# Laplace Transforms in Matlab

**Kurt Bryan and SIMIODE**

**The Laplace Transform:** To Laplace transform a function, e.g., f(t) = t^2, execute

```
syms s;
syms t;
laplace(t^2,t,s)
```

More generally, to compute the Laplace transform of f(t) execute "laplace(f(t),t,s)". The second argument "t" indicates the independent variable in f(t) and the third argument "s" indicates that the transform should have independent variable "s".

**The Inverse Laplace Transform:** The command is "ilaplace", e.g.,

```
ilaplace(2/s^3,s,t)
```

**Example: Solving a first-order ODE:** Consider the ODE u'(t) = -2*u(t) + t with initial data u(0) = 3.

Step 1: To solve using Laplace transforms (explicitly carrying out all the steps), first define the ODE

```
syms u(t);
ode = diff(u(t),t) == -2*u(t)+t
```

Step 2: Laplace transform both sides of the ODE, which can be done as

```
lapode = laplace(ode,t,s)
```

Matlab transformed both sides of the ODE, and knows the rule for transforming derivatives. Matlab uses the notation "laplace(u(t),t,s)" for the Laplace transform of u(t). Let use replace this with the notation U(s) (not necessary, just more aesthetically pleasing) and also substitute in the initial data u(0) = 3.

```
syms U;
lapode2 = subs(lapode,[u(0),laplace(u(t),t,s)],[3,U])
```

Step 3: We can now solve for U(s), e.g.,

```
Usol = solve(lapode2,U)
```

Step 4: The last step is to inverse Laplace transform "Usol" to find u(t), which we do as

```
usol(t) = ilaplace(Usol,s,t)
```

This is the solution to the ODE. As a quick check,

```
dsolve(ode,[u(0)==3])
```

**Example: A second-order ODE**. Consider the second-order ODE

```
ode = diff(u(t),t,2) + 4*diff(u(t),t) + 3*u(t) == sin(t)
```

with u(0) = 1 and u'(0) = 2. To solve follows the steps below.

Step 1: Laplace transform both sided of the ODE

```
lapode = laplace(ode,t,s)
```

Step 2: Substitute in the initial data and (optionally) use U(s) to replace "laplace(u(t),t,s)": Note that u'(0) is denoted by "subs(diff(u(t),t),t,0)" by Matlab.

```
Du = diff(u);
lapode2 = subs(lapode,[u(0),Du(0),laplace(u(t),t,s)],[1,2,U])
```

Step 3: Solve for U(s) as

```
Usol = solve(lapode2,U)
```

Step 4: Inverse transform to find u(t)

```
usol(t) = ilaplace(Usol,s,t)
```

**The Heaviside and Dirac Delta Functions:** The Heaviside function in Matlab is heavisdie(t),

```
heaviside(t)
```

For example

```
laplace(heaviside(t),t,s)
```

A plot:

```
fplot(heaviside(t),[-2,2])
```

The Dirac delta function is given by

```
dirac(t)
```

Example: The mass in a damped spring-mass system with mass 1 kg, damping constant 2 newtons per meter per second, and spring constant 5 newtons per meter is subjected to a constant force of 2 newtons for 0 < t < 4 seconds, at which point the force drops to zero. At time t = 6 seconds the mass is subjected to a hammer blow with total impulse 8 newton-meters. Find the motion of the mass if the mass starts at rest and at equilibrium, and plot this motion for 0 < t < 20 seconds.

The relevant ODE ii

```
syms u(t);
Du = diff(u)
ode = diff(Du) + 2*Du + 5*u(t) == 2 - 2*heaviside(t-4)  + 8*dirac(t-6)
```

We can solve exactly as in the second order example above. Transform both sides and substitute in the initial data:

```
syms s;
lapode = laplace(ode,t,s);
Du = diff(u);
lapode2 = subs(lapode,[u(0),Du(0),laplace(u(t),t,s)],[0,0,U])
```

Solve for U and inverse transform

```
Usol = solve(lapode2,U)
usol = simplify(ilaplace(Usol,s,t))
```

A plot:

```
fplot(usol, [0 20])
```

Alternatively, we the dsolve command can handle ODEs with Heaviside and Dirac functions directly, though the solution may appear superficially different.

```
usol2 = simplify(dsolve(ode,[u(0)==0,Du(0)==0]))
simplify(usol-usol2)
fplot(usol2, [0 20])
```