MAT 275 Laboratory 7
Laplace Transform and the Symbolic Math Toolbox

In this laboratory session we will learn how to

1. Use the Symbolic Math Toolbox
2. Calculate the Laplace and Inverse Laplace Transform.

The Symbolic Math Toolbox

The Symbolic Math Toolbox allows MATLAB to perform symbolic calculations. We will start our overview of this Toolbox with two simple and useful tools that come with it.

1. **A quick way to make plots:** `ezplot`. Here is the simplest way to make simple plots.
   - The command `ezplot sin(t)` makes a plot of the sin function on the default interval $(-2\pi, 2\pi)$. More elaborately you can also type `ezplot('sin(t)')`;
   - The command `ezplot('t^2/exp(t)',[-1,5])` plots the function t^2/e^t over the interval $(-1, 5)$.
2. **A fun tool:** `funtool`. Type `funtool` and you will be operating plotting calculator that does symbolic calculations. Use this when you need to do some quick checking about a function, its derivative, integral, inverse, etc. the `'help'` key explains what the other keys do.

Numeric vs Symbolic computations
What is the difference between numeric (plain MATLAB) and symbolic (the Symbolic Math Toolbox) computation? Let us say, you wanted to know the derivative of a function, $t^2 \sin(3t^{1/4})$.

The definition of derivative gives a crude way to *numerically* calculate the derivative of the function as $f' \approx (f(x+h) - f(x))/h)$ if h is small. The MATLAB commands below calculate this approximation of the derivative in the interval $(0, 2\pi)$, at a set of points spaced $\Delta t = h = 0.1$ apart.

```
h = .1;                          % delta t for the difference
t = 0:h:2*pi;                    % the region of interest for t
f = t.^2.*sin(3*t.^(1/4));       % the function at all t values
fprime = diff(f)/h;              % numerical approximation of derivative
                                 % Note: diff(f) has 1 less element than f
plot(t(1:end-1), fprime)         % plot the derivative
```

The derivative of the function is represented by the two lists of numbers `t(1:end-1)` and `fprime`. The derivative at `t(7)` is `fprime(7)`.

Compare this with the Symbolic Tool calculation of the derivative of the same function. Here is the command and response:

```
>> symb_deriv=diff(sym('t^2*sin(3*t^(1/4))'))
symb_deriv =
2*t*sin(3*t^(1/4)) + (3*t^(5/4)*cos(3*t^(1/4)))/4
```

The Symbolic Toolbox gives a formula for the derivative. If you are patient you can verify by hand that indeed

$$\frac{d}{dt}\left(t^2 \sin(3t^{1/4})\right) = 2t\sin(2\sqrt[4]{t}) + (3/4)t^{5/4}\cos(3\sqrt[4]{t}).$$

The command

```
ezplot(symb_deriv,[0,2*pi])
```

gives about the same curve as the previous calculation. If you type

```
int(symb_deriv)
```

you will see a very complicated and long expression. However, if you type

```
simplify(int(symb_deriv))
```

you will get back `t^2*sin(3*t^(1/4))` as expected by the fundamental theorem of calculus.

Notice that `diff` is two different commands. One is the numerical command (dealing with the difference between consecutive terms in a list) and one is the Symbolic Toolbox command (symbolically calculating the derivative). Which one of the commands MATLAB uses depends on whether you have typed something in the correct syntax for one or the other.

Getting help with the Symbolic Toolbox

- If you know the name of the command you want help with, say `diff`, you can see helpful explanations in any of three ways:

  1. type `help diff` at the command line. You will see a description of the numerical MATLAB command `diff` together with this helpful clue at the end:

     ```
     Overloaded methods:
        char/diff
        sym/diff
        fints/diff
        iddata/diff
        umat/diff
     ```

     "`Overloaded`" means the command `diff` has meaning outside of plain MATLAB. If you type `help sym/diff`, you will get help with the Symbolic Toolbox command also called `diff`
  2. Type `help sym/diff` to get help on the *symbolic* command `diff`.
  3. Type `helpdesk` at the MATLAB prompt. Then click on the Symbolic Math Toolbox and select *Functions*. You can then select *Calculus* and click on any one of the commands for more information.

- To see an organized list of the symbolic commands with a very short description you can do either of two things:

  1. type `help symbolic` on the command line. One line in this list is, for example,
     `diff - Differentiate`
  2. Click on the Help button at the top of the command window. Then click on `Function Browser` and select `Symbolic Math Toolbox`.

Using the Symbolic Math Toolbox

Lets see some other things the Symbolic Math Toolbox can do, besides `diff`, `int`, `ezplot`, `simplify` and `funtool`.

- **Basic manipulations:** Expand a polynomial, make it look nice, solve it, and check the solution.

```
syms x a                                % tell matlab that x and a are symbols
f = (x-1)*(x-a)*(x+pi)*(x+2)*(x+3)      % define f
pretty(f)                               % print f in a more readable form
g = expand(f)                           % rewrite f, multiply everything out
h = collect(g)                          % rewrite again by collecting terms
soln = solve(h,x)                       % find all solutions of h = 0
check = subs(f,x,soln(5))           % check, say the fifth solution
```

Comments:

- The **syms** is used to declare symbolic variables.
- Since `x` is a symbol, `f` is automatically treated as a symbolic expression.
- `solve` is a powerful command that tries all kinds of things to find a solution. Here `solve` manages to find all five roots of a fifth order polynomial (something that cannot always be done, by the way).
- `subs` is an often used command if you do math online. Here every occurrence of `x` in the expression of `f` is replaced with the fifth supposed solution. The result of this line of calculation is, predictably, 0. The `subs` command can be used with functions of more than one variable. For instance

  ```
  >> syms x a y
  >> subs(x*y^2,y,a)
  ans =
  a^2*x
  ```

  substitutes $y = a$ into xy^2 to get xa^2 (here x, y and a are symbolic variables). We can also substitute two things at once: `subs(x*y^2,{x,y},{2,3})` evaluates xy^2 for $x = 2$ and $y = 3$.

Laplace Transforms with MATLAB

The Laplace transform of a function f can be obtained using the MATLAB Symbolic Toolbox. The following example shows how to obtain the Laplace transform of $f(t) = \sin(t)$:

```
>> syms t
>> laplace(sin(t))
ans =
1/(s^2 + 1)
>> pretty(ans)

    1
  ------
   2
  s  + 1
```

To make the expression more readable, one can use the command `simplify`. Here is an example for the function $f(t) = -1.25 + 3.5te^{-2t} + 1.25e^{-2t}$

```
>> syms t
>> f=-1.25+3.5*t*exp(-2*t)+1.25*exp(-2*t);
>> F=laplace(f)
F =
5/(4*(s + 2)) + 7/(2*(s + 2)^2) - 5/(4*s)
>> simplify(F)
ans =
(s - 5)/(s*(s + 2)^2)
>> pretty(ans)

    s - 5
  ----------
           2
  s (s + 2)
```

Inverse Laplace Transform

The command for the inverse laplace transform is `ilaplace`. Lets calculate the inverse laplace of the previous function $F(s) = \dfrac{s-5}{s(s+2)^2}$.

```
>> syms s t
>> F=(s-5)/(s*(s+2)^2);
>> ilaplace(F)
ans =
5/(4*exp(2*t)) + (7*t)/(2*exp(2*t)) - 5/4
>> pretty(ans)

      5            7 t        5
  ---------- + ---------- - -
  4 exp(2 t)   2 exp(2 t)   4
```

Partial Fractions in MATLAB

The `residue` function converts a quotient of polynomials to *pole-residue* representation, and back again to polynomials.

```
   [r,p,k]=residue(B,A)
```

finds the residues, poles and direct term of a partial fraction expansion of the ratio of two polynomials $B(s)/A(s)$. The quotient of polynomials $B(s)$ and $A(s)$ is represented as

$$\frac{B(s)}{A(s)} = \frac{r_1}{s-p_1} + \frac{r_2}{s-p_2} + \ldots + \frac{r_n}{s-p_n} + k_s$$

where $\mathbf{r} = (r_1, r_2, \ldots, r_n)^T$ is a column vector of residues, $\mathbf{p} = (p_1, p_2, \ldots, p_n)^T$ is a column vector of pole locations (the roots of the denominator), and $\mathbf{k}$ is a row vector of direct terms (if the degree of the numerator is less than the degree of the denominator, then $\mathbf{k}$ will be an empty vector). The `residue` command requires two input vectors: one holding the coefficients of the numerator and one holding the coefficients of the denominator. The coefficients must be given in descending powers of s.

Consider for instance the following function:

$$Y(s) = \frac{5s-1}{s^3-3s-2}$$

To find the partial fraction decomposition of $Y(s)$ we must first enter the polynomials at the numerator and at the denominator as vectors. The numerator, $5s-1$, can be entered in MATLAB as

```
B = [5,-1];             % coefficients of the numerator in decreasing order
```

while the denominator is represented by the vector

```
A = [1,0,-3,-2];             % coefficients of the denominator in decreasing order
```

Note that the s^2 term is missing, thus we need to enter a zero as the second entry of the vector.
The `residue` command gives the following output:

```
[r, p, k] = residue(B,A)
r =
    1.0000
   -1.0000
    2.0000
p =
```

```
    2.0000
   -1.0000
   -1.0000
k =
     []
```

Thus the partial fraction decomposition of $Y(s)$ is

$$\frac{1}{s-2} + \frac{-1}{s+1} + \frac{2}{(s+1)^2}.$$

The last term in the partial fraction decomposition follows from the fact that the pole -1.0000 is repeated.

Solving ODEs using the Laplace and Inverse Laplace Transform

EXAMPLE
A mass-spring system is modeled by the equation

$$y'' + 4y = g(t), \quad y(0) = y'(0) = 0$$

where $g(t) = \begin{cases} 8t & 0 \le t \le 5 \\ 40 & t > 5 \end{cases}$

(a) Use MATLAB to find the laplace transform of the forcing term $g(t)$ and plot the forcing term. *Note:* The unit step function is denoted in MATLAb by `heaviside`.

Answer: We have $g(t) = 8t + u(t-5)(40-8t)$.
We can plot $g(t)$ and find its Laplace transform using the following MATLAB commands. The graph of $g(t)$ is shown in Fig. L7a.

```
syms s t
g = 8*t + heaviside(t-5)*(40-8*t);
ezplot(g,[0,10])
G= laplace(g)
G =
8/s^2 - 8/(s^2*exp(5*s))
```

Thus the Laplace of $g(t)$ is $G(s) = (1 - e^{-5s})\dfrac{8}{s^2}$.

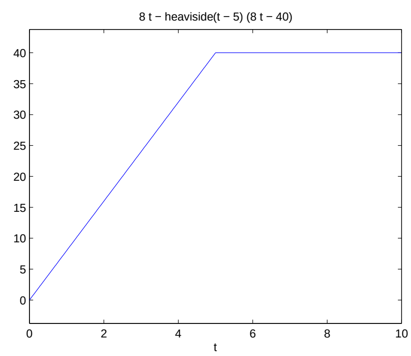

Figure L7a: Graph of $g(t)$

(b) Applying the Laplace transform to both sides of the Differential Equation and substituting the Initial Conditions gives

$$Y(s) = (1 - e^{-5s})\frac{8}{s^2(s^2+4)}$$

where $Y(s)$ is the Laplace transform of the solution $y(t)$. Let

$$F = \frac{8}{s^2(s^2+4)}$$

(i) Use the `residue` command to find the partial fraction decomposition of F and use this partial fraction decomposition to find (by hand) the inverse Laplace transform of $Y(s)$.

Answer: Note that $F = \dfrac{8}{s^4 + 4s^2}$.

```
>> num=[8];    % coefficients of the numerator in decreasing order
>> den=[1,0,4,0,0];  % coefficients of the denominator in decreasing order
>> [r,p,k]=residue(num,den)
r =
        0 + 0.5000i
        0 - 0.5000i
        0
   2.0000
p =
        0 + 2.0000i
        0 - 2.0000i
        0
        0
k =
     []
```

Thus

$$F = \frac{0.5i}{s-2i} - \frac{0.5i}{s+2i} + \frac{2}{s^2}$$

and combining the complex terms we obtain

$$F = -\frac{2}{s^2+4} + \frac{2}{s^2}.$$

Using a Table of Laplace transforms, we find that the inverse Laplace transform of F is

$$f(t) = 2t - \sin(2t)$$

and the inverse laplace of $Y(s)$ is

$$y(t) = f(t) - u(t-5)f(t-5)$$

(ii) Enter the function $F = \frac{8}{s^2(s^2+4)}$ in MATLAB, and then enter `diff(int(F))`. What is the output? Note that MATLAB computes the integral of rational functions by computing the partial fraction decomposition first.

Answer:

```
>> F= 8/(s^2*(s^2+4));
>> diff(int(F))
ans =
2/s^2 - 2/(s^2*(4/s^2 + 1))
>> pretty(ans)
```

```
2          2
-- - -------------
 2    2 / 4      \
s    s  | -- + 1 |
        |  2     |
        \ s      /
```

Simplifying the second term of the answer gives $\frac{2}{s^2} - \frac{2}{4+s^2}$ which is the partial fraction decomposition found in part (i).

(iii) Find the inverse Laplace transform of $Y(s)$ using the `ilaplace` command.

Answer:

```
>> Y = (1-exp(-5*s))*8/(s^2*(s^2+4));
>> y=ilaplace(Y)
y =
2*t - sin(2*t) + 8*heaviside(t - 5)*(sin(2*t - 10)/8 - t/4 + 5/4)
```

We can see that the solution found using `ilaplace` is the same as the one we found in part (i).

(c) Write the solution $y(t)$ you found in part (b) as a piecewise function. This solution is an oscillation around a slanted line for $0 \le t \le 5$ and around a horizontal line for $t > 5$. Determine this slanted and horizontal lines and plot them together with the solution. Compare the graph of the solution to the graph of the forcing term from part (a). Does the graph makes sense in the mass-spring context?

Answer: The solution $y(t)$ can be written as a piecewise function as follows;

$$y(t) = \begin{cases} 2t - \sin(2t) & 0 \le t \le 5 \\ 10 - \sin(2t) + \sin(2(t-5)) & t > 5 \end{cases}$$

Thus $y(t)$ is an oscillation around the line $y = 2t$ for $0 \le t \le 5$ and around the line $y = 10$ for $t > 5$.
We can combine the two lines in the single function $h(t) = 2t + u(t-5)(10-2t)$. We will plot the solution $y(t)$ in the default color for `ezplot` and the function $h(t)$ with a dashed red line. The output is shown in Fig. L7b.

```
p1 = ezplot(y,[0,10]);
hold on
h = 2*t + heaviside(t-5)*(10-2*t);
p2 = ezplot(h,[0,10]);
set(p2, 'Color','red','LineStyle','--')
axis([0,10,0,12])
title('')
legend('solution','lines y= 2t and y = 10',2)
hold off
```

It is much easier to generate overlay plots with different line styles using the `plot` command. The following commands produce the same picture as the commands above.

```
t = 0:0.1:10;
y = eval(vectorize(y));  % write a formula for y that works with arrays and
                        % evaluate it
h = 2.*t + heaviside(t-5).*(10-2*t);   % write a formula for h that works with arrays
plot(t,y,'b-',t,h,'r--')
legend('solution','lines y= 2t and y = 10',2)
```

Note the `vectorize` and `eval` commands. The `vectorize` command takes a formula that is good for scalars and puts dots in the right places. The `eval` command takes text that looks like MATLAB command and executes the command.

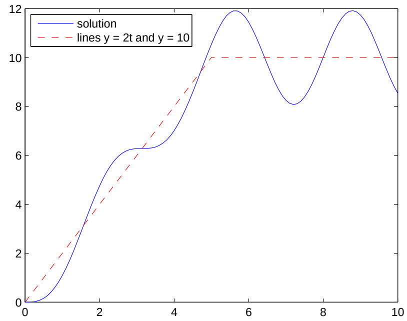

Figure L7b: Graph of the solution to $y'' + 4y = g(t),\ y(0) = y'(0) = 0$

Comparing Figs L7a and L7b from parts (a) and (c), we can see that the graph of the solution makes perfect sense in the mass spring context. The linearly increasing forcing term for the first 5 seconds causes the equilibrium to shift linearly. Once the forcing term stabilizes into a constant value so does the equilibrium and, since the system is undamped, the mass keeps oscillating around the new equilibrium with constant amplitude.

EXERCISES

Instructions: For your lab write-up follow the instructions of LAB 1.

Important! Before you start working on the exercises, enter `clear all` to clear all the variables from memory.

1. Consider the transfer function
$$Y(s) = \frac{4s^2 + 4s + 4}{s^2(s^2 + 3s + 2)}$$
   (a) Use the MATLAB function `residue` to find the residues and poles of $Y(s)$. Use the output to find the partial fraction decomposition of $Y(s)$.
   (b) (i) Use the Table of Laplace transforms and your answer to (a) to find the inverse Laplace transform of $Y(s)$.
   (ii) Enter the function Y in MATLAB and confirm your result from part (i) with the `ilaplace` command (do not forget to declare s as a symbolic variable).

2. Consider the IVP
$$y'' + 2y' + 5y = u(t-2) - u(t-6), \quad y(0) = y'(0) = 0.$$
   (a) Plot the forcing term $u(t-2) - u(t-6)$ in the interval $0 \le t \le 15$. You can use `ezplot` or `plot`.
   (b) Applying the Laplace Transform to both sides of the Differential Equation and substituting the Initial Conditions yields
$$Y(s) = (e^{-2s} - e^{-6s})\frac{1}{s(s^2 + 2s + 5)}$$
   where $Y(s)$ is the Laplace transform of the solution $y(t)$. Find the inverse Laplace of $Y(s)$ using the `ilaplace` command.

(c) Plot the graph of the solution $y(t)$ from part (b) for $0 \le t \le 15$, and compare with the graph of the forcing term from part (a). Does the graph make sense in the mass-spring context? Make sure you comment on both the short term and long term behaviors of the solution

3. A mass $m = 1$ is attached to a spring with constant $k = 5$ and damping constant $c = 4$. The mass is initially in equilibrium position and it is released with an initial velocity of 2 units. At the instant $t = \pi$ the mass is struck with a hammer and at $t = 2\pi$ it is struck again. The motion of the mass satisfies the following IVP:

$$y'' + 4y' + 5y = \delta(t - \pi) + \delta(t - 2\pi); \;\; y(0) = 0, \; y'(0) = 2.$$

The delta (dirac) function is entered in MATLAB with the command `dirac`.

(a) Use MATLAB to find the Laplace transform of the right hand side.

(b) Apply (by hand) the Laplace transform to both sides of the equation and determine $Y(s)$, the Laplace transform of the solution $y(t)$.
Use `ilaplace` to find the inverse Laplace of $Y(s)$.

(c) Plot the solution $y(t)$ in the interval $0 \le t \le 12$, and comment on the graph in the mass-spring context and in relation to the forcing term.

4. Consider a mass spring system with $m = k = 1$ and $x(0) = x'(0) = 0$. At each of the instants $t = 0, \pi, 2\pi, 3\pi, \ldots, n\pi, ..$, the mass is struck a hammer blow with a unit impulse.
Thus the position function of the mass satisfies the IVP

$$x'' + x = \sum_{n=0}^{\infty} \delta(t - n\pi), \quad x(0) = x'(0) = 0.$$

Applying the Laplace Transform we obtain

$$X(s) = \sum_{n=0}^{\infty} \frac{e^{-n\pi s}}{s^2 + 1}$$

so that the solution is

$$x(t) = \sum_{n=0}^{\infty} u(t - n\pi) \sin(t - n\pi).$$

Because $\sin(t - n\pi) = (-1)^n \sin t$ and $u(t - n\pi) = 0$ for $t \le n\pi$, we see that if $n\pi \le t \le (n+1)\pi$, then

$$x(t) = \sin t - \sin t + \sin t - \ldots + (-1)^n \sin t$$

that is

$$x(t) = \begin{cases} \sin t & \text{if } n \text{ is even} \\ 0 & \text{if } n \text{ is odd} \end{cases}$$

The physical explanation is that the first hammer blow (at time $t = 0$) starts the mass moving to the right; just as it returns to the origin, the second hammer blow stops it dead; it remains motionless until the third hammer blow starts it moving again, and so on.

(a) Use MATLAB to construct a figure showing the graph of this position function in the interval $[0, 5\pi]$.

Now consider the same spring and mass, initially at rest, but struck with a hammer at each of the instants $t = 0, 2\pi, 4\pi, \ldots$. Suppose that each hammer blow imparts an impulse of $+1$.
Thus the position function $x(t)$ of the mass satisfies the initial value problem

$$x'' + x = \sum_{n=0}^{\infty} \delta(t - 2n\pi), \quad x(0) = x'(0) = 0 \qquad \text{(L7.1)}$$

(b) Solve the IVP L7.1 to show that if $2n\pi < t < 2(n+1)\pi$, then $x(t) = (n+1)\sin t$.
Thus resonance occurs because the mass is struck each time it passes through the origin moving to the right - in contrast with the previous case, in which the mass was struck each time it returned to the origin.

(c) Use MATLAB to construct a figure showing the graph of this position function in the interval $[0, 6\pi]$.