Laplace Transform Lab: Solving ODEs using Laplace Transform in MATLAB

This lab will teach you to solve ODEs using a built in MATLAB Laplace transform function `laplace`.

There are five (5) exercises in this lab that are to be handed in. Write your solutions in a separate file, including appropriate descriptions in each step.

Include your name and student number in the submitted file.

Contents

Student Information

```
Student Name: Hikaru Kurosawa

Student Number: 10076725240
```

Using symbolic variables to define functions

In this exercise we will use symbolic variables and functions.

```
syms t s x y

f = cos(t)
h = exp(2*x)
```

```
f =

cos(t)


h =

exp(2*x)
```

Laplace transform and its inverse

```
% The routine |laplace| computes the Laplace transform of a function

F=laplace(f)
```

```
F =

s/(s^2 + 1)
```

By default it uses the variable `s` for the Laplace transform But we can specify which variable we want:

```
H=laplace(h)
laplace(h,y)

% Observe that the results are identical: one in the variable |s| and the
% other in the variable |y|
```

```
H =

1/(s - 2)


ans =

1/(y - 2)
```

We can also specify which variable to use to compute the Laplace transform:

```
j = exp(x*t)
laplace(j)
laplace(j,x,s)

% By default, MATLAB assumes that the Laplace transform is to be computed
% using the variable |t|, unless we specify that we should use the variable
% |x|
```

```
j =

exp(t*x)


ans =

1/(s - x)


ans =

1/(s - t)
```

We can also use inline functions with `laplace`. When using inline functions, we always have to specify the variable of the function.

```
l = @(t) t^2+t+1
laplace(l(t))
```

```
l =

  function_handle with value:

    @(t)t^2+t+1


ans =

(s + 1)/s^2 + 2/s^3
```

MATLAB also has the routine `ilaplace` to compute the inverse Laplace transform

```
ilaplace(F)
ilaplace(H)
ilaplace(laplace(f))
```

```
ans =

cos(t)


ans =

exp(2*t)


ans =

cos(t)
```

If `laplace` cannot compute the Laplace transform, it returns an unevaluated call.

```
g = 1/sqrt(t^2+1)
G = laplace(g)
```

```
g =

1/(t^2 + 1)^(1/2)
```

```
G =

laplace(1/(t^2 + 1)^(1/2), t, s)
```

But MATLAB "knows" that it is supposed to be a Laplace transform of a function. So if we compute the inverse Laplace transform, we obtain the original function

```
ilaplace(G)
```

```
ans =

1/(t^2 + 1)^(1/2)
```

The Laplace transform of a function is related to the Laplace transform of its derivative:

```
syms g(t)
laplace(diff(g,t),t,s)
```

```
ans =

s*laplace(g(t), t, s) - g(0)
```

Exercise 1

Objective: Compute the Laplace transform and use it to show that MATLAB 'knows' some of its properties.

Details:

(a) Define the function `f(t)=exp(2t)*t^3`, and compute its Laplace transform `F(s)`. (b) Find a function `f(t)` such that its Laplace transform is `(s - 1)*(s - 2))/(s*(s + 2)*(s - 3)` (c) Show that MATLAB 'knows' that if `F(s)` is the Laplace transform of `f(t)`, then the Laplace transform of `exp(at)f(t)` is `F(s-a)`

(in your answer, explain part (c) using comments).

Observe that MATLAB splits the rational function automatically when solving the inverse Laplace transform.

```
%a
f = @(t) exp(2*t)*t^3
F = laplace(f(t))
%6/(s - 2)^4

%b
F = ((s - 1)*(s - 2))/(s*(s + 2)*(s - 3))
ilaplace(F)
%f = (6*exp((-2*t)))/5 + (2*exp((3*t)))/15 - sym(1/3)

syms f(t) t s a

%c
F = laplace(f(t))
F_sa = laplace(exp(a*t)*f(t))
%taking the laplace of exp(a*t)*f(t) is producing a shift in the s domain,
%which implies that MATLAB knows of this property that an exponential in
%the t domain corresesopnds to a shift in the s domain
```

```
f =

  function_handle with value:

    @(t)exp(2*t)*t^3


F =

6/(s - 2)^4


F =

((s - 1)*(s - 2))/(s*(s + 2)*(s - 3))


ans =

(6*exp(-2*t))/5 + (2*exp(3*t))/15 - 1/3
```

```
F =

laplace(f(t), t, s)


F_sa =

laplace(f(t), t, s - a)
```

Heaviside and Dirac functions

These two functions are builtin to MATLAB: `heaviside` is the Heaviside function `u_0(t)` at 0

To define `u_2(t)`, we need to write

```
f=heaviside(t-2)
ezplot(f,[-1,5])

% The Dirac delta function (at |0|) is also defined with the routine |dirac|

g = dirac(t-3)

% MATLAB "knows" how to compute the Laplace transform of these functions

laplace(f)
laplace(g)
```

```
f =

heaviside(t - 2)


g =

dirac(t - 3)


ans =

exp(-2*s)/s


ans =

exp(-3*s)
```

Exercise 2

Objective: Find a formula comparing the Laplace transform of a translation of `f(t)` by `t-a` with the Laplace transform of `f(t)`

Details:

- Give a value to `a`
- Let `G(s)` be the Laplace transform of `g(t)=u_a(t)f(t-a)` and `F(s)` is the Laplace transform of `f(t)`, then find a formula relating `G(s)` and `F(s)`

In your answer, explain the 'proof' using comments.

```
f = @(t)t^3
g = @(t) heaviside(t-2)*f(t-2) %let a = 2
G = laplace(g(t))
F = laplace(f(t))

%relationship is as follows: G(s) = exp(-a*s)*F(s)

%let G(s) = laplace(u_a(t)f(t-a))
%then, applying the definition of laplace transform, G(s) = integral(exp(-s*t)*u_a(t)f(t-a))*dt, evaluated at 0 to infinity
%u_a(t) is just a unit function that has a value of a to inifinty, G(s) = integral(exp(-s*t)*f(t-a))*dt, evaluated at a to infinity
%defining a new variable, u = t-a, G(s) = integral(exp(-s*(u+a))*f(u))*du 0 to infinity
%this expression can be expanded as  to G(s) = exp(-s*(a))integral(exp(-s*u))*f(u))*du
%the right hand side of the equation is the definiton of laplace transform
%of f(t), so right hand side equates to F(s)

%therefore, G(s) = exp(-s*(a))*F(s)
```

```
f =

  function_handle with value:

    @(t)t^3
```

```
g =

  function_handle with value:

    @(t)heaviside(t-2)*f(t-2)


G =

(6*exp(-2*s))/s^4


F =

6/s^4
```

Solving IVPs using Laplace transforms

Consider the following IVP, `y''-3y = 5t` with the initial conditions `y(0)=1` and `y'(0)=2`. We can use MATLAB to solve this problem using Laplace transforms:

```
% First we define the unknown function and its variable and the Laplace
% tranform of the unknown

syms y(t) t Y s

% Then we define the ODE

ODE=diff(y(t),t,2)-3*y(t)-5*t == 0

% Now we compute the Laplace transform of the ODE.

L_ODE = laplace(ODE)

% Use the initial conditions

L_ODE=subs(L_ODE,y(0),1)
L_ODE=subs(L_ODE,subs(diff(y(t), t), t, 0),2)

% We then need to factor out the Laplace transform of |y(t)|

L_ODE = subs(L_ODE,laplace(y(t), t, s), Y)
Y=solve(L_ODE,Y)

% We now need to use the inverse Laplace transform to obtain the solution
% to the original IVP

y = ilaplace(Y)

% We can plot the solution

ezplot(y,[0,20])

% We can check that this is indeed the solution

diff(y,t,2)-3*y
```

```
ODE =

diff(y(t), t, t) - 3*y(t) - 5*t == 0


L_ODE =

s^2*laplace(y(t), t, s) - s*y(0) - subs(diff(y(t), t), t, 0) - 5/s^2 - 3*laplace(y(t), t, s) == 0


L_ODE =

s^2*laplace(y(t), t, s) - s - subs(diff(y(t), t), t, 0) - 5/s^2 - 3*laplace(y(t), t, s) == 0


L_ODE =

s^2*laplace(y(t), t, s) - s - 5/s^2 - 3*laplace(y(t), t, s) - 2 == 0


L_ODE =

Y*s^2 - s - 3*Y - 5/s^2 - 2 == 0


Y =

(s + 5/s^2 + 2)/(s^2 - 3)
```

```
y =

cosh(3^(1/2)*t) - (5*t)/3 + (11*3^(1/2)*sinh(3^(1/2)*t))/9


ans =

5*t
```

Exercise 3

Objective: Solve an IVP using the Laplace transform

Details: Explain your steps using comments

- Solve the IVP
- `y'''+2y''+y'+2*y=-cos(t)`
- `y(0)=0`, `y'(0)=0`, and `y''(0)=0`
- for `t` in `[0,10*pi]`
- Is there an initial condition for which `y` remains bounded as `t` goes to infinity? If so, find it.

```
syms y(t) t Y s

%define function for homogeneous
ODE=diff(y(t),t,3)+2*diff(y(t),t,2)+diff(y(t),t,1)+ 2*y(t)-cos(t) == 0
% compute laplace transform


L_ODE = laplace(ODE)

% Use the initial conditions

L_ODE=subs(L_ODE,y(0),0)
L_ODE=subs(L_ODE,subs(diff(y(t), t), t, 0),0)
L_ODE=subs(L_ODE,subs(diff(y(t), t,2), t, 0),0)


L_ODE = subs(L_ODE,laplace(y(t), t, s), Y)
Y=solve(L_ODE,Y)

%use inverse laplace transform

y = ilaplace(Y)

(2*cos(t))/25 - (2*exp((-2*t)))/25 - (3*sin(t))/50

% plot the solution

ezplot(y,[0,20])

% solution

%initial condition to keep the y bounded
% (2*cos(t))/25 - (2*exp((-2*t)))/25 - (3*sin(t))/50 - (t*cos(t))/10 + (t*sin(t))/5

%there is no value of the initial conditioins such that the y remains
%bounded.
%in order for the y to be bounded, -tcos(t)/10 and tsin(t)/5 terms of the
%solution must be canceled out. This can be done by adding the homogenous
%solution of the function to the non-homogenous solution and find the initial conditions that gets rid of the tsin(t) and tcos(t) terms.

%However, when the homogenous system is solved, the general solution of the
%homogenous equation turns out to be y=c1*exp{-2t}+ c2*cos(t)+ c3*sin(t) in
%which c1, c2, c3 are determinined by the initial conditions. As seen, this
%general solution doesn't have the any values multiplied by t, so there is
%no value to cancel out the t*sin(t) and t*cos(t) terms. Therefore, there
%are no initial values that can keep the y bounded.
```

```
ODE =

2*y(t) - cos(t) + diff(y(t), t) + 2*diff(y(t), t, t) + diff(y(t), t, t, t) == 0


L_ODE =

s*laplace(y(t), t, s) - y(0) - 2*s*y(0) - s*subs(diff(y(t), t), t, 0) - s/(s^2 + 1) + 2*s^2*laplace(y(t), t, s) + s^3*laplace(y(t), t, s) - 2*subs(diff(y(t)


L_ODE =
```

```
s*laplace(y(t), t, s) - s*subs(diff(y(t), t), t, 0) - s/(s^2 + 1) + 2*s^2*laplace(y(t), t, s) + s^3*laplace(y(t), t, s) - 2*subs(diff(y(t), t), t, 0) - subs

L_ODE =

s*laplace(y(t), t, s) - s/(s^2 + 1) + 2*s^2*laplace(y(t), t, s) + s^3*laplace(y(t), t, s) - subs(diff(y(t), t, t), t, 0) + 2*laplace(y(t), t, s) == 0


L_ODE =

s*laplace(y(t), t, s) - s/(s^2 + 1) + 2*s^2*laplace(y(t), t, s) + s^3*laplace(y(t), t, s) + 2*laplace(y(t), t, s) == 0


L_ODE =

2*Y + Y*s - s/(s^2 + 1) + 2*Y*s^2 + Y*s^3 == 0


Y =

s/((s^2 + 1)*(s^3 + 2*s^2 + s + 2))


y =

(2*cos(t))/25 - (2*exp(-2*t))/25 - (3*sin(t))/50 - (t*cos(t))/10 + (t*sin(t))/5


ans =

(2*cos(t))/25 - (2*exp(-2*t))/25 - (3*sin(t))/50
```

Exercise 4

Objective: Solve an IVP using the Laplace transform

Details:

- Define
- `g(t) = 3 if 0 < t < 2`
- `g(t) = t+1 if 2 < t < 5`
- `g(t) = 5 if t > 5`
- Solve the IVP
- `y''+2y'+5y=g(t)`
- `y(0)=2 and y'(0)=1`
- Plot the solution for `t` in `[0,12]` and `y` in `[0,2.25]`.

In your answer, explain your steps using comments.

```
syms y(t) t Y s

%define the ODE
g = @(t) 3+ heaviside(t-2)*(t-2) - heaviside(t-5)*(t-4)
ezplot(g(t),[0,12])

ODE=diff(y(t),t,2)+2*diff(y(t),t,1)+ 5*y(t)-g(t) == 0

% Now we compute the Laplace transform of the ODE.

L_ODE = laplace(ODE)

% Use the initial conditions y'(0) = 1, y(0) = 2

L_ODE=subs(L_ODE,y(0),2)
L_ODE=subs(L_ODE,subs(diff(y(t), t), t, 0),1)

% We then need to factor out the Laplace transform of |y(t)|

L_ODE = subs(L_ODE,laplace(y(t), t, s), Y)
Y=solve(L_ODE,Y)

% We now need to use the inverse Laplace transform to obtain the solution
% to the original IVP

y = ilaplace(Y)
%plot the solution
ezplot(y,[0,12])
ylim([0 2.25])
```

```
g =

  function_handle with value:
```

```
    @(t)3+heaviside(t-2)*(t-2)-heaviside(t-5)*(t-4)


ODE =

5*y(t) - heaviside(t - 2)*(t - 2) + heaviside(t - 5)*(t - 4) + 2*diff(y(t), t) + diff(y(t), t, t) - 3 == 0


L_ODE =

2*s*laplace(y(t), t, s) - 2*y(0) - s*y(0) - exp(-2*s)/s^2 + s^2*laplace(y(t), t, s) - subs(diff(y(t), t), t, 0) - 3/s + (exp(-5*s)*(s + 1))/s^2 + 5*laplace(


L_ODE =

2*s*laplace(y(t), t, s) - 2*s - exp(-2*s)/s^2 + s^2*laplace(y(t), t, s) - subs(diff(y(t), t), t, 0) - 3/s + (exp(-5*s)*(s + 1))/s^2 + 5*laplace(y(t), t, s)


L_ODE =

2*s*laplace(y(t), t, s) - 2*s - exp(-2*s)/s^2 + s^2*laplace(y(t), t, s) - 3/s + (exp(-5*s)*(s + 1))/s^2 + 5*laplace(y(t), t, s) - 5 == 0


L_ODE =

5*Y - 2*s + 2*Y*s - exp(-2*s)/s^2 + Y*s^2 - 3/s + (exp(-5*s)*(s + 1))/s^2 - 5 == 0


Y =

(2*s + exp(-2*s)/s^2 + 3/s - (exp(-5*s)*(s + 1))/s^2 + 5)/(s^2 + 2*s + 5)


y =

heaviside(t - 2)*(t/5 + (2*exp(2 - t)*(cos(2*t - 4) - (3*sin(2*t - 4))/4))/25 - 12/25) - heaviside(t - 5)*(t/5 + (2*exp(5 - t)*(cos(2*t - 10) - (3*sin(2*t -
```

Exercise 5

Objective: Use the Laplace transform to solve an integral equation

Verify that MATLAB knowns about the convolution theorem by explaining why the following transform is computed correctly.

```
syms t tau y(tau) s
I=int(exp(-2*(t-tau))*y(tau),tau,0,t)
laplace(I,t,s)

%the convolution thereorem states that the fouriuer transform of the
%convolutoin equals the product of the fourier transform of the two
%functions
%in equation form, laplace((f*g)(t)) =  laplace(f) * laplace(g)
%here, I is the convolution of f and g, where f = exp(-2*t) and g = y(t)
%furtheremore, the laplace(f) = 1/(s+2) and laplace(g) = laplace(y(t), t, s)
%therefore, the fact that laplace(I,t,s) is computed as
% laplace(y(t),t,s)/(s+2) which is the product of the laplace transforms of f and g
% shows that MATLAB knows about the convolution theorem.
```

```
I =

int(exp(2*tau - 2*t)*y(tau), tau, 0, t)


ans =

laplace(y(t), t, s)/(s + 2)
```