Laplace Transforms in Mathematica

Kurt Bryan and SIMIODE

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

In[1]:=
```
LaplaceTransform[t^2, t, s]
```

More generally, to compute the Laplace transform of f(t) execute "LaplaceTransform[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 "InverseLaplaceTransform", e.g.,

In[2]:=
```
InverseLaplaceTransform[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

In[3]:=
```
ode = u'[t] == -2*u[t]+t
```

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

In[4]:=
```
lapode = LaplaceTransform[ode, t, s]
```

Mathematica transformed both sides of the ODE, and knows the rule for transforming derivatives. Mathematica uses the notation "LaplaceTransform[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.

In[5]:=
```
lapode2 = lapode /. {u[0] → 3, LaplaceTransform[u[t], t, s] → U[s]}
```

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

In[6]:=
```
sol = Solve[lapode2, U[s]];
Usol = U[s] /. sol[[1]]
```

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

In[8]:=
```
usol = InverseLaplaceTransform[Usol, s, t]
```

This is the solution to the ODE. A quick check using DSolve:

In[9]:=
```
DSolve[{ode, u[0] == 3}, u[t], t]
```

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

In[10]:=
```
ode = u''[t] + 4*u'[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

In[11]:=
```
lapode = LaplaceTransform[ode, t, s]
```

Step 2: Substitute in the initial data and (optionally) use U(s) to replace "LaplaceTransform[u[t],t,s]":

In[12]:=
```
lapode2 = lapode /. {u[0] → 1, u'[0] → 2, LaplaceTransform[u[t], t, s] → U[s]}
```

Step 3: Solve for U[s] as

In[13]:=
```
sol = Solve[lapode2, U[s]]
Usol = U[s] /. sol⟦1⟧
```

Step 4: Inverse transform to find u(t)

In[15]:=
```
usol = InverseLaplaceTransform[Usol, s, t]
```

The Heaviside and Dirac Delta Functions: The Heaviside function in Mathematica is denoted by "UnitStep". For example

In[16]:=
```
LaplaceTransform[UnitStep[t], t, s]
```

A plot of the Heaviside function:

In[17]:=
```
Plot[UnitStep[t], {t, -2, 2}]
```

The "UnitStep" function takes the definition that UnitStep[0] = 1. Mathematica has a variation that leaves the Heaviside function undefined at t = 0, namely "HeavisideTheta[t]".

The Dirac delta function is denoted by "DiracDelta[t]". For example

In[18]:=
```
LaplaceTransform[DiracDelta[t], t, s]
```

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 is

In[19]:=
```
ode = u''[t] + 2*u'[t] + 5*u[t] == 2 - 2*UnitStep[t - 4] + 8*DiracDelta[t - 6]
```

We can solve exactly as in the second order example above, or simple execute

In[20]:=
```
sol = DSolve[{ode, u[0] == 0, u'[0] == 0}, u[t], t];
usol = Simplify[u[t] /. {sol⟦1⟧}]
```

A plot:

In[22]:=
```
Plot[usol, {t, 0, 20}, PlotRange → {-1, 2.5}]
```