι ολοκήρωση?

Ολοκλήρωση

Μέτρηση επιφάνειας οριζόμενης από συνάρτηση.

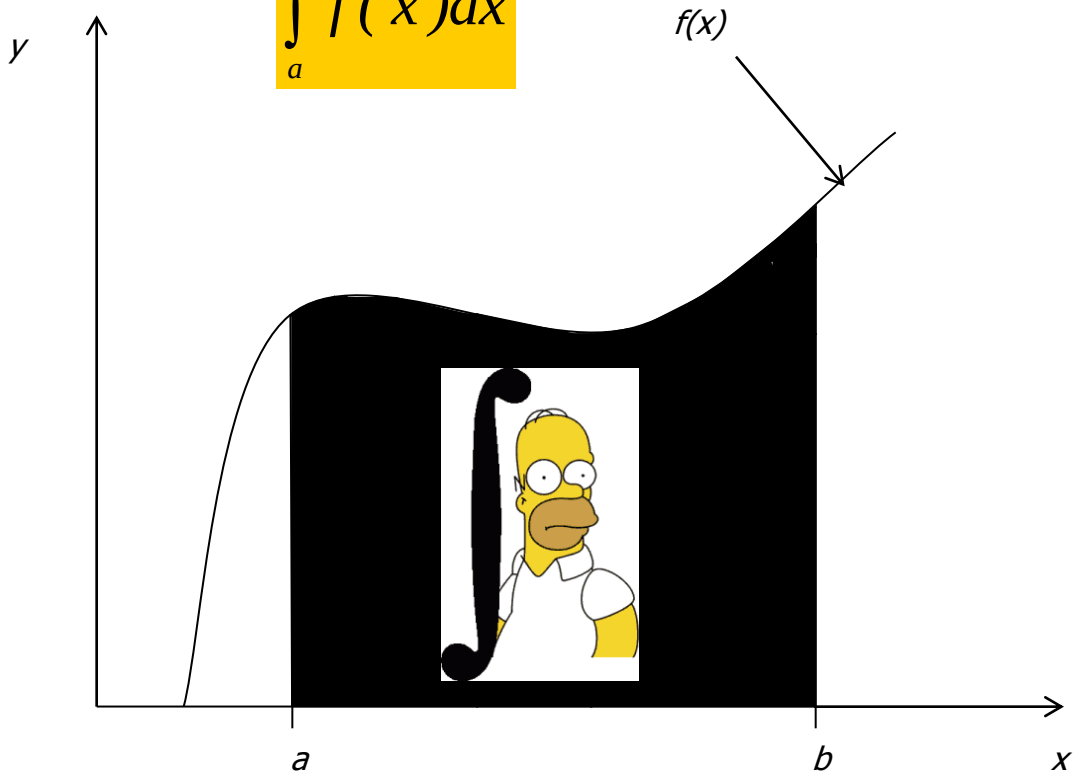
$$I = \int_a^b f(x)dx$$

Where:

a= κάτω όριο

b= άνω όριο

$$= \int_a^b f(x)dx$$



Μέθοδος εύρεσης ολοκληρώματος γεωμετρικών καταβολών

Η περιοχή κάτω από την καμπύλη είναι ένα τραπεζοειδές.

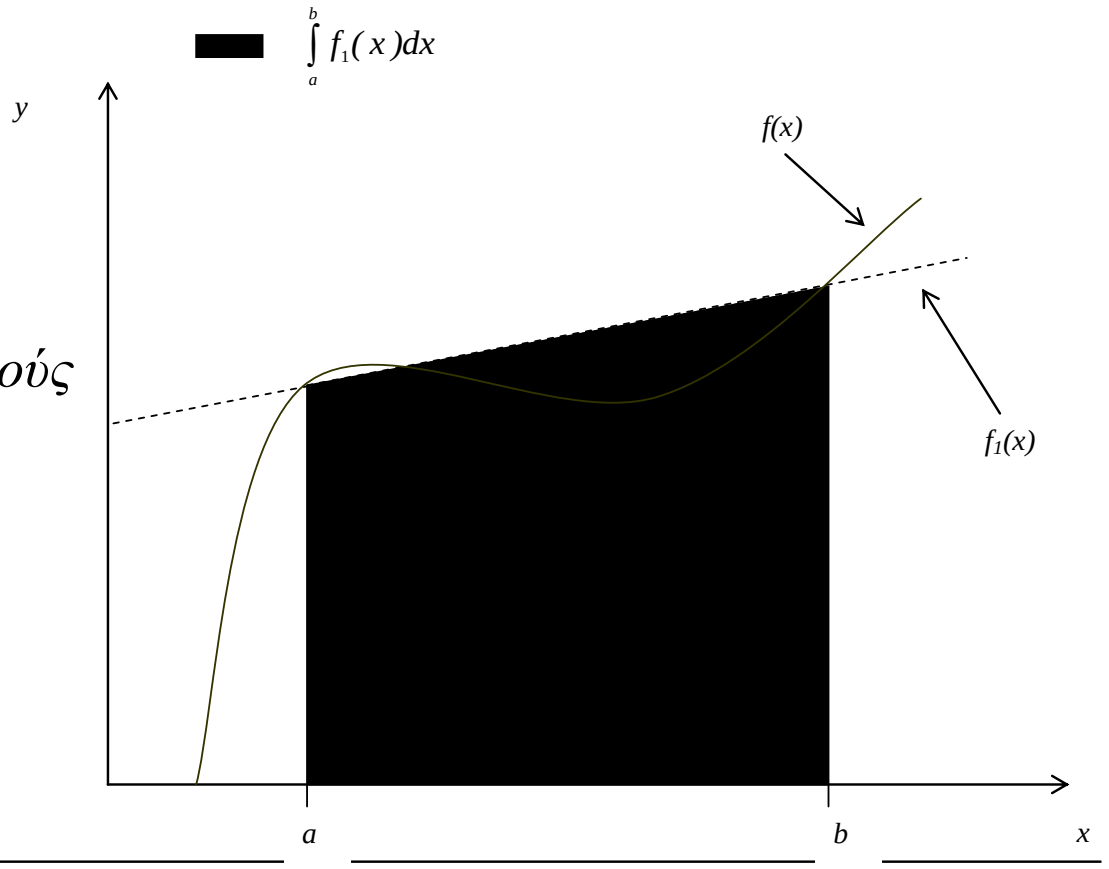
Το ολοκλήρωμα

$$\int_a^b f(x)dx \approx \text{Επιφάνεια τραπεζοειδούς}$$

$$= \frac{1}{2}(\text{πλευρές})(\text{ύψος})$$

$$= \frac{1}{2}\big(f(b)+f(a)\big)(b-a)$$

$$= (b-a)\left[\frac{f(a)+f(b)}{2}\right]$$



Γεωμετρική αναπαράσταση

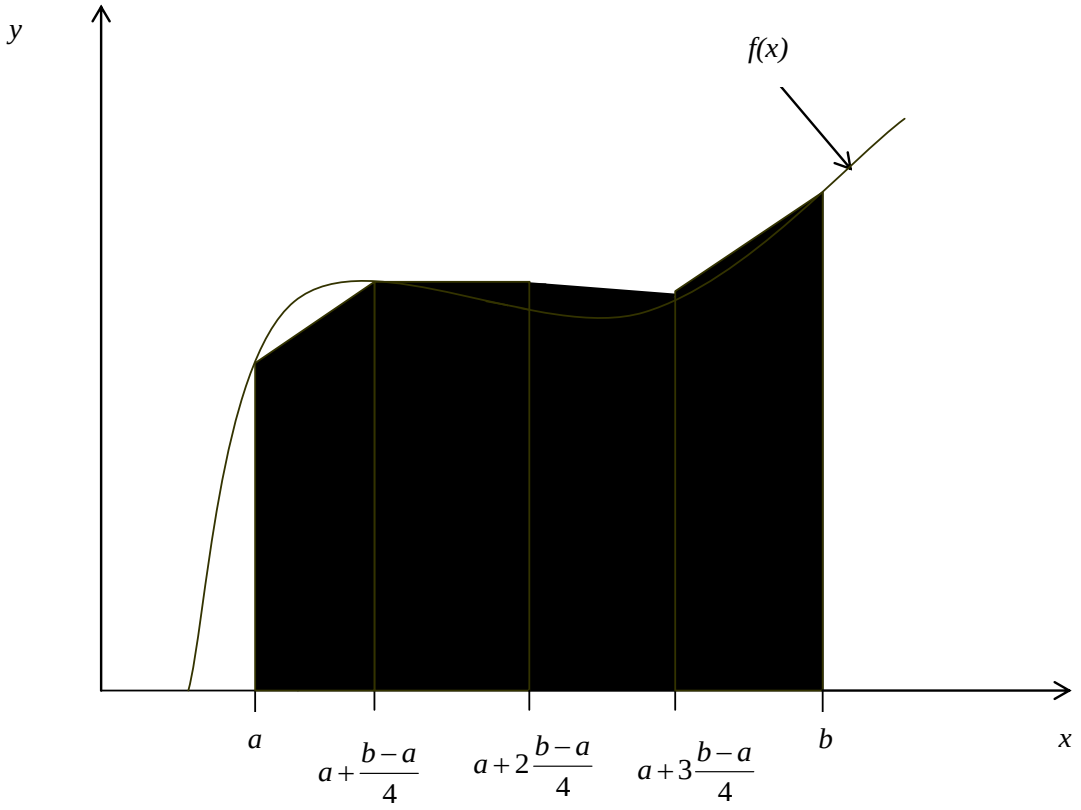
Κανόνας τραπεζίου

Χωρίζουμε σε ίσα τμήματα.Τό ύψος κάθε τμήματος είναι:

$$h = \frac{b-a}{n}$$

Το ολοκλήρωμα I είναι:

$$I = \int_a^b f(x)dx$$



Πολλαπλά(n=4) τμήματα

Multiple Segment Trapezoidal Rule

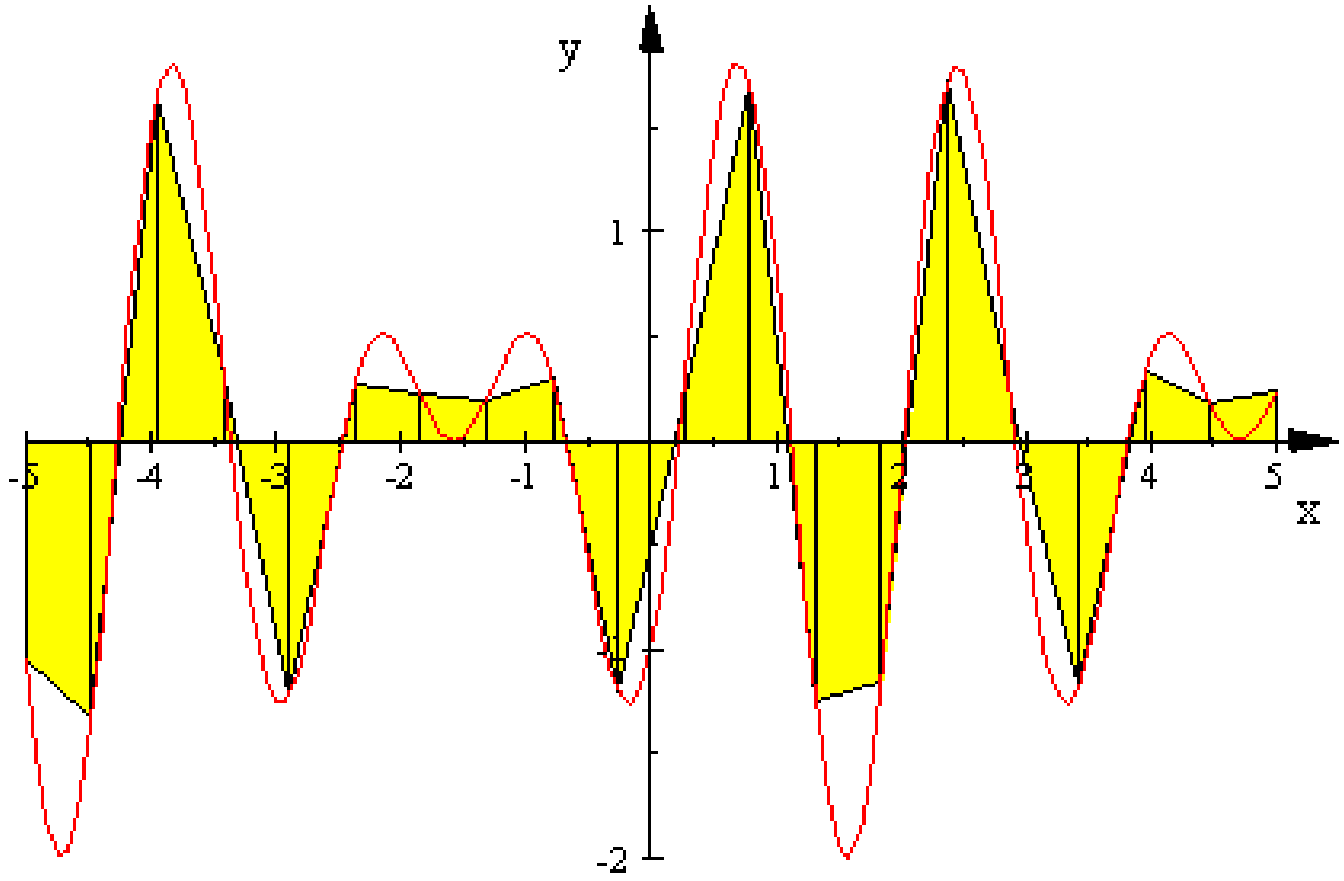
Το ολοκλήρωμα I μπορεί να διαιρεθεί σε h ολοκληρώματα:

$$\int_a^b f(x)dx = \int_a^{a+h} f(x)dx + \int_{a+h}^{a+2h} f(x)dx + \ldots + \int_{a+(n-2)h}^{a+(n-1)h} f(x)dx + \int_{a+(n-1)h}^{b} f(x)dx$$

Εφαρμόζοντας τον κανόνα του τραπεζίου σε κάθε τμήμα λαμβάνουμε:

$$\int_a^b f(x)dx = \frac{b-a}{2n}\left[f(a) + 2\left\{\sum_{i=1}^{n-1} f(a+ih)\right\} + f(b)\right]$$

Πολλαπλός κανόνας τραπεζίου



Κανόνας τραπεζίου

Συνοπτικά:

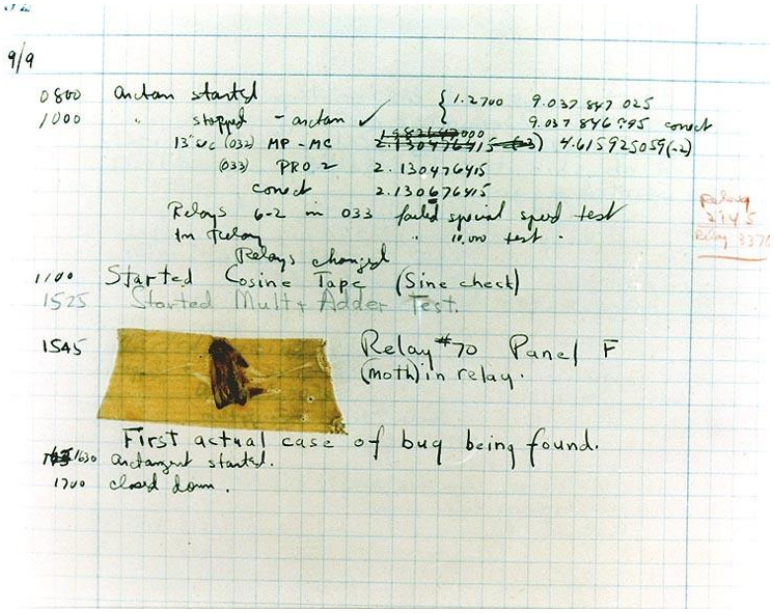
$$\int_a^b f(x)dx \cong c_1 f(a) + c_2 f(b)$$

$$= \frac{b-a}{2} f(a) + \frac{b-a}{2} f(b)$$

Και κάτι τελευταίο…

Το 1947 η Grace Hopper εργαζόμενη στον υπολογιστή Mark II του πανεπιστημίου του Harvard, διαπίστωσε ότι η προβληματική λειτουργία του υπολογιστή οφείλονταν σε ένα έντομο (bug) που είχε τρυπώσει σε κάποιο από τα κυκλώματα του υπολογιστή.

Ο όρος ***bug*** και ***debugging*** επικράτησε για τα σφάλματα του υπολογιστή και τη διαδικασία αποσφαλμάτωσης αντιστοίχως.



Το πρώτο «bug» τοποθετήθηκε στο ημερολόγιο του Mark II.

Τέλος

Αν. Καθ. Κωνσταντίνος Καρατζάς

Τμήμα Μηχανολόγων Μηχανικών ΑΠΘ

Σημείωμα Αναφοράς

- Copyright Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης, Κωνσταντίνος Καρατζάς. «Πληροφορική. Ενότητα 5: Α. Λογικές εκφράσεις και δείκτες (Β΄ μέρος- Διαχείριση πινάκων). Β. Αριθμητικές μέθοδοι επίλυσης προβλημάτων». Έκδοση: 1.0. Θεσσαλονίκη 2014. Διαθέσιμο από τη δικτυακή διεύθυνση: https://opencourses.auth.gr/courses/OCRS328/

Σημείωμα Αδειοδότησης

Τέλος ενότητας

Θεσσαλονίκη, Εαρινό Εξάμηνο 2014-2015

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ ΚΑΙ ΘΡΗΣΚΕΥΜΑΤΩΝ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Σημειώματα

Διατήρηση Σημειωμάτων

Οποιαδήποτε αναπαραγωγή ή διασκευή του υλικού θα πρέπει να συμπεριλαμβάνει:

- το Σημείωμα Αναφοράς
- το Σημείωμα Αδειοδότησης
- τη δήλωση Διατήρησης Σημειωμάτων
- το Σημείωμα Χρήσης Έργων Τρίτων (εφόσον υπάρχει)

μαζί με τους συνοδευόμενους υπερσυνδέσμους.