12. ΣΥΜΒΟΛΙΚΑ ΜΑΘΗΜΑΤΙΚΑ

Η **εργαλειοθήκη συμβολικών μαθηματικών** (Symbolic Math Toolbox) είναι προαιρετική (αυτό απλά σημαίνει ότι δεν συμπεριλαμβάνεται στο βασικό πακέτο της MATLAB και έτσι κάποιος πρέπει να πληρώσει το απαιτούμενο επιπλέον ποσό για να την αποκτήσει). Για να βεβαιωθείτε ότι το δικό σας πακέτο την περιλαμβάνει χρησιμοποιείστε την εντολή ver και ελέγξτε αν αυτή περιλαμβάνεται στο σχετικό κατάλογο.

Η εργαλειοθήκη συμβολικών μαθηματικών βασίζεται στον πυρήνα της MAPLE ο οποίος εκτελεί όλους τους συμβολικούς υπολογισμούς καθώς και τους υπολογισμούς **μεταβλητής ακρίβειας** (variable precision computations). Η MAPLE είναι γνωστό λογισμικό πακέτο για συμβολικούς υπολογισμούς.

Η εργαλειοθήκη χρησιμοποιεί ένα νέο τύπο δεδομένων, τα **συμβολικά αντικείμενα** (symbolic objects) τα οποία είναι επίσης γνωστά και σαν **αντικείμενα κλάσης** 'sym'. Τα συμβολικά αντικείμενα παράγονται με τις συναρτήσεις **sym** και **syms.**

Η συνάρτηση sym κατασκευάζει συμβολικούς αριθμούς, μεταβλητές και αντικείμενα. Η εντολή

```
s=sym(A)
```

κατασκευάζει από το A ένα συμβολικό αντικείμενο (τάξης 'sym'). Αν το A είναι αλφαριθμητικό, το αποτέλεσμα είναι ένας συμβολικός αριθμός ή μεταβλητή. Αν το A είναι αριθμητικό διάνυσμα ή πίνακας, το αντικείμενο s μας δίνει μια συμβολική αναπαράσταση των αντίστοιχων αριθμητικών τιμών. Έτσι

- η **`x=sym('x')`** δημιουργεί τη συμβολική μεταβλητή με όνομα 'x' και αποθηκεύει το αποτέλεσμα στο x
- η **`a=sym('alpha')`** δημιουργεί τη συμβολική μεταβλητή με όνομα 'alpha' και αποθηκεύει το αποτέλεσμα στο a
- η **`pi=sym('pi')`** δημιουργεί τον συμβολικό αριθμό pi
- η **`s=sym('sqrt(2)')`** δημιουργεί τη συμβολική μεταβλητή s για την $\sqrt{2}$
- η **`e=sym('exp(1)')`** δημιουργεί τη συμβολική μεταβλητή e για τον αριθμό e
- η **`delta=sym(0.001)`** δημιουργεί το συμβολικό αριθμό που αντιστοιχεί στο 0.001 και μάλιστα στην κλασματική μορφή 1/1000, αποφεύγοντας έτσι το εγγενές σφάλμα της αριθμητικής κινητής υποδιαστολής

Αν ορίσουμε τις μεταβλητές

```
>> x=sym('x')
x =
x
```

```
>> pi=sym('pi')
pi =
pi
>> delta=sym(0.001)
delta =
1/1000
```

η εντολή whos μας δίνει:

```
>> whos
  Name        Size              Bytes  Class    Attributes

  delta       1x1                 136  sym
  pi          1x1                 128  sym
  x           1x1                 126  sym
```

Ας κάνουμε τώρα μερικές συμβολικές πράξεις:

```
>> (pi+x)*delta
ans =
pi/1000 + x/1000
>> (pi+x)*(delta+x)
ans =
(pi + x)*(x + 1/1000)
```

Παρατηρούμε στην πρώτη περίπτωση ότι η MATLAB εκτέλεσε τις πράξεις με τη συμβολική σταθερά. Δεν έκανε το ίδιο όμως στη δεύτερη περίπτωση που είχαμε πολλαπλασιασμό συμβολικών παραστάσεων. Για το σκοπό αυτό χρησιμοποιείται η συνάρτηση expand όπως θα δούμε πιο κάτω.

Σημειώνουμε εδώ τη διαφορά των συμβολικών μεταβλητών **sym(0.01)** και **sym('0.01'):**

```
>> dd=sym(0.01)
dd =
1/100
>> ee=sym('0.01')
ee =
0.01
>>
>> (x+dd)*dd
ans =
1/100*x + 1/10000
>> (x+ee)*ee
ans =
0.01*x + 0.0001
```

Η εντολή **syms** κάνει ότι και η sym χωρίς να τυπώνει αποτελέσματα στο παράθυρο εντολών. Αυτή είναι ιδιαίτερα βολική όταν ορίζονται πολλές συμβολικές μεταβλητές του ίδιου τύπου. Για παράδειγμα η εντολή

```
>> syms alpha beta gamma
```

είναι ισοδύναμη με τις

```
>> alpha=sym('alpha'); beta=sym('beta'); gamma=sym('gamma');
```

Οι συμβολικές μεταβλητές χρησιμοποιούνται σε διάφορες παραστάσεις και σαν ορίσματα συναρτήσεων. Για παράδειγμα

```
>> syms x y
>> r=x^2+y^2
r =
x^2 + y^2
>> w=sqrt(r)
w =
(x^2 + y^2)^(1/2)
```

Η sym δέχεται και δεύτερο όρισμα εισόδου που καθορίζει τον τύπο μιας συμβολικής μεταβλητής όπως φαίνεται στον πιο κάτω πίνακα:

Εντολή	Αποτέλεσμα
`x=sym('x','real')`	Ο x είναι πραγματικός
`x=sym('x','positive')`	Ο x είναι θετικός
`x=sym('x','unreal')`	Ο x είναι μια τυπική μεταβλητή (ούτε πραγματικός, ούτε θετικός)

Η εντολή

```
>> syms x y z positive
```

είναι ισοδύναμη με τις

```
>> x=sym('x','positive'); y=sym('y','positive')
z=sym('z','positive');
```

Η πληροφορία ότι μια μεταβλητή είναι πραγματική ή θετική μπορεί να είναι κρίσιμη σε αρκετούς συμβολικούς υπολογισμούς. Ο τύπος 'unreal' χρησιμοποιείται συνήθως για να ακυρώσει έναν από τους άλλους δύο τύπους για μια δοσμένη μεταβλητή.

Παράδειγμα 12.1

Όπως φαίνεται με τις εντολές που ακολουθούν η MATLAB απλοποιεί τις συμβολικές παραστάσεις, συγκεντρώνει τους ομοβάθμιους όρους και μάλιστα τους κατατάσσει έτσι ώστε οι μεγαλύτερου βαθμού όροι να προηγούνται:

```
>> syms x
>> 3*x^2-x*x+x*x*x-2
ans =
x^3 + 2*x^2 - 2
>>
>> x+x^3-x^2
ans =
x^3 - x^2 + x
>>
>> 2*3*4*x*x*x
ans =
24*x^3
>>
>> x*x*x*cos(x)/x^2
ans =
```

```
x*cos(x)
```

Παράδειγμα 12.2

Θεωρούμε το συμβολικό πραγματικό πίνακα

$$A = \begin{bmatrix} a & b \\ -b & a \end{bmatrix}$$

Θα υπολογίσουμε συμβολικά τον ανάστροφο, τον αντίστροφο και την ορίζουσα του Α.

```
>> a=sym('a','real'); b=sym('b','real');
>> A=[a b; -b a]
A =
[  a, b]
[ -b, a]
>> A'
ans =
[ a, -b]
[ b,  a]
>> inv(A)
ans =
[ a/(a^2 + b^2), -b/(a^2 + b^2)]
[ b/(a^2 + b^2),  a/(a^2 + b^2)]
>> det(A)
ans =
a^2 + b^2
```

Θα βρούμε επίσης τις ιδιοτιμές του Α:

```
>> eig(A)
ans =
 a - b*i
 a + b*i
```

Μια συμβολική μεταβλητή μπορεί να αντιστοιχεί σε μια παράσταση, όπως

```
>> b=sym('2*k*pi*x/R')
b =
2*k*pi*x/R
```

μια συνάρτηση όπως

```
>> y=sym('3*x^2*sin(x)')
y =
3*x^2*sin(x)
```

ή ακόμα σε μια εξίσωση όπως

```
>> Elaw=sym('E = m*c^2')
Elaw =
E=m*c^2
```

Η συνάρτηση subs

Όταν βρούμε τη συμβολική λύση ενός προβλήματος, για παράδειγμα μιας εξίσωσης, επιθυμούμε συνήθως να την υπολογίσουμε ή να τη σχεδιάσουμε για κάποιες αντιπροσωπευτικές τιμές των παραμέτρων της. Αυτό μπορεί να γίνει με τη συνάρτηση **subs** η οποία αντικαθιστά τις τρέχουσες τιμές των μεταβλητών και

υπολογίζει την αντίστοιχη μορφή ή την τιμή μιας συμβολικής παράστασης. Θα πάρουμε σαν παράδειγμα τη συμβολική συνάρτηση

$$y = ae^{x} + be^{-x}$$

η οποία ορίζεται ως εξής:

```
>> syms a b x
>> y=a*exp(x)+b*exp(-x)
y =
a*exp(x) + b/exp(x)
```

Αν ορίσουμε μια συμβολική μεταβλητή τότε μπορούμε να βρούμε την ειδική μορφή της y γι'αυτή την περίπτωση με τη subs:

```
>> a=2; subs(y)
ans =
2*exp(x) + b/exp(x)
```

Σημειώνουμε ότι με την απόδοση τιμής σε μια συμβολική μεταβλητή, αυτή παύει να είναι συμβολική. Αυτό ισχύει για το a στο παράδειγμά μας:

```
>> whos
  Name        Size              Bytes  Class     Attributes

  a           1x1                   8  double
  ans         1x1                 184  sym
  b           1x1                 126  sym
  x           1x1                 126  sym
  y           1x1                 184  sym
```

Αν θέλουμε να διατηρήσουμε την a σαν συμβολική μεταβλητή, μπορούμε να χρησιμοποιήσουμε τη subs ως εξής:

```
>> subs(y,a,2)
ans =
2*exp(x) + b/exp(x)
```

Αν οι προσδιοριζόμενες μεταβλητές είναι δύο ή περισσότερες η subs συντάσσεται ως εξής:

```
subs(y, {a, b, …} {valueofa, valueofb, ... })
```

ή ισοδύναμα

```
subs(y, {a,b, ….} [valueofa, valueofb,  ..])
```

όπου το δεύτερο και το τρίτο όρισμα εισόδου είναι πίνακες κελίων (cell arrays).

Έτσι στο παράδειγμά μας μπορούμε να γράψουμε

```
>> subs(y,{a,b},{2,-1})
ans =
2*exp(x) - 1/exp(x)
>>
>> subs(y,{a,b},{3,2})
ans =
2/exp(x) + 3*exp(x)
```

ή ακόμα

```
>> subs(y,{a,b,x},[1,1,0])
ans =
     2
```

Παράδειγμα 12.3

Θα ορίσουμε πρώτα συμβολικά τη διπαραμετρική συνάρτηση

$$y = \sin(ax^b)$$

```
>> syms a b x
>> y=sin(a*x^b)
y =
sin(a*x^b)
```

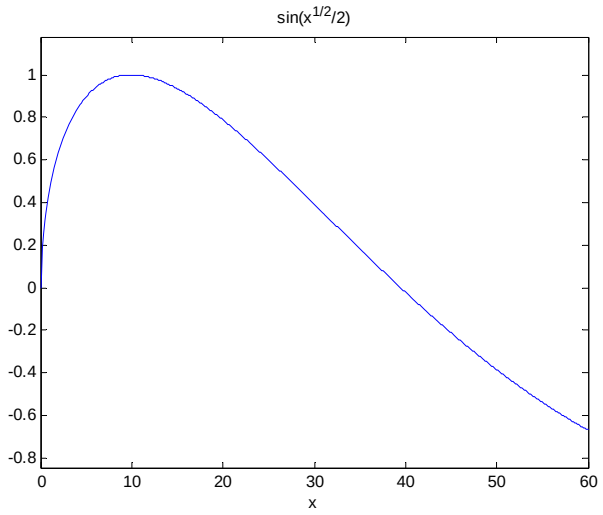
Μπορούμε τώρα εύκολα να ορίσουμε τώρα την

$$y = \sin\left(\frac{1}{2}\sqrt{x}\right)$$

με τη συνάρτηση subs. Αν για παράδειγμα θέλουμε να κάνουμε τη γραφική παράσταση της πιο πάνω συνάρτησης αρκεί να γράψουμε:

```
>> ezplot( subs(y,{a,b},{.5,.5}),[0,60] )
```

Παίρνουμε έτσι το ακόλουθο γράφημα:



Η συνάρτηση findsym

Η συνάρτηση **findsym** βρίσκει τις συμβολικές μεταβλητές σε μια συμβολική παράσταση ή πίνακα. Η εντολή

```
findsym(S)
```

επιστρέφει τις συμβολικές μεταβλητές της παράστασης S σε λεξικογραφική σειρά χωρίζοντάς τις με κόμματα. Οι σταθερές pi, i και j δεν θεωρούνται μεταβλητές. Επίσης η

```
findsym(S,n)
```

επιστρέφει τις n πλησιέστερες στο x ή το X μεταβλητές. Για παράδειγμα,

```
>> findsym( a*x+b *z +c *w)
ans =
a,b,c,w,x,z
>> findsym( a*x+b *z +c *w, 4)
ans =
x,w,z,c
```

Παράδειγμα 12.4

```
>> findsym( (x^2-a^2)/(x-a) )
ans =
a,x
>> findsym( (sin(x+h)-sin(x))/h )
ans =
h,x
>> findsym( pi* i + x* j +y)
ans =
x,y
```

Στις επόμενες παραγράφους θα συζητήσουμε τις σπουδαιότερες συναρτήσεις της MATLAB για συμβολικούς υπολογισμούς. Μπορούμε να δούμε ένα κατάλογο αυτών των εντολών πληκτρολογώντας

```
>> help symbolic
```

Σημειώνουμε ακόμα ότι υπάρχουν συναρτήσεις της MATLAB που εκτελούν διαφορετικές εργασίες αν τα ορίσματα εισόδου είναι αριθμητικά και άλλες αν αυτά είναι συμβολικά. Παράδειγμα μιας τέτοιας συνάρτησης είναι η **diff.** Με την εντολή

```
>> help diff
```

παίρνουμε τη βοήθεια για αριθμητικό όρισμα εισόδου ενώ με την

```
>> help sym/diff
```

παίρνουμε τη βοήθεια για τη συμβολική εκδοχή της εντολής.

12.1. Οι συναρτήσεις expand και simplify

Η συνάρτηση **expand** αναπτύσσει δυνάμεις πολυωνύμων καθώς επίσης τριγωνομετρικές, εκθετικές και λογαριθμικές συναρτήσεις. Για παράδειγμα

- η `expand((x+1)^3)` επιστρέφει x^3 + 3*x^2 + 3*x + 1
- η `expand(exp(x+y))` επιστρέφει exp(x)*exp(y)
- η `expand(sin(x+y))` επιστρέφει cos(x)*sin(y) + cos(y)*sin(x)
- η `expand(sinh(x+y))` επιστρέφει cosh(x)*sinh(y) + cosh(y)*sinh(x)

Παράδειγμα 12.1.1

Θα βρούμε τα αναπτύγματα των sin(a+b), cos(a+b) και tan(a+b):

```
>> syms a b
>> expand( sin(a+b) )
ans =
cos(a)*sin(b) + cos(b)*sin(a)
>>
>> expand( cos(a+b) )
ans =
cos(a)*cos(b) - sin(a)*sin(b)
>>
>> expand( tan(a+b) )
ans =
-(tan(a) + tan(b))/(tan(a)*tan(b) - 1)
```

Η τελευταία απάντηση διαβάζεται πιο εύκολα αν χρησιμοποιήσουμε την «όμορφη» εκδοχή της που μας δίνει η συνάρτηση **pretty**:

```
>> pretty(ans)

     tan(a) + tan(b)
  - -----------------
    tan(a) tan(b) - 1
```

Παράδειγμα 12.1.2

Θα υπολογίσουμε τις παραστάσεις

$$(x+y+z)^2 \quad \text{και} \quad (x+y+z)^3$$

```
>> expand( (x+y+z)^2 )
ans =
x^2 + 2*x*y + 2*x*z + y^2 + 2*y*z + z^2
>>
>> expand( (x+y+z)^3 )
ans =
x^3 + 3*x^2*y + 3*x^2*z + 3*x*y^2 + 6*x*y*z + 3*x*z^2 + y^3 +
3*y^2*z + 3*y*z^2 + z^3
```

Ένα χαρακτηριστικό όλων των λογισμικών πακέτων για συμβολικούς υπολογισμούς είναι το ότι σχεδόν πάντα απαιτείται μετεπεξεργασία (postprocessing) των αποτελεσμάτων προκειμένου αυτά να αναχθούν σε μια ισοδύναμη μεν αλλά πιο εύχρηστη μορφή. Η συνάρτηση **simplify** απλοποιεί κατά το δυνατό μια συμβολική παράσταση. Για παράδειγμα

```
>> simplify( exp(log(x^2)) )
```

```
ans =
x^2
>>
>> simplify( cos(x)^2 + sin(x)^2 )
ans =
1
>> simplify( (x+1)*x*(x-1) )
ans =
x^3 - x
```

Στη γενική περίπτωση η simplify έχει δύο ορίσματα εισόδου. Με την εντολή

```
simplify(S,n)
```

η MATLAB απλοποιεί τη συμβολική παράσταση S σε n βήματα (η προεπιλογή είναι n=50).

Παράδειγμα 12.1.3

Θα υπολογίσουμε τους τριγωνομετρικούς αριθμούς της γωνίας $\cos^{-1} x$:

```
>> sin( acos(x) )
ans =
(1 - x^2)^(1/2)
>> cos( acos(x) )
ans =
x
>> tan( acos(x) )
ans =
(1 - x^2)^(1/2)/x
>> pretty(ans)

        2 1/2
  (1 - x )
  -----------
       x
```

Με τη simplify μπορούμε εύκολα να υπολογίσουμε συμβολικά την

$$\tan\left(\frac{1}{2}\cos^{-1} x\right)$$

```
>> tan( acos(x)/2 )
ans =
tan(acos(x)/2)
>> simplify( tan( acos(x)/2 ) )
ans =
-(x - 1)/(1 - x^2)^(1/2)
```

Παράδειγμα 12.1.4

Ο αναγνώστης μπορεί εύκολα να βρει εύκολα τριγωνομετρικούς τύπους όπως για παράδειγμα ο

$$\tan(a+b+c) = \frac{\tan a + \tan b + \tan c - \tan a \tan b \tan c}{\tan a \tan b + \tan a \tan c + \tan b \tan c - 1}$$

```
>> syms a b c
>> pretty( expand(tan(a+b+c)) )
```

```
     tan(a) + tan(b) + tan(c) - tan(a) tan(b) tan(c)
- ------------------------------------------------
    tan(a) tan(b) + tan(a) tan(c) + tan(b) tan(c) - 1
```

Θα δείξουμε ακόμα ότι

$$\tan \quad = \frac{4\tan a - 4(\tan a)^3}{(\tan a)^4 - 6(\tan a)^2 + 1}$$

Πράγματι:

```
>> pretty(expand(tan(4*a)))

                         3
    4 tan(a) - 4 tan(a)
  ----------------------
        4            2
  tan(a)  - 6 tan(a)  + 1
```

Παράδειγμα 12.1.5

Θεωρούμε τον πίνακα

$$A = \begin{bmatrix} \cos t & -\sin t \\ \sin t & \cos t \end{bmatrix}$$

```
>> A=[cos(t) -sin(t); sin(t) cos(t)]
A =
[ cos(t), -sin(t)]
[ sin(t),  cos(t)]
```

Θα υπολογίσουμε και θα απλοποιήσουμε τους A^2 και A^{10}:

```
>> A^2
ans =
[ cos(t)^2 - sin(t)^2,  (-2)*cos(t)*sin(t)]
[     2*cos(t)*sin(t), cos(t)^2 - sin(t)^2]
>> simplify(A^2)
ans =
[ cos(2*t), -sin(2*t)]
[ sin(2*t),  cos(2*t)]
>> simplify(A^10)
ans =
[ cos(10*t), -sin(10*t)]
[ sin(10*t),  cos(10*t)]
```

Παράδειγμα 12.1.6

Θα δείξουμε ότι

$$\begin{vmatrix} a & b+c & a^2 \\ b & c+a & b^2 \\ c & a+b & c^2 \end{vmatrix} = (a-b)(a-c)(b-c)(a+b+c)$$

```
>> syms a b c
>> A=[ a b+c a^2; b c+a b^2; c a+b c^2]
A =
[ a, b + c, a^2]
[ b, a + c, b^2]
[ c, a + b, c^2]
>> det(A)
ans =
```

```
a^3*b - a^3*c - a*b^3 + a*c^3 + b^3*c - b*c^3
>> simplify(ans)
ans =
(a - b)*(a - c)*(b - c)*(a + b + c)
```

Η συνάρτηση simple

Η simple αναζητεί την απλούστερη μορφή μιας συμβολικής παράστασης ή ενός συμβολικού πίνακα, δείχνει όλες τις μικρότερου μήκους εκδοχές και επιστρέφει την συντομότερη εξ αυτών.

Παράδειγμα 12.1.7

Η συνάρτηση

$$F = \tan^2\left(\frac{1}{2}\cos^{-1} x\right) x(x-1)(x+1)$$

ορίζεται συμβολικά με την εντολή

```
>> F=sym('tan( acos(x)/2 )^2* x* (x-1)* (x+1)')
F =
tan(acos(x)/2)^2*x*(x-1)*(x+1)
```

Με την εντολή simplify η παράσταση απλοποιείται ως εξής:

```
>> simplify(F)
ans =
2*x^2 - x^3 - x
```

Με την εντολή simple παίρνουμε διαδοχικά τις ακόλουθες μορφές της F:

```
>> simple(F)
simplify:
x*tan(acos(x)/2)^2*(x - 1)*(x + 1)
radsimp:
x*tan(acos(x)/2)^2*(x - 1)*(x + 1)
simplify(100):
-x*(x - 1)^2
combine(sincos):
x*tan(acos(x)/2)^2*(x - 1)*(x + 1)
combine(sinhcosh):
x*tan(acos(x)/2)^2*(x - 1)*(x + 1)
combine(ln):
x*tan(acos(x)/2)^2*(x - 1)*(x + 1)
factor:
x*tan(acos(x)/2)^2*(x - 1)*(x + 1)
expand:
x^3*tan(acos(x)/2)^2 - x*tan(acos(x)/2)^2
combine:
x*tan(acos(x)/2)^2*(x - 1)*(x + 1)
rewrite(exp):
(x*(x - 1)*(x + 1)*(i - i*x + (1 - x^2)^(1/2))^2)/(x + i*(1 -
x^2)^(1/2) + 1)^2
rewrite(sincos):
(x*sin(acos(x)/2)^2*(x - 1)*(x + 1))/cos(acos(x)/2)^2
rewrite(sinhcosh):
-(x*sinh(-(i*acos(x))/2)^2*(x - 1)*(x + 1))/cosh(-(i*acos(x))/2)^2
rewrite(tan):
x*tan(acos(x)/2)^2*(x - 1)*(x + 1)
```

```
collect(x):
x^3*tan(acos(x)/2)^2 - x*tan(acos(x)/2)^2
mwcos2sin:
(x*sin(acos(x)/2)^2*(x - 1)*(x + 1))/cos(acos(x)/2)^2
ans =
-x*(x - 1)^2
```

Η συμβολική συνάρτηση factor

Η συμβολική συνάρτηση factor παραγοντοποιεί μια συμβολική παράσταση. Για παράδειγμα για την

$$x^3 - 6x^2 + 11x - 6$$

βρίσκουμε

```
>> factor( x^3 - 6*x^2 + 11*x - 6 )
ans =
(x - 3)*(x - 1)*(x - 2)
```

Ομοίως για την

$$x^{12} - 1$$

παίρνουμε:

```
>> factor(x^12-1)
ans =
(x - 1)*(x + 1)*(x^2 + x + 1)*(x^2 + 1)*(x^2 - x + 1)*(x^4 - x^2 +
1)
```

Η συνάρτηση collect

Η εντολή

```
collect(S,v)
```

συγκεντρώνει τους όρους της συμβολικής παράστασης S και τη γράφει σαν πολυώνυμο του v. Με την εντολή

```
collect(S)
```

η S γράφεται σαν πολυώνυμο της μεταβλητής που καθορίζεται από τη findsym (για παράδειγμα της x)

Παράδειγμα 12.1.8

Η συνάρτηση

$$S = x^2 - 4x^2 \cos y + 2x - x \cos^2 y + 3 \cos^2 y$$

ορίζεται συμβολικά με την εντολή

```
>> S=x^2-4*x^2*cos(y)+2*x-x*(cos(y))^2+3*(cos(y))^2
S =
x^2 - 4*x^2*cos(y) - x*cos(y)^2 + 2*x + 3*cos(y)^2
```

Θα συγκεντρώσουμε τους όρους της S θεωρώντας την πρώτα ως πολυώνυμο του x και μετά ως πολυώνυμο του cosy:

```
>> collect(S)
ans =
3*cos(y)^2 - x*(cos(y)^2 - 2) - x^2*(4*cos(y) - 1)
>>
>> collect(S,cos(y))
ans =
2*x - 4*x^2*cos(y) - cos(y)^2*(x - 3) + x^2
```

12.2. Όρια, παράγωγοι και ολοκληρώματα

12.2.1 Η συνάρτηση limit

Η συνάρτηση limit βρίσκει το όριο μας συμβολικής παράστασης. Για παράδειγμα μπορούμε να υπολογίσουμε το όριο

$$\lim_{x\to 0}\frac{\sin x}{x}$$

με την εντολή.

```
>> limit( sin(x)/x )
ans =
1
```

Αν θέλουμε να βρούμε το όριο μιας συμβολικής παράστασης F για x→a τότε γράφουμε

```
limit(F,x,a)
```

Οι διάφορες επιλογές της limit συνοψίζονται στον πιο κάτω πίνακα:

Εντολή	**Περιγραφή**
`limit(F)`	Όριο καθώς η πλησιέστερη στο x ανεξάρτητη μεταβλητή τείνει στο 0.
`limit(F,a)`	Όριο καθώς η πλησιέστερη στο x ανεξάρτητη μεταβλητή τείνει στο a.
`limit(F,x,a)`	Κανονικό όριο: $\lim_{x\to a} F$
`limit(F,x,a,'left')`	Αριστερό όριο: $\lim_{x\to a^-} F$
`limit(F,x,a,'right')`	Δεξιό όριο: $\lim_{x\to a^+} F$

Παράδειγμα 12.2.1

Θα υπολογίσουμε τα όρια

$$\lim_{x\to\infty}\frac{x^2+t}{2x^2-2x+t^2}$$

και

$$\lim_{t\to-\infty}\frac{x^2+t}{2x^2-2x+t^2}$$

```
>> limit( (x^2+t)/(2*x^2-2*x+t^2), inf )
ans =
1/2
>> limit( (x^2+t)/(2*x^2-2*x+t^2), t, inf )
ans =
0
```

Παράδειγμα 12.2.2

Θα υπολογίσουμε τα πλευρικά όρια

$$\lim_{x\to 0^-} \frac{1}{x}$$

και

$$\lim_{x\to 0^+} \frac{1}{x}$$

```
>> limit(1/x,x,0,'left')
ans =
-Inf
>> limit(1/x,x,0,'right')
ans =
Inf
>> limit(1/x,0)
ans =
NaN
```

Παράδειγμα 12.2.3

Θα υπολογίσουμε το όριο

$$\lim_{x\to a} \frac{x^3-a^3}{x-a}$$

```
>> limit( (x^2-a^2)/(x-a), a )
ans =
2*a
```

Παράδειγμα 12.2.4

Θα βρούμε την παράγωγο του sin(x) χρησιμοποιώντας τον ορισμό της παραγώγου:

$$\lim_{h\to 0} \frac{\sin(x+h)-\sin x}{h}$$

```
>> limit( (sin(x+h)-sin(x))/h, h, 0)
ans =
cos(x)
```

θα δούμε πιο κάτω ότι μπορούμε να υπολογίσουμε την παράγωγο μιας συμβολικής συνάρτησης με την συνάρτηση diff.

Παράδειγμα 12.2.5

Θεωρούμε την παράσταση

$$\sin(a+bt)e^{cw}$$

Η πλησιέστερη στο x μεταβλητή είναι η w:

```
>> findsym ( sin(a+b*t)* exp(c*w) , 3 )
```

```
ans =
w,t,c
```

Άρα με την εντολή

```
>> limit ( sin(a+b*t)* exp(c*w) )
ans =
sin(a + b*t)
```

υπολογίζεται το όριο

$$\lim_{w\to 0} \sin(a+bt)\, e^{cw} = \sin\,(a+bt)$$

Στο παράδειγμα που ακολουθεί θα υπολογίσουμε το όριο ενός συμβολικού πίνακα.

Παράδειγμα 12.2.6

Θα βρούμε το όριο του συμβολικού πίνακα

$$\left[1+\frac{1}{x} \quad \left(1+\frac{1}{x}\right)^{x} \quad \frac{1-ax^3}{1+bx^3}\right]$$

καθώς το x τείνει στο άπειρο.

```
>> S=[ 1+1/x, (1+1/x)^x, (1-a*x^3)/(1+b*x^3) ]
S =
[ 1/x + 1, (1/x + 1)^x, -(a*x^3 - 1)/(b*x^3 + 1)]
>> limit(S,inf)
ans =
[ 1, exp(1), -a/b]
```

12.2.2 Η συνάρτηση diff

Η εργαλειοθήκη συμβολικών μαθηματικών μας δίνει τη δυνατότητα παραγώγισης μιας συμβολικής συνάρτησης με τη συνάρτηση **diff**. Για παράδειγμα

```
>> syms x t
>> diff(4*x^3)
ans =
12*x^2
>> diff( cos(2*x) )
ans =
-2*sin(2*x)
>>
>> int( t^6 )
ans =
1/7*t^7
```

Παρατηρούμε ότι η x δεν είναι απαραίτητα η ανεξάρτητη μεταβλητή. Αν μια συμβολική έκφραση περιλαμβάνει περισσότερες από μια μεταβλητές, η παραγώγιση και η ολοκλήρωση γίνονται ως προς την αλφαβητικά πλησιέστερη προς το x. Η MATLAB μας επιτρέπει να ορίσουμε την επιθυμητή μεταβλητή ως δεύτερο όρισμα. Για παράδειγμα, για τον υπολογισμό της dx^a/dx χρησιμοποιούμε την

```
>> diff(x^a)
ans =
a*x^(a - 1)
```

ενώ για τον υπολογισμό της dx^a/da χρησιμοποιούμε την

```
>> diff(x^a,a)
ans =
x^a*log(x)
```

Στη γενική περίπτωση η παράγωγος

$$\frac{d^n f}{dx^n}$$

υπολογίζεται με την εντολή

`diff(f,x,n)` **ή** **`diff(f,n,x)`**

Παράδειγμα 12.2.7

Θα βρούμε τις πρώτες τέσσερις παραγώγους της

$$y = \log(a + bx)$$

```
>> syms a b x
>> for i=1:4
diff( log(a+b*x), i)
end
ans =
b/(a + b*x)
ans =
-b^2/(a + b*x)^2
ans =
(2*b^3)/(a + b*x)^3
ans =
-(6*b^4)/(a + b*x)^4
```

Είναι φανερό ότι με την diff μπορούμε εύκολα να υπολογίσουμε μερικές παραγώγους. Για παράδειγμα για την

$$\frac{\partial^3 f}{\partial y \partial x^2}$$

χρησιμοποιούμε την `diff(diff(f,2),y)`.

Παράδειγμα 12.2.8

Θα βρούμε τις μερικές παραγώγους πρώτης και δεύτερης τάξης της

$$f(x,y) = 4xy^3 + 12y^2 + 2xy - 3x + 1$$

```
>> syms x y
>> f=4*x*y^3+12*y^2+2*x*y+3*x+1
f =
4*x*y^3 + 12*y^2 + 2*x*y + 3*x + 1
>> fx=diff(f)
fx =
4*y^3 + 2*y + 3
>> fy=diff(f,y)
fy =
```

```
12*x*y^2 + 24*y + 2*x
>> fxx=diff(f,2)
fxx =
0
>> fyy=diff(f,y,2)
fyy =
24*x*y + 24
>> fxy=diff(fy)
fxy =
12*y^2 + 2
```

Παράδειγμα 12.2.9

Το m-file claplace.m υπολογίζει τη Λαπλασιανή μιας συνάρτησης f(x,y),

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

```
function D=claplace(f,x,y)
% CLAPLACE
% Calculates the Laplacian of f(x,y)
D=simplify( diff(f,x,2)+diff(f,y,2) );
% End
```

Για την

$$f(x,y) = e^{xy}$$

βρίσκουμε:

```
>> claplace(exp(x*y),x,y)
ans =
exp(x*y)*(x^2 + y^2)
```

12.2.3 Η συνάρτηση jacobian

Η συνάρτηση αυτή υπολογίζει τον ***Ιακωβιανό πίνακα*** (Jacobian matrix) ενός διανυσματικού πεδίου

$$\mathbf{F} = \mathbf{F}(\mathbf{x}) = (F_1(x_1, \cdots, x_n), \cdots, F_n(x_1, \cdots, x_n)),$$

ο οποίος ορίζεται ως εξής:

$$\mathbf{J} = \frac{\partial \mathbf{F}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial F_1}{\partial x_1} & \frac{\partial F_1}{\partial x_2} & \cdots & \frac{\partial F_1}{\partial x_n} \\ \frac{\partial F_2}{\partial x_1} & \frac{\partial F_2}{\partial x_2} & \cdots & \frac{\partial F_2}{\partial x_n} \\ \vdots & \vdots & & \vdots \\ \frac{\partial F_n}{\partial x_1} & \frac{\partial F_n}{\partial x_2} & \cdots & \frac{\partial F_n}{\partial x_n} \end{bmatrix}$$

Παρατηρούμε ότι το ij-στοιχείο του Ιακωβιανού πίνακα είναι το

$$J_{ij} = \frac{\partial F_i}{\partial x_j}$$

Στην περίπτωση που έχουμε βαθμωτό πεδίο

$$f = f(x_1, \cdots, x_n)$$

τότε ο Ιακωβιανός πίνακας αντιστοιχεί στην κλίση:

$$\nabla f = \left(\frac{\partial f}{\partial x_1}, \cdots, \frac{\partial f}{\partial x_n}\right)$$

Μια ιδιαίτερα χρήσιμη ποσότητα είναι η απόλυτη τιμή της ορίζουσας του Ιακωβιανού πίνακα που είναι γνωστή ως ***Ιακωβιανή ορίζουσα***.

Στη MATLAB ο Ιακωβιανός πίνακας ενός διανυσματικού ή βαθμωτού πεδίου F ως προς τις συντεταγμένες που ορίζονται σε κάποιο διάνυσμα v υπολογίζεται με την εντολή

```
jacobian(F,v)
```

Παράδειγμα 12.2.10

Θα βρούμε τον Ιακωβιανό πίνακα και την Ιακωβιανή ορίζουσα της συνάρτησης

$$F(x, y, z) = (x^2 + y^2, yz + zx, x + y$$

```
>> J=jacobian([x^2+y^2, y*z+z*x, x+y],[x,y,z])
J =
[ 2*x, 2*y,     0]
[   z,   z, x + y]
[   1,   1,     0]
>> det(J)
ans =
2*y^2 - 2*x^2
```

Παράδειγμα 12.2.11

Η συνάρτηση

$$F(r, \theta, z) = (r\cos\theta, r\sin\theta, z)$$

μετασχηματίζει τις κυλινδρικές σε καρτεσιανές συντεταγμένες. Θα βρούμε τον τον Ιακωβιανό πίνακα και την Ιακωβιανή ορίζουσα της συνάρτησης:

```
F>> syms r theta z
>> J=jacobian([r*cos(theta), r*sin(theta), z],[r,theta,z])
J =
[ cos(theta), -r*sin(theta), 0]
[ sin(theta),  r*cos(theta), 0]
[          0,             0, 1]
>> det(J)
ans =
r*cos(theta)^2 + r*sin(theta)^2
>> simplify(det(J))
ans =
r
```

12.2.4 Η συνάρτηση taylor

Με τη συνάρτηση taylor μπορούμε να υπολογίσουμε το ανάπτυγμα Taylor μιας συνάρτησης γύρω από δοσμένο σημείο. Στην απλούστερη εκδοχή, η

```
taylor(f)
```

υπολογίζει τους όρους έως τάξης 5 γύρω από το μηδέν (ανάπτυγμα Maclaurin). Αν θέλουμε να βρούμε τους πρώτους n όρους γύρω από το σημείο x0 χρησιμοποιούμε την γενική εκδοχή:

```
taylor(f,n,x0)
```

ή ακόμα την

```
taylor(f,t,n,t0)
```

αν η ανεξάρτητη μεταβλητή είναι η t. Άλλες πιθανές εκδοχές σύνταξης της συνάρτησης είναι οι `taylor(f,t)`, `taylor(f,x0)` και `taylor(f,n)`. Για παράδειγμα, για την sinx βρίσκουμε:

```
>> taylor(sin(x))
ans =
x^5/120 - x^3/6 + x
>> taylor(sin(x),10)
ans =
x^9/362880 - x^7/5040 + x^5/120 - x^3/6 + x
>> taylor(sin(x),pi/2,6)
ans =
(pi/2 - x)^4/24 - (pi/2 - x)^2/2 + 1
```

Παράδειγμα 12.2.12

Θα βρούμε το ανάπτυγμα Taylor ως προς w της

$$u = \frac{3}{w^2} log\left(\frac{\cosh w}{\cosh yw}\right)$$

Εδώ βολεύει να βρούμε πρώτα το ανάπτυγμα Maclaurin της

$$v = log\left(\frac{\cosh w}{\cosh yw}\right)$$

και μετά να πολλαπλασιάσουμε με $3/w^2$.

```
>> syms w x
>> v=simplify( taylor( log(cosh(w)/cosh(w*x)), w ) )
v =
(w^2*(x^2 - 1)*(w^2*x^2 + w^2 - 6))/12
>> u=simplify( v*3/w^2 )
u =
((x^2 - 1)*(w^2*x^2 + w^2 - 6))/4
```

Βρήκαμε δηλαδή ότι

$$u = \frac{1}{4}(x^2 - 1)(w^2x^2 + w^2 - 6) + O(w^3)$$

Ας γράψουμε τώρα τη u σαν ανάπτυγμα του w:

```
>> u=collect(u,w)
u =
(w^2*(x^2 - 1)*(x^2 + 1))/4 - (3*x^2)/2 + 3/2
```

Βρίσκουμε έτσι ότι

$$u = \frac{3}{2}(1 - x^2) - \frac{1}{4}(1 - x^4)w^2 + O(w^3)$$

12.2.5 Η συνάρτηση int

Με τη συνάρτηση **int** μπορούμε να υπολογίσουμε αόριστα και ορισμένα ολοκληρώματα. Για παράδειγμα

```
>> int(x^2)
ans =
x^3/3
```

ή

```
>> int( cos(a*pi*t) )
ans =
sin(pi*a*t)/(pi*a)
```

Παρατηρούμε ότι η σταθερά ολοκλήρωσης παραλείπεται. Όπως σημειώσαμε ήδη Παρατηρούμε αν μια συμβολική έκφραση περιλαμβάνει περισσότερες από μια μεταβλητές, η ολοκλήρωση γίνεται ως προς την αλφαβητικά πλησιέστερη προς το x. Για τον υπολογισμό του ολοκληρώματος

$$\int x^{\alpha}\ dx$$

χρησιμοποιούμε την

```
>> int( x^a )
ans =
piecewise([a = -1, ln(x)], [a <> -1, x^(a + 1)/(a + 1)])
```

ενώ για το ολοκλήρωμα

$$\int x^{a}\ da$$

χρησιμοποιούμε την

```
>> int( x^a, a)
ans =
x^a/ln(x)
```

Είναι βέβαια αξιοσημείωτο το ότι η MATLAB διάκρινε όλες τις πιθανές περιπτώσεις κατά τον υπολογισμό του πρώτου ολοκληρώματος.

Για να υπολογίσουμε ορισμένα ολοκληρώματα πρέπει φυσικά να προσδιορίσουμε τα όρια ολοκλήρωσης a και b. Οι διάφορες μορφές σύνταξης της εντολής int φαίνονται στον πιο κάτω πίνακα.

Εντολή	**Περιγραφή**

`int(f)` **ή** `int(sym(f))`	$\int f dx$ όπου x=findsym(f)
`int(f,v)`	$\int f dv$
`int(f,a,b)`	$\int_a^b f dx$ όπου x=findsym(f)
`int(f,v,a,b)`	$\int_a^b f dv$

Παράδειγμα 12.2.12
Θα βρούμε τα αόριστα ολοκληρώματα

$$\int e^{ax}\,dx = \frac{e^{ax}}{a} \qquad \text{και} \qquad \int xe^{ax}\,dx = \frac{e^{ax}}{a^2}(ax-1)$$

```
>> int( exp(a*x) )
ans =
exp(a*x)/a
>> int( x*exp(a*x) )
ans =
(exp(a*x)*(a*x - 1))/a^2
```

Θα βρούμε τώρα τα ολοκληρώματα:

$$\int x \sin ax\,dx, \qquad \int \frac{xdx}{x^2-a^2} \qquad \text{και} \qquad \int xJ_0(x)dx$$

```
>> int( x*sin(a*x) )
ans =
(sin(a*x) - a*x*cos(a*x))/a^2
>> int( x/(x^2-a^2) )
ans =
log(x^2 - a^2)/2
>> int( x*besselj(0,x) )
ans =
x*besselj(1, x)
```

Παράδειγμα 12.2.13
Θα βρούμε τα ορισμένα ολοκληρώματα

$$\int_0^1 lnx \ln(1+x)\,dx = 2 - 2ln2 - \frac{\pi^2}{12} \qquad \text{και} \qquad \int_0^\infty e^{-x^2}dx = \frac{\sqrt{\pi}}{2}$$

```
>> int( log(x)*log(1+x), 0, 1)
ans =
```

```
2 - pi^2/12 - log(4)
>> int( exp(-x^2), 0, inf)
ans =
pi^(1/2)/2
```

Παράδειγμα 12.2.14

Με τις εντολές

```
>> int( 1/sqrt(a^2-x^2), 0, a)
ans =
pi/2
>> int( log(1+tan(x)), 0, pi/4 )
ans =
(pi*log(2))/8
```

βρίσκουμε ότι

$$\int_0^{\alpha} \frac{dx}{\sqrt{a^2 - x^2}} = \frac{\pi}{2}$$

και

$$\int_0^{\pi/4} \log(1 + \tan x) = \frac{\pi}{8} log2$$

Μπορούμε τώρα να βρούμε την αριθμητική τιμή του ολοκληρώματος σε διπλή ακρίβεια με τη συνάρτηση **double**. Η συνάρτηση αυτή χρησιμοποιείται για παραστάσεις ή αποτελέσματα που δεν περιέχουν συμβολικές μεταβλητές. Στο παράδειγμα μας τώρα βρίσκουμε:

```
>> double(ans)
ans =
   0.272198261287950
```

Όπως και κάθε λογισμικό πακέτο για συμβολικούς υπολογισμούς, η MATLAB δεν μπορεί φυσικά να υπολογίσει ολοκληρώματα ή λύσεις που δεν έχουν κλειστή μορφή. Θα πάρουμε για παράδειγμα το ελλειπτικό ολοκλήρωμα δεύτερου είδους

$$\int \sqrt{2 - sin^2 x}\, dx$$

για το οποίο δεν παίρνουμε αποτέλεσμα.

```
>> syms x
>> int( sqrt(2-sin(x)^2) )
Warning: Explicit integral could not be found.
> In sym.int at 64
ans =
int((2 - sin(x)^2)^(1/2), x)
```

Ας δοκιμάσουμε να βρούμε το ορισμένο ολοκλήρωμα στο [0, π] παίρνουμε:

```
>> int( sqrt(2-sin(x)^2), 0, pi )
Warning: Explicit integral could not be found.
> In sym.int at 64
ans =
int((2 - sin(x)^2)^(1/2), x = 0..pi)
```

Αν και δεν μπορούμε να πάρουμε συμβολικό αποτέλεσμα, μπορούμε να έχουμε ένα αρκετά ικανοποιητικό αριθμητικό αποτέλεσμα (σε διπλή ακρίβεια) με τη συνάρτηση double:

```
>> double(ans)
ans =
   3.820197789027712
```

12.3. Επίλυση συμβολικών αλγεβρικών εξισώσεων

Με την εντολή **solve** μπορούμε να λύσουμε συμβολικές αλγεβρικές εξισώσεις. Με την

```
solve('eqn')
```

επιλύεται η εξίσωση eqn ως προς τη μεταβλητή με το πλησιέστερο στο x όνομα (δηλ. την findsym(eqn). Για παράδειγμα για την εξίσωση

$$ax + b = 0$$

βρίσκουμε:

```
>> syms a b x
>> solve(a*x+b)
ans =
-b/a
```

Αν επιθυμούμε να λύσουμε την εξίσωση ως προς τη μεταβλητή var, τότε καλούμε τη solve ως εξής:

```
solve('eqn',var)
```

όπου η var είναι συμβολική μεταβλητή ή συμβολοσειρά που προσδιορίζει την άγνωστη μεταβλητή. Έτσι αν λύσουμε την πιο πάνω εξίσωση ως προς τη μεταβλητή a παίρνουμε:

```
>> solve(a*x+b,a)
ans =
-b/x
```

Στην περίπτωση που δεν υπάρχει αναλυτική λύση η solve θα προσπαθήσει να βρει αριθμητική λύση. Σημειώνουμε επίσης ότι στα πιο πάνω παραδείγματα είχαμε εξίσωση της μορφής f(x)=0. Αν η εξίσωση είναι της μορφής f(x)=q(x) τότε πρέπει να τη γράψουμε σαν συμβολοσειρά μέσα σε τόνους. Για παράδειγμα

```
>> solve('a*x+b=a+x')
ans =
(a - b)/(a - 1)
```

Παράδειγμα 12.3.1

Θα βρούμε τις λύσεις της συμβολικής δευτεροβάθμιας εξίσωσης

$$ax^2 + bx + c = 0$$

```
>> syms a b c x
>> y=solve(a*x^2+b*x+c)
y =
 -(b + (b^2 - 4*a*c)^(1/2))/(2*a)
 -(b - (b^2 - 4*a*c)^(1/2))/(2*a)
```

Η απάντηση είναι ένα 2×1 συμβολικό αντικείμενο (διάνυσμα) με στοιχεία τις δύο λύσεις. Ας κάνουμε τώρα ένα απλό έλεγχο κατά πόσο τα στοιχεία του y ικανοποιούν πράγματι τη δευτεροβάθμια εξίσωση. Επειδή το y είναι διάνυσμα υπολογίζουμε την παράσταση κατά στοιχεία ως εξής:

```
>> a*y.^2+b*y+c
```

```
ans =
 c + (b + (b^2 - 4*a*c)^(1/2))^2/(4*a) - (b*(b + (b^2 -
4*a*c)^(1/2)))/(2*a)
 c + (b - (b^2 - 4*a*c)^(1/2))^2/(4*a) - (b*(b - (b^2 -
4*a*c)^(1/2)))/(2*a)
```

Το αποτέλεσμα που παίρνουμε δεν είναι μηδέν όπως αναμέναμε και φαίνεται αρκετά περίπλοκο. Ας εφαρμόσουμε όμως τη συνάρτηση simplify:

```
>> simplify(ans)
ans =
 0
 0
```

Για να βρούμε τις ρίζες της δευτεροβάθμιας εξίσωσης για δοσμένες τιμές των a, b και c χρησιμοποιούμε τη συνάρτηση subs. Για παράδειγμα,

```
>> subs(y, {a,b,c}, [1,0,-1] )
ans =
    -1
     1
>> subs(y, {a,b,c}, [1,3,2] )
ans =
    -2
    -1
```

Παράδειγμα 12.3.2

Θα βρούμε τις λύσεις της εξίσωσης

$$\frac{ax}{\sqrt{1-x^2}} - \tan(\cos^{-1} x) = 0$$

```
>> solve(a*x/sqrt(1-x^2)-tan(acos(x)))
ans =
   (1/(a + 1))^(1/2)
  -(1/(a + 1))^(1/2)
  (-1/(a - 1))^(1/2)
 -(-1/(a - 1))^(1/2)
```

Συστήματα αλγεβρικών εξισώσεων

Στην περίπτωση που έχουμε σύστημα αλγεβρικών εξισώσεων μπορούμε να καλέσουμε τη solve ως εξής

```
>> solve('eqn1','eqn2',...,'eqnN')
```

αν οι άγνωστοι είναι αυτοί που καθορίζονται με τη findsym. Διαφορετικά μπορούμε να χρησιμοποιήσουμε την

```
>> solve('eqn1','eqn2',...,'eqnN',var1,var2,...,varN)
```

Παράδειγμα 12.3.3

Θα βρούμε τα σημεία τομής της έλλειψης

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

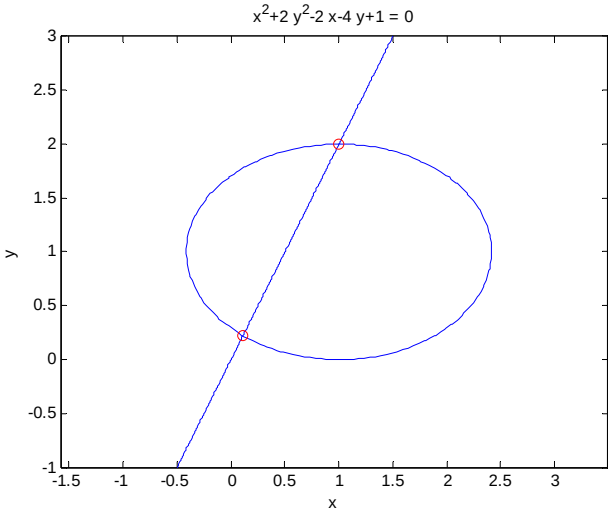
με την ευθεία y=2x.

```
>> [x,y]=solve('x^2+2*y^2-2*x-4*y+1','y-2*x')
x =
   1
 1/9
y =
   2
 2/9
```

Παρατηρούμε ότι τα x και y είναι διανύσματα.

Ας σχεδιάσουμε τώρα την έλλειψη, την ευθεία και τα σημεία τομής τους. Για τα τελευταία θα χρησιμοποιήσουμε τη συνάρτηση subs:

```
>> ezplot('y-2*x',[-1 3]); axis equal; set(gcf,'Color','w'); hold on
>> ezplot('x^2+2*y^2-2*x-4*y+1')
>> plot(subs(x),subs(y),'ro')
```



Παράδειγμα 12.3.4

Γνωρίζουμε ήδη ότι μπορούμε να βρούμε τις ιδιοτιμές του συμβολικού πίνακα

$$A = \begin{bmatrix} a & b \\ b & a \end{bmatrix}$$

με τη συνάρτηση eig:

```
>> syms a b x
>> eig([a b; b a])
ans =
 a + b
 a - b
```

Μπορούμε εναλλακτικά να επιλύσουμε τη χαρακτηριστική του εξίσωση,

$$\begin{vmatrix} a-\lambda & b \\ b & a-\lambda \end{vmatrix} = 0$$

με τις εντολές:

```
>> syms a b x
>> solve( det( [a-x b; b a-x] ) )
ans =
```

```
 a + b
 a - b
```

Παράδειγμα 12.3.5

Θεωρούμε τώρα το συμβολικό γραμμικό σύστημα

$$\begin{aligned} x + 2y &= a \\ 2x + y &= b \end{aligned}$$

Αυτό λύνεται εύκολα με αριστερή διαίρεση:

```
>> [1 2; 2 1]\[a b]'
ans =
 (2*b)/3 - a/3
 (2*a)/3 - b/3
```

Μπορούμε εναλλακτικά να χρησιμοποιήσουμε τη solve:

```
>> S=solve('x+2*y=a','2*x+y=b',x,y)
S =
    x: [1x1 sym]
    y: [1x1 sym]
```

Η μοναδική λύση για το x βρίσκεται στο x-πεδίο της δομής S. Ομοίως η λύση για το y βρίσκεται στο y-πεδίο της S.

```
>> S.x
ans =
(2*b)/3 - a/3
>> S.y
ans =
(2*a)/3 - b/3
```

Στην περίπτωση που έχουμε περισσότερες από μια λύση τότε τα S.x και S.y είναι διανύσματα.

Παράδειγμα 12.3.6

Θεωρούμε το μη γραμμικό σύστημα:

$$\left.\begin{aligned} x^2 - y^2 - z^2 &= 0 \\ x + y &= 1 \\ z^2 - 2z &= 3 \end{aligned}\right\}$$

Χρησιμοποιώντας τη solve βρίσκουμε:

```
>> S=solve(x^2-y^2-z^2,x+y-1,z^2-2*z-3)
S =
    x: [2x1 sym]
    y: [2x1 sym]
    z: [2x1 sym]
```

Μπορούμε να δούμε τις λύσεις για τα x, y και z ως εξής:

```
>>
>> S.x
ans =
 1
 5
>> S.y
```

```
ans =
  0
 -4
>> S.z
ans =
 -1
  3
```

Παρατηρούμε ότι το σύστημά μας έχει δύο λύσεις. Προγραμματιστικά, μπορούμε να βρούμε το πλήθος των λύσεων με την εντολή:

```
>> length(S.x)
ans =
     2
```

Αν θέλουμε να μελετήσουμε τη δεύτερη λύση, μπορούμε να χρησιμοποιήσουμε την εντολή:

```
>> s2=[ S.x(2) S.y(2) S.z(2)]
s2 =
[ 5, -4, 3]
```

Αν τέλος θέλουμε να κατασκευάσουμε τον πίνακα λύσεων γράφουμε:

```
>> SM=[S.x S.y S.z]
SM =
[ 1,  0, -1]
[ 5, -4,  3]
```

Οι συναρτήσεις digits και vpa

Στην περίπτωση που η solve αδυνατεί να βρει συμβολική λύση μιας εξίσωσης ή ενός συστήματος εξισώσεων, τότε προσπαθεί να βρει μια αριθμητική λύση με ακρίβεια 32 σημαντικών ψηφίων που είναι η προεπιλεγμένη ακρίβεια αριθμητικών υπολογισμών της Maple. Μπορούμε να δούμε αυτή τη προεπιλογή με την εντολή

```
>> digits
```

ή να αλλάξουμε την ακρίβεια σε n ψηφία με την εντολή

```
>> digits(n)
```

Μια σχετική συνάρτηση είναι η **vpa** (variable precision arithmetic) η οποία υπολογίζει αριθμητικά τα στοιχεία ενός πίνακα με τη χρήση αριθμητικής κινητής υποδιαστολής μεταβλητής ακρίβειας με ακρίβεια τόσων ψηφίων όσων και η τρέχουσα επιλογή της digits. Σημειώνεται ότι το αποτέλεσμα της πράξης είναι συμβολικό. Έτσι με τις εντολές

```
>> vpa(pi)
ans =
3.1415926535897932384626433832795
>> vpa(exp(1))
ans =
2.7182818284590455348848081484903
```

παίρνουμε τους αριθμούς π και e με ακρίβεια 32 ψηφίων. Μπορούμε να αλλάξουμε την ακρίβεια μιας συμβολικής παράστασης S σε D ψηφία με την εντολή

```
>> vpa(S,D)
```

Για παράδειγμα:

```
>> vpa(pi,50)
ans =
3.1415926535897932384626433832795028841971693993751
>> vpa(exp(1),50)
ans =
2.7182818284590455348848081484902650117874145507812
```

12.4 Ασκήσεις

12.1 Γράψτε μια ισοδύναμη εντολή της MATLAB για κάθε ακολουθία εντολών:

(α)
```
>> a=sym('a'); t=sym('t'); delta=sym('delta'); A=sym('A')
```

(β)
```
>> a=sym('a','real'); beta=sym('b','real'); c=sym('c','real');
```

12.2 Να υπολογιστούν οι ορίζουσες

$$\begin{vmatrix} 1 & x & yz \\ 1 & y & xz \\ 1 & z & xy \end{vmatrix}, \quad \begin{vmatrix} 1 & x & y+z \\ 1 & y & x+z \\ 1 & z & x+y \end{vmatrix}, \quad \begin{vmatrix} 1 & x & x^3 \\ 1 & y & y^3 \\ 1 & z & z^3 \end{vmatrix} \quad \text{και} \quad \begin{vmatrix} 1 & x & y \\ x & 1 & y \\ x & y & 1 \end{vmatrix}$$

12.3 Αν

$$s_j = a^j + b^j + c^j$$

υπολογίστε τις ορίζουσες

$$\begin{vmatrix} 3 & s_1 & s_2 \\ s_1 & s_2 & s_3 \\ s_2 & s_3 & s_4 \end{vmatrix} \quad \text{και} \quad \begin{vmatrix} 3 & s_1 & s_2 \\ s_1 & s_2 & s_3 \\ s_2 & s_3 & s_4 \end{vmatrix}$$

12.4 Να βρεθούν τα αναπτύγματα των

(α) $(x-y)^5$

(β) $(x+y)^6$

(γ) tan(a-b)

(δ) cot(a+b)

(ε) sinh(x+y)

(στ) cosh(x+y)

(ζ) tanh(x+y)

12.5 Παραγοντοποιείστε τα εξής:

(α) x^5-y^5

(β) x^6-y^6

(γ) $x^4+x^2y^2+y^4$

(δ) x^4+4y^4

12.6 Βρείτε τους τύπους που ισχύουν για τα πιο κάτω

(α) sin3a

(β) cos3a

(γ) tan3a

(δ) sinh3x

(ε) tanh3x

12.7 Υπολογίστε τα αόριστα ολοκληρώματα

(α)

$$\int xJ_1(x)dx$$

(β)

$$\int J_0(x)dx$$

(γ)

$$\int \frac{a+bx}{c+dx}\,dx$$

(δ)

$$\int \sqrt{\tan x}\,dx$$

12.8 Υπολογίστε τα ορισμένα ολοκληρώματα

(α)

$$\int_0^1 \frac{tan^{-1}x}{x^{3/2}}\mathrm{dx}$$

(β)

$$\int_0^{2\pi} \frac{dx}{(a+b\sin x)^2}$$

(γ)

$$\int_0^{\infty} \frac{\sin x\,dx}{\sqrt{x}}$$

(δ)

$$\int_0^{\infty} e^{-ax} J_0(bx)dx$$

12.10 Βρείτε τα αναπτύγματα Taylor των

(α) a^x

(β) $e^{\sin x}$

(γ) $e^x \sin x$

(δ) $\tanh^{-1}x$

12.11 Γράψτε ένα function m-file με όνομα taylor2 το οποίο θα υπολογίζει το ανάπτυγμα Taylor μιας συνάρτησης δύο μεταβλητών μέχρι και τους όρους $2^η$ς τάξης.

12.12 Βρείτε το ανάπτυγμα ως προς w μέχρι και τους όρους τέταρτης τάξης της

$$u = \frac{3}{w^2} log\left(\frac{\cosh w}{\cosh yw}\right)$$

12.13 Βρείτε το ανάπτυγμα ως προς w μέχρι και τους όρους τέταρτης τάξης της

$$u = \frac{8}{w^2} log\left(\frac{\mathrm{I}_0(w)}{\mathrm{I}_0(yw)}\right)$$

όπου I_0 η τροποποιημένη συνάρτηση Bessel πρώτου είδους και μηδενικής τάξης.

12.14 Βρείτε το ανάπτυγμα ως προς ε μέχρι και τους όρους τέταρτης τάξης της

$$p = \frac{1}{e}[I_0(\epsilon\alpha r)e^{-\varepsilon z} - 1]$$

όπου I_0 η τροποποιημένη συνάρτηση Bessel πρώτου είδους και μηδενικής τάξης.