RAL

Algebraic and robust control

Introduction to Symbolic math toolbox

Algebraic theory

- Teacher: Ing. Lukáš Zezula
- Lukas.Zezula@ceitec.vutbr.cz
- Office no. T12/SE 2.136
- Consultation: by mail

Rules for evaluation:

- 70 points exam
  - Assessment awarded after receiving 10+ points from exercise classes
- 30 points computer exercises
  - 1 microproject: algebraic control – 15
  - 1 microproject: robust control – 15

  - +1 point – activity during exrcises
  - –2 points – second and following absence without the apology

Symbolic Math Toolbox

- **Symbolic Math Toolbox** allows you to perform symbolic calculations in the MATLAB that is otherwise oriented more for numerical calculations.

https://www.mathworks.com/products/symbolic.html..html

sym ('')

- Create symbolic variables, expressions, functions, matrices

- ***x = sym('x')***
- creates symbolic variable x.

- ***r = sym('r','real')***
- creates symbolic variable r that is the type of real.

- ***k = sym('k','positive')***
- creates real positive symbolic variable k.

syms

- ***syms x y z***
- definition of several symbolic variables, same as commands
  x = sym('***x***'), y = sym('***y***'), z = sym('***z***')

- ***syms x y z real***
- definition of several real symbolic variables

- ***syms x y z positive***
- Definition of several positive symbolic variables

symvar

- ***symvar(S)***
- searches for symbolic variable in the symbolic expression and return them in a cell array.

```
>> syms y x z
>> c=10;
>> symvar(y*(4+3*i) + 6*j+2*x+c)
ans =[ x, y]
```

diff

- ***diff(S)***
- calculates derivative according to a symbolic variable
- (if there are more than one symbolic variable in the expression **S**, the derivative is calculated with respect to the first variable that the command ***symvar()*** would find)

- ***diff(S,y)***
- derivative is calculated with respect to y.

- ***diff(S,n)***
- calculates the n-th derivative.

diff – examples

Commands	Results
>> syms x y	
>>S=3*x^2+7*y+exp(x*y)	S =7*y+ exp(x*y) + 3*x^2
>> diff(cos(x))	ans =-sin(x)
>> diff(S)	ans =6*x + y*exp(x*y)
>> diff(S,x)	ans =6*x + y*exp(x*y)
>> diff(S,y)	ans =x*exp(x*y) + 7
>> diff(S,y,2)	ans =x^2*exp(x*y)

int

- ***int(S)***
- indefinite integral of **S** with respect to its symbolic variable as defined by ***symvar()***.

- ***int(S,y)***
- indefinite integral of **S** with respect to symbolic variable y.

- ***int(S,x,a,b)***
- definite integral of symbolic expression **S** with respect to x from a to b.

int – examples

Commands	Results
>> syms x y	
>> int(1/(1+x^2))	ans =atan(x)
>> int(sin(x*y),y)	ans =-cos(x*y)/x
>> int(x*log(1+x),0,1)	ans =1/4

pretty

- ***pretty(S)***
- ***Pretty()*** print a symbolic expression in a clear form.

Commands	Results
>> syms x c m	
>> B=x^2+c/m*3+m	B =m + (3*c)/m + x^2
>> pretty(B)	$m + \frac{3\,c}{m} + x^2$

solve

- ***solve(eqn1, eqn2, …, eqnN)***
- symbolic solution of algebraic equations.

- ***solve(eqn1, eqn2, …, eqnN, var1, var2, …, varN)***
- symbolic solution of set of algebraic equations with respect to symbolic variables.

- ***solve(eqn1, …, eqnN, var1, …, varN, 'ReturnConditions', true)***
- symbolic solution of set of algebraic equations. The command returns the conditions under which the solutions are valid.

solve – examples

Commands	Results
>> solve(x^2+5*x+6 == 0)	ans = –3 –2
>> solve(p*sin(x) == r, x)	ans =asin(r/p) pi – asin(r/p)
>> [a,b] = solve(x^2 + x*y + y == 3, x^2 – 4*x + 3 == 0)	a = 1 3 b =1 –3/2

subs

- ***subs(S, old, new)***
- symbolic substitution, replaces **old** with **new** in the symbolic expression **S**.

- ***subs(S, new)***
- replaces the default symbolic variable in expression **S** with **new**. Default variables are defined by the ***symvar()*** command.

subs – examples

Commands	Results
>> syms a, b	
>> subs(a + b, a, 10)	ans = b + 10
>> subs(a*b^2, a*b, 2)	ans = 2*b
>> syms x y a	
>> symvar(x + y, 1)	ans = x
>> subs(x + y, a)	ans = a + y

dsolve

- ***dsolve(eqn1, …, eqnN, var1, ..., varN')***
- Symbolic solution of ordinary differential equations.

- By default, the independent variable is **'t'**. The independent variable may be changed from **'t'** to some other symbolic variable by including that variable as the last input argument.

- the symbol **D** denotes the derivative of the variable according to **'t‘**.
- the initial conditions can be defined as y(0)=a, Dy(0)=b, ….

dsolve – examples

$$x'(t) + x = 0; x(0) = 1$$

```
>>syms x(t)
 dsolve(Dx+x == 0 ,x(0) == 1)

ans = exp(-t)
```

$$y'''(t) + 3y''(t) + 7y'(t) + 5y(t) = 10e^{-t},$$

$$y''(0) = -1, y'(0) = 1, y(0) = 2$$

```
>> syms y(t)
>> Dy = diff(y); D2y = diff(y,2); D3y = diff(y,3);
>> y=dsolve(D3y+3*D2y+7*Dy+5*y == 10*exp(-t), D2y(0)==-1,
   Dy(0)==1,y(0)==2)

y =(11*exp(-t))/4 + (5*t*exp(-t))/2 - (3*cos(2*t)*exp(-t))/4 + (sin(2*t)*exp(-t))/4
```

simplify

- ***simplify(S)***
- simplifies each element of the symbolic expression **S**.

```
>>syms x y t
>> simplify(sin(x)^2 + cos(x)^2)
ans =1

>> simplify(exp(t*log(sqrt(x+y))))
ans =(x + y)^(t/2)

>> S=simplify(2*cos(x)^2-sin(x)^2)
S = 2 - 3*sin(x)^2
```

collect

- ***collect(S,v)***
- rewrites symbolic expression **S** in terms of the powers of **v**.

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

>> f = -1/4*x*exp(-2*x)+3/16*exp(-2*x);
>> collect(f, exp(-2*x))
 ans =(3/16 - x/4)/exp(-2*x)
```

expand

- ***expand(S)***
- **expand()** is most often used on polynomials, writes each element of a symbolic expression **S** as a product of its factors.

```
>> expand((x+1)^3)
ans =x^3 + 3*x^2 + 3*x + 1

>> expand(sin(x+y))
 ans =cos(x)*sin(y) + cos(y)*sin(x)
```

factor

- ***factor(S,x)***
- returns an array of factors **S**, where **x** specifies the variables of interest.
- the number is decomposed into multiples of prime numbers.

```
>> factor(x^9-1)
ans = [ x - 1, x^2 + x + 1, x^6 + x^3 + 1]
```

```
>> factor(sym('67890'))
 ans =[ 2, 3, 5, 31, 73]
```

laplace

$$F(s) = \int_0^\infty f(t)e^{-st}\,dt$$

- ***L=laplace(F)***
- the command performs a Laplace transform of the symbolic expression **F** with default variable **t**. function.
- the default return is a function of operator **s**.

laplace – examples

Commands	Results
>> syms s t a f(t)	
>> laplace(exp(-3*t))	ans =1/(s + 3)
>> laplace(sym(1))	ans =1/s
>> laplace(t)	ans =1/s^2
>> laplace(cos(a*t))	ans =s/(a^2 + s^2)
>> laplace(t^5)	ans =120/s^6
>> laplace(diff(f))	ans =s*laplace(f(t), t, s) - f(0)

ilaplace

- ***F=ilaplace(L)***
- the command performs the inverse Laplace transform of the symbolic **L** with default variable **s**. The default return is a function of **t**.

```
>> syms t s a
>> ilaplace(1/(s-1))
ans =exp(t)

>> ilaplace(1/(s-a))
 ans =exp(a*t)
```

fourier

$$F(j\omega) = \int_{-\infty}^{\infty} f(t)e^{-j\omega t}\,dt$$

- ***F=fourier(f)***
- The command performs the Fourier transform of the symbolic expression or function **f** with default variable **x**. By default, the result **F** is a function of **w**.

fourier – examples

Commands	Results
>> syms x w a	
>> fourier(exp(-x^2))	ans =pi^(1/2)/exp(w^2/4)
>> fourier(exp(-3*abs(x)))	ans =6/(w^2 + 9)
>> fourier(cos(x))	ans =pi*(dirac(w - 1) + dirac(w + 1))
>> fourier(sym('1'))	ans =2*pi*dirac(-w)

ifourier

- ***f=ifourier(F)***
- The command performs the inverse Fourier transform of the symbolic expression **F** with default variable **w**. If **F** does not contain **w**, then the default variable is determined by symvar(). By default, the result **f** is a function of **x**.

```
>> ifourier(2*pi*dirac(-w))
ans =1

>> ifourier(pi^(1/2)/exp(w^2/4))

ans =(3991211251234741*exp(-x^2))
  /(2251799813685248*pi^(1/2))=exp(-x^2)
```

Matrices

- ***det(A)***
- The command returns the determinant of a square matrix A.
- The symbolic function is used only if the input of the function is a symbolic matrix!

```
>> syms a b c d;
>> det([a, b; c, d])
ans = a*d - b*c

>> A = [2/3 1/3; 1 1];
>> r = det(A)
r =1/3
```

  - The following commands are used similarly: **inv**, **rank**, …
  - The symbolic functions are used only if the input of the functions is a symbolic expression.

numden

- ***numden(F)***
- The command returns numerator and denominator of a symbolic expression.

```
>> syms a b
>> F=(a-2*3-b^2)/(a*b/2+1)
>> [n,d]=numden(F)

n =- 2*b^2 + 2*a - 12
d =a*b + 2
```

- The following commands are used similarly: **real**, **imag**, **angle**, **abs**, …
- The symbolic functions are used only if the input of the functions is a symbolic expression.

gcd

- ***G=gcd(A,B)***
- The command returns the greatest common divisor (GCD) of corresponding elements of A and B.

```
>> syms x
>> A = x^3 + 13*x^2 + 32*x + 20
>> B = x^4 + 3*x^3 + 2*x^2
>> G = gcd(A,B)

G = (x + 1)*(x + 2)
```

sym2poly

- ***sym2poly(P)***
- The command converts symbolic polynomial **P** to MATLAB coefficient vector

```
>> syms x
>> sym2poly(x^5+3*x^3-2*x-5)

ans =1     0     3     0    -2    -5
```

poly2sym

- ***poly2sym(P)***
- the command converts coefficient of the vector to symbolic polynomial. By default, the symbolic variable **x** is used.

- ***poly2sym(P,v)***
- the command converts vector coefficients to symbolic polynomial in the symbolic variable **v**.

```
>> poly2sym([1    0    3    0    -2    -5])
ans = x^5 + 3*x^3 - 2*x – 5

>> syms v
>> poly2sym([1    0    3    0    -2    -5],v)
 ans = v^5 + 3*v^3 - 2*v - 5
```

latex

- ***latex(S)***
- returns the LaTeX representation of the symbolic expression **S**.

```
>> syms a b c
>> c=a+b*a/5-2^a/b
>> latex(c)
ans =a + \frac{a\, b}{5} - \frac{2^a}{b}
```

Example 1

- Find the analytical solution of the quadratic equation.

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

- Use commands **solve()** and **pretty()**
- Result:

```
+-                       -+
|          2         1/2  |
|    b + (b  - 4 a c)     |
|  - ------------------   |
|            2 a          |
|                         |
|          2         1/2  |
|    b - (b  - 4 a c)     |
|  - ------------------   |
|            2 a          |
+-                       -+
```

- Substitute a = 1; b = -3; c = 4; and find concrete solution.

Example 2

- Express the symbolic variable R_s from the given transfer function $F(j\omega)$.

$$F(j\omega) = \frac{\frac{1}{R_s} - \frac{j * \omega * L_d}{R_s^2}}{1 + \omega^2 * \frac{L_q}{R_s}}$$

- Define symbolic variables: Ld, Lq, Rs, ω, A, T, y_s

- Get the real and imaginary part of the transfer function $F(j\omega)$
- Knowing that: $$y_s = \frac{A*T}{2} * \Re(F(j\omega))$$

- Derive a formula for the calculation R_s

- Result : $$R_s = \frac{A*T}{2*y_s} - L_q * \omega^2$$

Example 3

- Simplify the transfer function $F(s) = \dfrac{\frac{R_S^2 * Lq}{L_d^2 * s}}{\frac{R_S^2 * Lq}{L_d} + \frac{R_S^3 * Lq}{L_d^2 * s}}$
- Define symbolic variables: Ld, Lq,R_s, s
- Reshape transfer function $F(s)$ to the form first-order plant with the transfer function $F(s) = \frac{\mathrm{K}}{T * s + 1}$ (where T is time constant and K is gain).
- From the expression for the time constant (T = ?) express the formula for calculating L_d
- Result : $\mathrm{L_d} = R_s * T$

End

References

- http://www.cak.fs.cvut.cz/SPM/Prednaska34.ppt

- Help MATLAB 2014b, MATLAB 2015b, MATLAB 2016b, MATLAB 2017a, MATLAB 2018b, MATLAB 2020b,