Lesson 4
the symbolic maths toolbox

Instructor:
ir drs E.J Boks
Elecrical Engineering
Embedded Systems Engineering
phone:(026) 3658173
Room: D2.03
e-mail:ewout.boks@han.nl

Answers last session

```
function output = sigfunc(frequency,time)

% Demonstration function for HAN MSCS course

% This plots a number of goniometric functions

% (c)2003 E.J Boks, Hogeschool van Arnhem en
      Nijmegen

omega=2*pi*frequency;

output = 20*cos(omega*time).^2 - 15*(sin((3*omega-
      (2/3)*pi)*time)).^3;
```

```
% finding all the zeros in a sampled signal
% Matlab course MSCS
% (c)2003 Hogeschool van Arnhem en Nijmegen

frequency=4;
sampling=5*frquency;
t=[0:1/sampling:3];

y=sigfunc(frequency,t)-30;

yhandle = @y;

tzero = fzero(yhandle,1)
```

Answers last session

Observe the output – multiple frequencies due to sqr cosine and 3rd power sine :

```
sampling=500;

points=512;

t=[0:1/sampling:3];

frequency=200;


% het oorspronkelijke signaal

y1=sigfunc(frequency,t);


% Het vervuilde signaal

y2=sigfunc(frequency,t)+20*rand(size(t));
```

Answers last session

```
freqdata = fft(y2,points);


power = freqdata.*conj(freqdata)/points;
f=sampling*(0:points/2)/points;


subplot(3,1,1), plot(t,y1);
subplot(3,1,2), plot(t,y2);
subplot(3,1,3), plot(f,power(1:1+points/2));
```

Symbolic Math(s) toolbox

What is the Symbolic Math Toolbox?

The Symbolic Math Toolbox incorporates symbolic computation into the numeric environment of MATLAB.

Symbolic computation means that the tool attempts to resolve mathematical expressions non-numerically, e:g

$$x^2 = 3a + b$$
$$x = \sqrt{(3a+b)} \vee x = -\sqrt{(3a+b)}$$

These toolboxes supplement MATLAB numeric and graphical facilities with several other types of mathematical computation, which are summarized in following table.

Symbolic Math toolbox

Facilities :

- Solving calculus
- Solving Linear Algebra
- Simplify mathematical expressions
- Solving equations
- Perform mathematical transformations
- Variable precision arithmetic

Symbolic start

- The toolbox relies on *symbolic* variables (as opposed to ordinary Matlab numeric variables).
- Symbolic variable creation:
  - Simple syntax: syms <var_1> <var_2> ... <var_n>
  - Extended syntax: <var> = sym('label')
- When a variable must be declared real, ie within the bounds < -∞ , ∞> , add the word real to the above declaration.

Run a demo

- Run the symbolic math demo *calculus*
- Go the cmd line: type demos and select the *symbolic math toolbox*, then the *calculus* demo.
- Start the demo and examine the cmds and syntax carefully.

Calculus

Mathematical Operator	MATLAB Command
$\frac{df}{dx}$	`diff(f)` or `diff(f,x)`
$\frac{df}{da}$	`diff(f,a)`
$\frac{d^2f}{db^2}$	`diff(f,b,2)`
$J = \frac{\partial(r,t)}{\partial(u,v)}$	`J = jacobian([r:t],[u,v])`

Mathematical Operation	MATLAB Command
$\lim_{x \to 0} f(x)$	`limit(f)`
$\lim_{x \to a} f(x)$	`limit(f,x,a)` or `limit(f,a)`
$\lim_{x \to a-} f(x)$	`limit(f,x,a,'left')`
$\lim_{x \to a+} f(x)$	`limit(f,x,a,'right')`

Mathematical Operation	MATLAB Command
$\int x^n dx = \frac{x^{n+1}}{n+1}$	`int(x^n)` or `int(x^n,x)`
$\int_0^{\pi/2} \sin(2x)dx = 1$	`int(sin(2*x),0,pi/2)` or `int(sin(2*x),x,0,pi/2)`

$g = \cos(at+b)$
$\int g(t)dt = \sin(at+b)/a$
	`g = cos(a*t + b)`
	`int(g)` or
	`int(g,t)`

$\int J_1(z)dz = -J_0(z)$	`int(besselj(1,z))` or `int(besselj(1,z),z)`

Linear Algebra

Basic algebraic operations on symbolic objects are the same as operations on MATLAB objects of type double. This is illustrated in the following example:

The Givens transformation produces a plane rotation through the angle t. The statements

```
syms t;

G = [cos(t) sin(t); -sin(t) cos(t)]
```

create this transformation matrix.

```
G =

[  cos(t),  sin(t) ]

[ -sin(t),  cos(t) ]
```

Applying the Givens transformation twice should simply be a rotation through twice the angle. The corresponding matrix can be computed by multiplying G by itself or by raising G to the second power. Both

```
A = G*G  and A = G^2  produce

A =

[cos(t)^2-sin(t)^2,   2*cos(t)*sin(t)]

[ -2*cos(t)*sin(t), cos(t)^2-sin(t)^2]
```

Simplifying expressions

- The toolbox can aid in the process of mathematical simplifications with:
  - collect() : transform a sequence of factors into a polynominal. It collects all coefficients with the same power of x.

  Example: collect((x-1)*(x-2)*(x-3))
  - expand() : distributes products over sums and applies other identities when applicable.

  Example: expand(exp(a+b))
  - Horner() : transform a polynominal into the horner representation.

  Example: horner(x^3-6*x^2+11*x-6)

Simplifying expressions

- The toolbox can aid in the process of mathematical simplifications with:
  - factor() : opposite of collect() .
  - simplify() : the 'Swiss army knife' of simplification.
  - simple() : reduce and expression to the *smallest* number of arguments (as opposed to an *optimal* simplification).

Substitutions

- subexpr
  - Rewrites a symbolic expression in terms of common subexpressions.
  - Syntax:
  - [Y,SIGMA] = subexpr(X,SIGMA)
  - [Y,SIGMA] = subexpr(X,'SIGMA')
- subs
  - Symbolic substitution in a symbolic expression or matrix
  - Syntax
  - R = subs(S)
  - R = subs(S,old,new)

Equation solving

- Solve(S) attempts to find the value for which the symbolic variable S is equal to zero.
- syntaxes:
  - g = solve(eq)
  - g = solve(eq,var)
  - g = solve(eq1,eq2,...,eqn)
  - g = solve(eq1,eq2,...,eqn,var1,var2,...,varn)

solving a differential equation

- dsolve() has the same function as solve() but is used to solve differential equations.
- Syntax:
  - r = dsolve('eq1,eq2,...', 'cond1,cond2,...', 'v')
  - r = dsolve('eq1','eq2',...,'cond1','cond2',...,'v')

Compose and inverse a function

- compose(): compose a function using other functions:
- Syntax
  - compose(f,g)
  - compose(f,g,z)
  - compose(f,g,x,z)
  - compose(f,g,x,y,z)
- finverse(): attempts to find the inverse of a function:
- Syntax:
  - g = finverse(f)
  - g = finverse(f,u)

Exercise 1

- Calculate the 2nd differential of :

$$f(x)=\cos\frac{(3x-2)}{x^2-1}$$

- plot the result.
- Try to determine the original function f by integrating the result twice. Compare the result with the original function, either graphically or mathematically.

Exercise 2

- assume the following matrix A, compute the eigenvalues and eigenvectors of A:

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

- use subexpr() to simplify the expression for the eigenvectors by subtituting a value *S*.
- Then, substitute *S* into the expression for the eigenvalues using subs().

Exercise 3

- Solve this differential equation:

$$\frac{d^2 y}{dx^2} + 2y(x) = x * y$$

$$y(0) = 0,\ y(3) = \frac{1}{\pi} K_{\frac{1}{3}}(2\sqrt{3})$$

In case you don't understand how to model this equation, try looking in Matlab's help section.

the *K* denotes the Bessel function of the second kind ⇒see Matlab help on this.

Symbolic Math toolbox

End of Session four