Laplace Transforms in Mathematica

Craig Beasley
Department of Electrical and Systems Engineering
Washington University in St. Louis
St. Louis, MO

February 8, 2012

Mathematica can be used to take a complicated problem like a Laplace transform and reduce it to a series of commands. The output from each command is used as the input for the next. The advantage to this approach is that you focus on what operations you perform to solve your problem rather than how you perform each operation. After explaining each command and what it does, they will all be put together at the end of the document to illustrate how an entire problem can be worked out using Mathematica.

Laplace Transforms

Laplace transforms are fairly simple and straightforward. The syntax is as follows:

LaplaceTransform [expression , original variable , transformed variable]

```
In[7]:= LaplaceTransform[t^2 + Cos[t], t, s]
Out[7]= 2/s^3 + s/(1 + s^2)
```

Inverse Laplace Transforms

Inverse Laplace transforms work very much the same as the forward transform. The only difference is that the order of variables is reversed.

InverseLaplaceTransform [expression , transformed variable , original variable]

```
In[8]:= InverseLaplaceTransform[2/s^3 + s/(1 + s^2), s, t]
Out[8]= t^2 + Cos[t]
```

Partial Fractions

The Apart command decomposes an expression into its partial fractions.

```
In[16]:= Apart[(3 s^2 - 3 s - 18)/((s + 1) (s - 1) (s - 5) (s + 4))]
Out[16]= 7/(36 (-5 + s)) + 9/(20 (-1 + s)) - 1/(3 (1 + s)) - 14/(45 (4 + s))
```

Unit Step Function

There are two ways to generate a unit step function. You may use either the UnitStep command or the HeavisideTheta command.

Both commands are mathematical functions, like Sin or Cos, and therefore you can multiply it with another expression in order to take the Laplace transform of a t-shifted function.

```
In[4]:= LaplaceTransform[Cos[x - Pi / 2] * HeavisideTheta[x - Pi / 2], x, s]

Out[4]= e^(-(π s)/2) s / (1 + s^2)
```

```
In[19]:= LaplaceTransform[Cos[x - Pi / 2] * UnitStep[x - Pi / 2], x, s]

Out[19]= e^(-(π s)/2) s / (1 + s^2)
```

Impulse Function

The impulse function is structured very much the same as the unit step function. All that changes is the name of the command. You use the DiracDelta command with the same syntax.

```
In[23]:= LaplaceTransform[125 * DiracDelta[t - Pi / 3], t, s]

Out[23]= 125 e^(-(π s)/3)
```

Convolution

Convolution uses the Convolve command, which is somewhat tricky and requires a bit of explanation. The syntax is:

Convolve [first function , second function , dummy variable , final variable]

Convolution is defined in Mathematica as an integral from -∞ to +∞, which is consistent with its use in signal processing. Your book defines convolution as an integral from 0 to t. The Heaviside function will be required in order to input functions into the Convolve command.

```
In[10]:= Convolve[Sin[τ] * UnitStep[τ], Cos[τ] * UnitStep[τ], τ, t]

Out[10]= 1/2 t Sin[t] UnitStep[t]
```

Alternatively, Mathematica can be used to evaluate the integral directly.

```
In[13]:= ∫_0^t Sin[τ] * Cos[t - τ] dτ

Out[13]= 1/2 t Sin[t]
```

The only difference between the two is the presence of the Heaviside function multiplied onto the result. However, that is fully consistent with the limits on the convolution integral.

Initial Value Problems

To put everything together, I define a shortcut for the differential equation I wish to solve. I picked one that uses an impulse function.

```
In[16]:= de = y''[t] + 3 y'[t] + 2 y[t] == 10 (Sin[t] + DiracDelta[t - 1])
Out[16]= 2 y[t] + 3 y'[t] + y''[t] == 10 (DiracDelta[-1 + t] + Sin[t])
```

Next, take a Laplace transform of that equation.

```
In[17]:= LaplaceTransform[de, t, s]
Out[17]= 2 LaplaceTransform[y[t], t, s] + s^2 LaplaceTransform[y[t], t, s] +
          3 (s LaplaceTransform[y[t], t, s] - y[0]) -
          s y[0] - y'[0] == 10 (E^-s + 1/(1 + s^2))
```

All instances of Y(s) that you would have on your paper while working the problem out by hand have been replaced by LaplaceTransform[y[t],t,s]. The terms are also arranged in a different order. Mathematica output starts with constants and ends with the highest order derivative or the highest power of a variable.

Two new commands will be introduced here. One is the Out command, which recalls a particular output line. The other is a ReplaceAll command, which replaces all instances of a particular term with another. The Out command can also be called with the % key, and ReplaceAll can also be called with /. – there is no space between the forward slash and the period. If you use the Out command, you must specify which output line you wish to recall. Using % as a shortcut always recalls the most recent output, or you can use %n to recall the nth output. This will plug in the initial values.

```
In[19]:= Out[17] /. {y[0] → 1, y'[0] → -1}
Out[19]= 1 - s + 2 LaplaceTransform[y[t], t, s] + s^2 LaplaceTransform[y[t], t, s] +
          3 (-1 + s LaplaceTransform[y[t], t, s]) == 10 (E^-s + 1/(1 + s^2))
```

The Solve command is used to speed through algebra. Instead of solving for Y, you solve for LaplaceTransform[y[t],t,s].

```
In[20]:= Solve[
          1 - s + 2 LaplaceTransform[y[t], t, s] + s^2 LaplaceTransform[y[t], t, s] +
            3 (-1 + s LaplaceTransform[y[t], t, s]) == 10 (E^-s + 1/(1 + s^2)),
          LaplaceTransform[y[t], t, s]]
Out[20]= {{LaplaceTransform[y[t], t, s] → (2 + s + 10 (E^-s + 1/(1+s^2)))/(2 + 3 s + s^2)}}
```

Transform the result back into the time domain.

```
In[21]:= InverseLaplaceTransform[%20, s, t]

Out[21]= {{y[t] → e^(-2 t) (-2 + 6 e^t - 3 e^(2 t) Cos[t] +
            10 e (-e + e^t) HeavisideTheta[-1 + t] + e^(2 t) Sin[t])}}
```

Remember: The HeavisideTheta function is the same as the UnitStep function! This was mentioned on page #2.