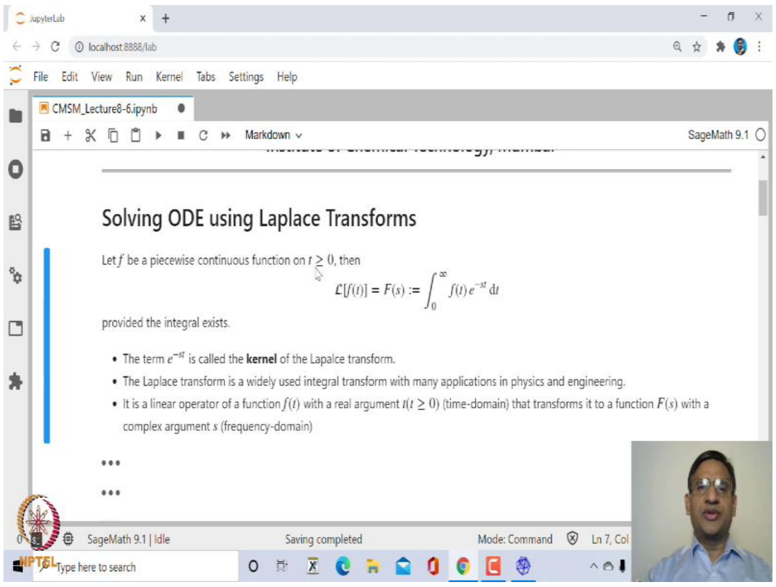
Computational Mathematics with SageMath

Prof. Ajit Kumar
Department of Mathematics
Institute of Chemical Technology, Mumbai

Lecture - 54
Solving ODE using Laplace Transforms in SageMath

Welcome to the 54th lecture on Computational Mathematics with SageMath. In this lecture, we shall look at solving differential equations using Laplace Transforms.

(Refer Slide Time: 00:34)



Solving ODE using Laplace Transforms

Let f be a piecewise continuous function on $t \geq 0$, then

$$\mathcal{L}[f(t)] = F(s) := \int_0^\infty f(t)\, e^{-st}\, dt$$

provided the integral exists.

- The term e^{-st} is called the **kernel** of the Lapalce transform.
- The Laplace transform is a widely used integral transform with many applications in physics and engineering.
- It is a linear operator of a function $f(t)$ with a real argument $t (t \geq 0)$ (time-domain) that transforms it to a function $F(s)$ with a complex argument s (frequency-domain)

Let us first look at what is Laplace transform of a function. Suppose you have a function f which is piecewise continuous defined on t greater than equal to 0 that is t varying between 0 to infinity. Then, we define Laplace transform of f(t), which we denote also by F(s), for a parameter s and it is defined as improper integral of f t into e to the power minus s times t dt integral going from 0 to infinity.

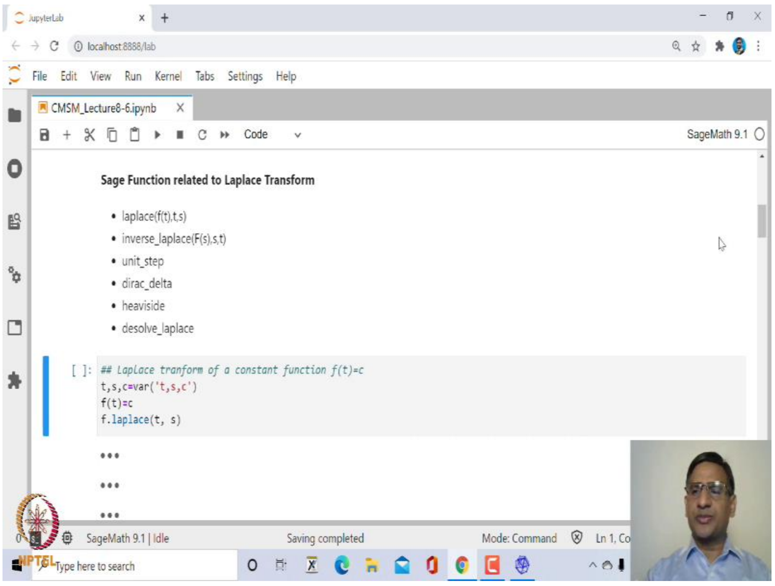
Of course, provided this integral exists. Now, there are conditions on f under which this integral exists and function should have what is called exponential order alpha.

So, in case f is piecewise continuous and is of exponential order alpha, then this Laplace transform will exist for all s bigger than alpha.

If you look at this term, e to the power minus st, this is what is called kernel of this Laplace transform. Laplace transform is one of the integral transforms and integral transforms are defined in terms of its kernel function.

This Laplace transform has wide range of applications in physics and engineering. If you look at what we are doing in this Laplace transform, we are transforming function of t, here we call t as time domain into a function capital F of s, s which is also known as frequency domain. So, you can see here this is actually just a change of coordinates. We are changing from coordinate t to coordinate s.

(Refer Slide Time: 02:47)



Sage Function related to Laplace Transform

- laplace(f(t),t,s)
- inverse_laplace(F(s),s,t)
- unit_step
- dirac_delta
- heaviside
- desolve_laplace

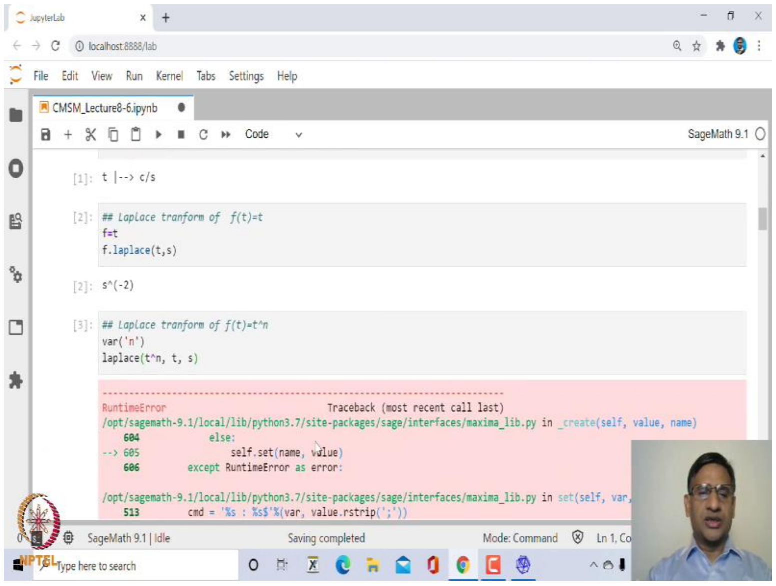
```
[ ]: ## Laplace tranform of a constant function f(t)=c
t,s,c=var('t,s,c')
f(t)=c
f.laplace(t, s)
```

Now, let us look at what are the functions and methods that we are going to be used in order to deal with this particular topic in SageMath. We will be using laplace which is an inbuilt function to find Laplace transform of a function f(t), we need to mention t comma s and we need to define this s as a variable. Then you can also find inverse Laplace transform of capital F(s).

In case Laplace transform of small f(t) is capital F(s), then the inverse Laplace transform of capital F(s) will be f(t) and that can be obtained using inverse Laplace transform. Apart from that if you have learnt Laplace transforms, then, you make use of what are called unit step functions, dirac delta functions, heaviside functions etc. These are also inbuilt in SageMath.

Sage also provides an inbuilt function to solve ordinary differential equation of initial value using a function called desolve_laplace. So, these are the functions from SageMath that we will be exploring in this particular lecture.

Now, let us look at, suppose we have to find Laplace transform of a constant function, let us say f (t) is equal to c, where f is defined on 0 to infinity. First let us declare t, s and c as variables and then f(t) is equal to c. Then all we need to mention is f.laplace(t,s), you can simply say here with respect to t and s. So, t is the variable and s is the parameter, the frequency domain. So, let us execute this, when we execute this, it should give me answer, which is Laplace transform of a constant function, and it should be c upon s. (Refer Slide Time: 04:58)

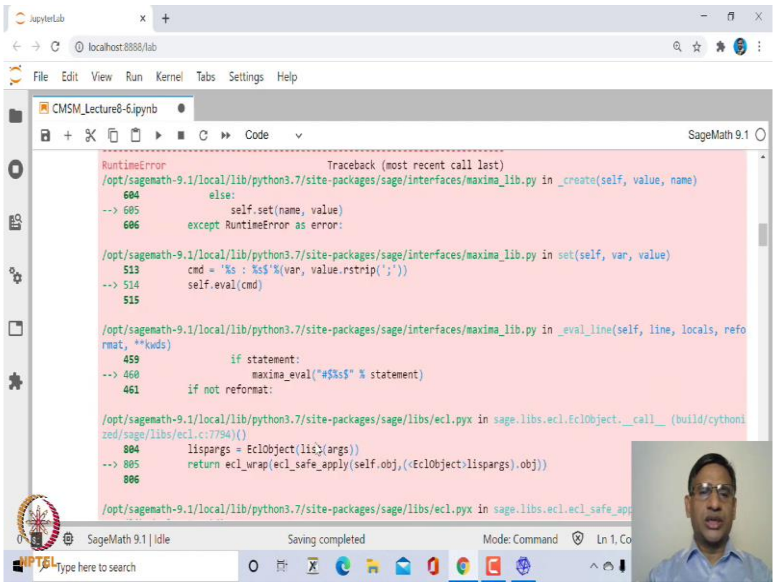

```
[1]: t |--> c/s

[2]: ## Laplace tranform of  f(t)=t
     f=t
     f.laplace(t,s)

[2]: s^(-2)

[3]: ## Laplace tranform of f(t)=t^n
     var('n')
     laplace(t^n, t, s)

---------------------------------------------------------------------------
RuntimeError                              Traceback (most recent call last)
/opt/sagemath-9.1/local/lib/python3.7/site-packages/sage/interfaces/maxima_lib.py in _create(self, value, name)
    604             else:
--> 605                 self.set(name, value)
    606         except RuntimeError as error:

/opt/sagemath-9.1/local/lib/python3.7/site-packages/sage/interfaces/maxima_lib.py in set(self, var,
    513         cmd = '%s : %s$'%(var, value.rstrip(';'))
```

You can see here this s is not this is not defined for 0 and this is defined only for s positive.

Now, suppose we want to find Laplace transform of this identity function f(t)=t, then we can simply say f. laplace(t,s), which is s to the power minus 2, that is 1 upon s square.

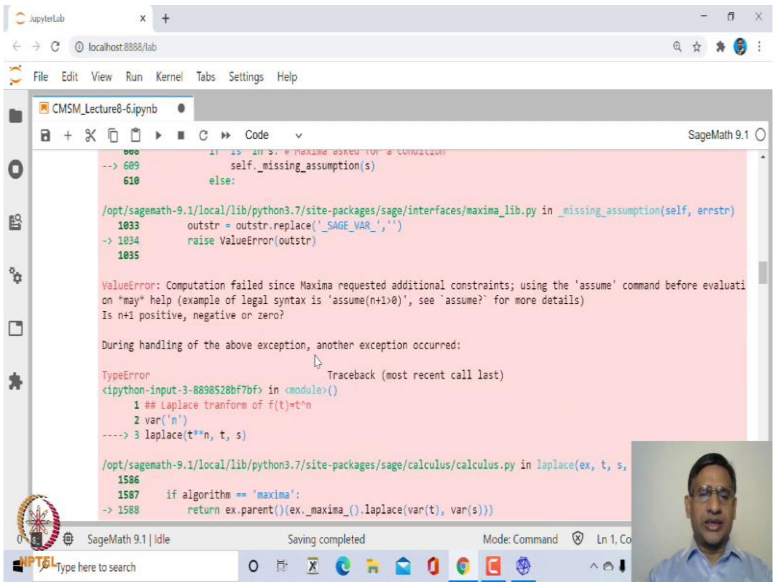
So, next we can find out Laplace transform of f(t) is equal to t to the power n, where n is any integer. Now, if you try to apply this Laplace transform of t to the power n comma t comma s it gives you a RuntimeError.

(Refer Slide Time: 05:52)



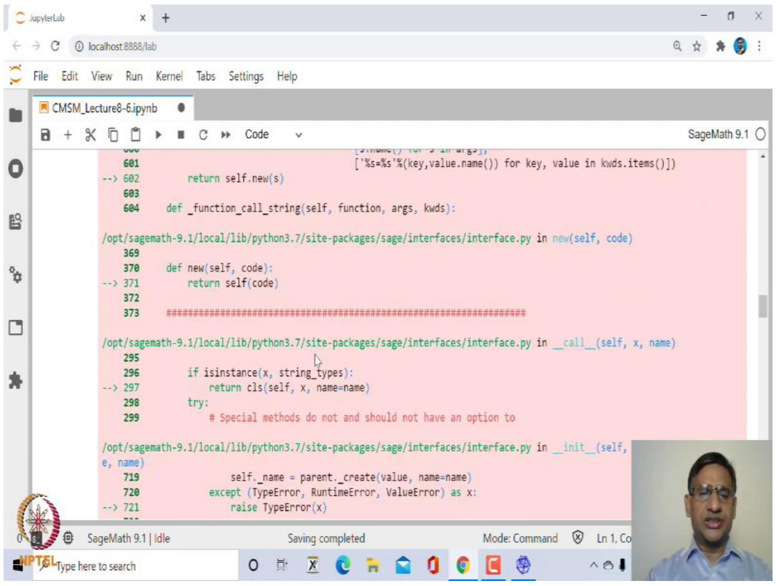
```
RuntimeError                              Traceback (most recent call last)
/opt/sagemath-9.1/local/lib/python3.7/site-packages/sage/interfaces/maxima_lib.py in _create(self, value, name)
    604             else:
--> 605                 self.set(name, value)
    606         except RuntimeError as error:

/opt/sagemath-9.1/local/lib/python3.7/site-packages/sage/interfaces/maxima_lib.py in set(self, var, value)
    513         cmd = '%s : %s$'%(var, value.rstrip(';'))
--> 514         self.eval(cmd)
    515 

/opt/sagemath-9.1/local/lib/python3.7/site-packages/sage/interfaces/maxima_lib.py in _eval_line(self, line, locals, refo
rmat, **kwds)
    459                 if statement:
--> 460                     maxima_eval("#$%s$" % statement)
    461         if not reformat:

/opt/sagemath-9.1/local/lib/python3.7/site-packages/sage/libs/ecl.pyx in sage.libs.ecl.EclObject.__call__ (build/cythoni
zed/sage/libs/ecl.c:7794)()
    804         lispargs = EclObject(list(args))
--> 805         return ecl_wrap(ecl_safe_apply(self.obj,(<EclObject>lispargs).obj))
    806 

/opt/sagemath-9.1/local/lib/python3.7/site-packages/sage/libs/ecl.pyx in sage.libs.ecl.ecl_safe_app
```

(Refer Slide Time: 05:55)



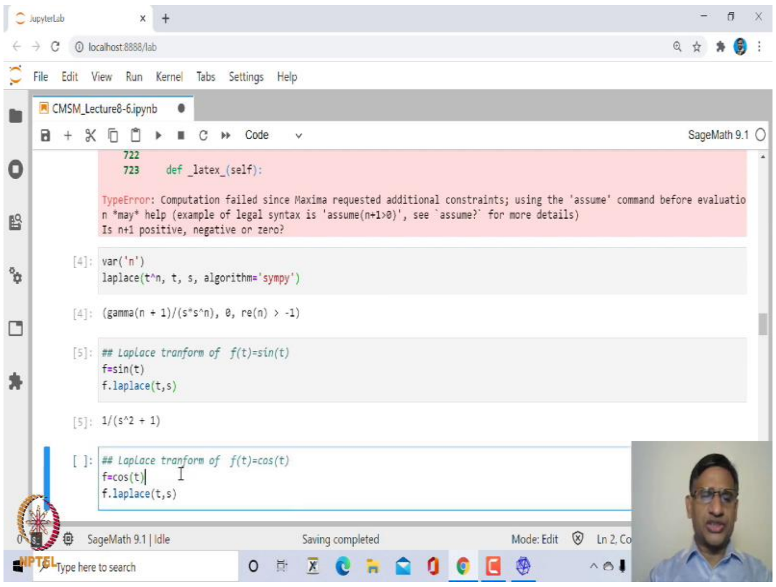
```
--> 609             self._missing_assumption(s)
    610         else:

/opt/sagemath-9.1/local/lib/python3.7/site-packages/sage/interfaces/maxima_lib.py in _missing_assumption(self, errstr)
   1033         outstr = outstr.replace('_SAGE_VAR_','')
-> 1034         raise ValueError(outstr)
   1035

ValueError: Computation failed since Maxima requested additional constraints; using the 'assume' command before evaluati
on *may* help (example of legal syntax is 'assume(n+1>0)', see `assume?` for more details)
Is n+1 positive, negative or zero?

During handling of the above exception, another exception occurred:

TypeError                                 Traceback (most recent call last)
<ipython-input-3-8898528bf7bf> in <module>()
      1 ## Laplace tranform of f(t)=t^n
      2 var('n')
----> 3 laplace(t**n, t, s)

/opt/sagemath-9.1/local/lib/python3.7/site-packages/sage/calculus/calculus.py in laplace(ex, t, s,
   1586
   1587     if algorithm == 'maxima':
-> 1588         return ex.parent()(ex._maxima_().laplace(var(t), var(s)))
```

(Refer Slide Time: 05:56)



```
    601                                      ['%s=%s'%(key,value.name()) for key, value in kwds.items()])
--> 602         return self.new(s)
    603
    604     def _function_call_string(self, function, args, kwds):

/opt/sagemath-9.1/local/lib/python3.7/site-packages/sage/interfaces/interface.py in new(self, code)
    369
    370     def new(self, code):
--> 371         return self(code)
    372
    373     ###################################################################

/opt/sagemath-9.1/local/lib/python3.7/site-packages/sage/interfaces/interface.py in __call__(self, x, name)
    295
    296         if isinstance(x, string_types):
--> 297             return cls(self, x, name=name)
    298         try:
    299             # Special methods do not and should not have an option to

/opt/sagemath-9.1/local/lib/python3.7/site-packages/sage/interfaces/interface.py in __init__(self,
e, name)
    719                 self._name = parent._create(value, name=name)
    720             except (TypeError, RuntimeError, ValueError) as x:
--> 721                 raise TypeError(x)
```

(Refer Slide Time: 05:57)



```
    722
    723     def _latex_(self):

TypeError: Computation failed since Maxima requested additional constraints; using the 'assume' command before evaluatio
n *may* help (example of legal syntax is 'assume(n+1>0)', see `assume?` for more details)
Is n+1 positive, negative or zero?
```

```
[4]: var('n')
     laplace(t^n, t, s, algorithm='sympy')

[4]: (gamma(n + 1)/(s*s^n), 0, re(n) > -1)

[5]: ## Laplace tranform of  f(t)=sin(t)
     f=sin(t)
     f.laplace(t,s)

[5]: 1/(s^2 + 1)

[ ]: ## Laplace tranform of  f(t)=cos(t)
     f=cos(t)
     f.laplace(t,s)
```

And if you go down, what it says is computation failed since maxima requested additional constraint using assume function. So, it requires extra assumption on n, it is asking whether n plus 1 is positive or negative . So, when you say Laplace function in SageMath it uses different algorithm one of them is using maxima and there are other methods as well.

So, in case you mentioned algorithm is equal to sympy or you can mention what happens to n plus 1 if n plus 1 is positive then also it may work. Here we can instead of using the default algorithm which is maxia, we use sympy symbolic Python package and then it will work.

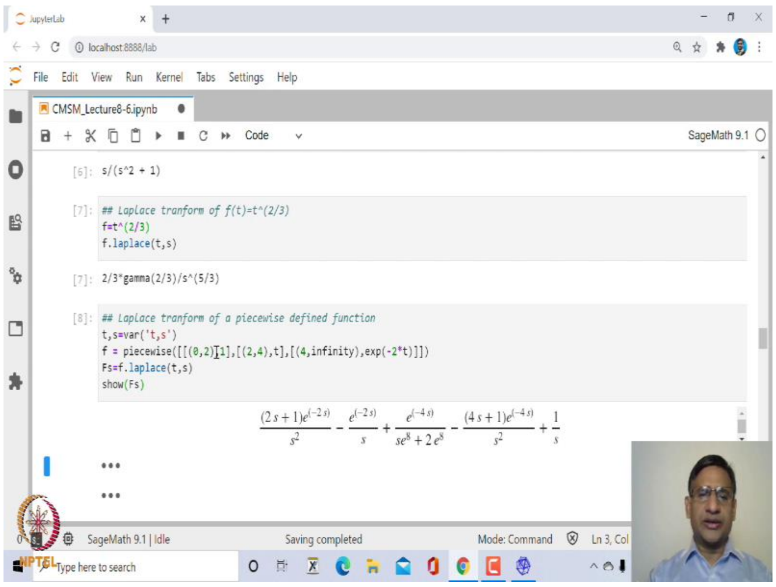
So, it tells you that Laplace transform of t to the power n is equal to gamma n plus 1 divided by actually this should be s to the power n plus 1. So, that is the gamma n plus 1, is nothing but n factorial for integer positive integer n. We have seen that gamma n plus 1 is n factorial, therefore, Laplace transform of t to the power n, where n is positive integer is nothing but gamma n+1 divided by s to the power n plus 1.

And of course, here it is already putting this condition that real part of n is bigger than minus 1. So, that is same as saying n plus 1 is a positive right.

Let us look at some more functions and its Laplace transforms.

Suppose f is sin(t), then its Laplace transform is 1 upon s square plus 1. Similarly, if you look at Laplace transform of cos(t) then it is s upon s square plus 1.

(Refer Slide Time: 07:52)



```
[6]: s/(s^2 + 1)

[7]: ## Laplace tranform of f(t)=t^(2/3)
     f=t^(2/3)
     f.laplace(t,s)

[7]: 2/3*gamma(2/3)/s^(5/3)

[8]: ## Laplace tranform of a piecewise defined function
     t,s=var('t,s')
     f = piecewise([[(0,2),1],[(2,4),t],[(4,infinity),exp(-2*t)]])
     Fs=f.laplace(t,s)
     show(Fs)
```

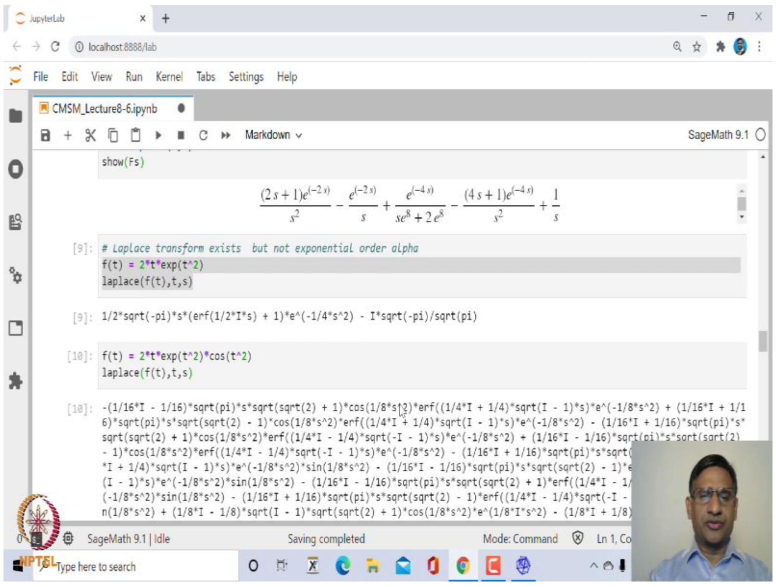
$$\frac{(2s+1)e^{(-2s)}}{s^2} - \frac{e^{(-2s)}}{s} + \frac{e^{(-4s)}}{se^8 + 2e^8} - \frac{(4s+1)e^{(-4s)}}{s^2} + \frac{1}{s}$$

If you find Laplace transform of t to the power 2 by 3, its not necessary that you should always have integer power, but you can have any positive power and you can find its Laplace transform. So, in this case Laplace transform of t to the power 2 by 3 is 2 by 3 into gamma of 2 by 3 divided by s to the power of 5 by 3, 5 by 3 is nothing but 1 plus 2 by 3.

So, you can find Laplace transform of standard functions, of course, in case the Laplace transform does not exist then it will not give you answer. You can also find Laplace transform of piecewise define function. Suppose we take a function f which is defined as 1 in the interval 0 to 2 it is defined as t in interval 2 to 4 and it is defined as exponential of minus 2 t in the interval 4 to infinity.

You can find Laplace transform of this, which is something like this, its looks quite complicated. But this is piecewise defined function defined in 3 pieces namely 0 to 2, 2 to 4 and 4 to infinity. You can also define for example, Laplace transform of 2 t into exponential of t square.

(Refer Slide Time: 09:22)



```
show(Fs)
```

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

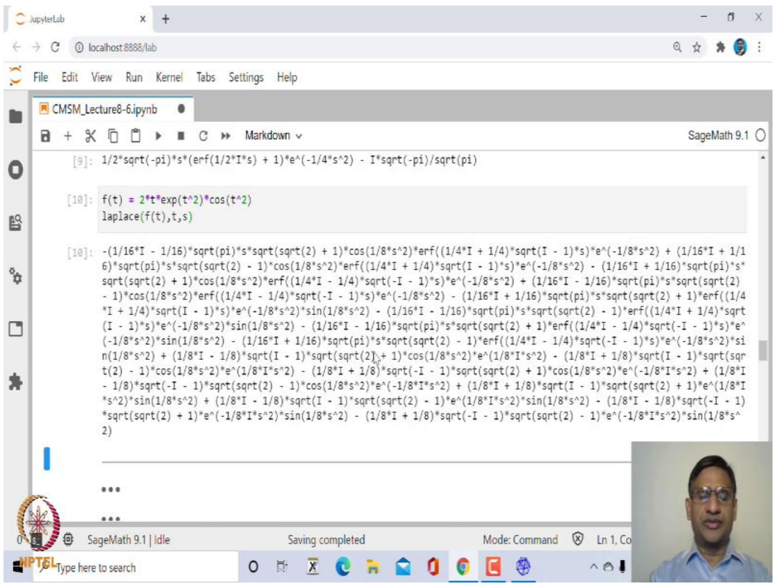
```
[9]: # Laplace transform exists  but not exponential order alpha
     f(t) = 2*t*exp(t^2)
     laplace(f(t),t,s)

[9]: 1/2*sqrt(-pi)*s*(erf(1/2*I*s) + 1)*e^(-1/4*s^2) - I*sqrt(-pi)/sqrt(pi)

[10]: f(t) = 2*t*exp(t^2)*cos(t^2)
      laplace(f(t),t,s)

[10]: -(1/16*I - 1/16)*sqrt(pi)*s*sqrt(sqrt(2) + 1)*cos(1/8*s^2)*erf((1/4*I + 1/4)*sqrt(I - 1)*s)*e^(-1/8*s^2) + (1/16*I + 1/1
      6)*sqrt(pi)*s*sqrt(sqrt(2) - 1)*cos(1/8*s^2)*erf((1/4*I + 1/4)*sqrt(I - 1)*s)*e^(-1/8*s^2) - (1/16*I + 1/16)*sqrt(pi)*s*
      sqrt(sqrt(2) + 1)*cos(1/8*s^2)*erf((1/4*I - 1/4)*sqrt(-I - 1)*s)*e^(-1/8*s^2) + (1/16*I - 1/16)*sqrt(pi)*s*sqrt(sqrt(2)
      - 1)*cos(1/8*s^2)*erf((1/4*I - 1/4)*sqrt(-I - 1)*s)*e^(-1/8*s^2) - (1/16*I + 1/16)*sqrt(pi)*s*sqrt(
      *I + 1/4)*sqrt(I - 1)*s)*e^(-1/8*s^2)*sin(1/8*s^2) - (1/16*I - 1/16)*sqrt(pi)*s*sqrt(sqrt(2) - 1)*e
      (I - 1)*s)*e^(-1/8*s^2)*sin(1/8*s^2) - (1/16*I - 1/16)*sqrt(pi)*s*sqrt(sqrt(2) + 1)*erf((1/4*I - 1/
      (-1/8*s^2)*sin(1/8*s^2) - (1/16*I + 1/16)*sqrt(pi)*s*sqrt(sqrt(2) - 1)*erf((1/4*I - 1/4)*sqrt(-I -
      n(1/8*s^2) + (1/8*I - 1/8)*sqrt(I - 1)*sqrt(sqrt(2) + 1)*cos(1/8*s^2)*e^(1/8*I*s^2) - (1/8*I + 1/8)
```

In fact, this exponential of t square is not a function having exponential order alpha, but still its Laplace transform you will be able to find. It is given in terms of error function and this is complex so, when we says Laplace transform of f (t), is capital F(s), this s could be complex. So, it converts this real domain t into complex domain s.

Therefore, this exponential order alpha is not a necessary and sufficient condition it is not a sufficient condition. Here is an example of a function which is not of exponential order alpha, but still its Laplace transform exist.

Similarly you can try even this more complicated function. Let us say, 2 t into exponential of t square into cos t square. This is again not having exponential order alpha but, still you can find its Laplace transform and it may be quite complicated.

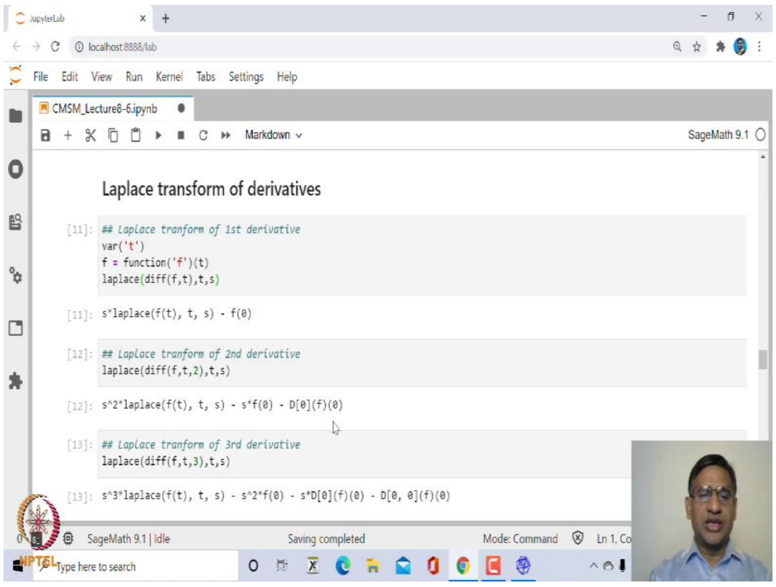
(Refer Slide Time: 10:34)



```
[9]: 1/2*sqrt(-pi)*s*(erf(1/2*I*s) + 1)*e^(-1/4*s^2) - I*sqrt(-pi)/sqrt(pi)

[10]: f(t) = 2*t*exp(t^2)*cos(t^2)
      laplace(f(t),t,s)

[10]: -(1/16*I - 1/16)*sqrt(pi)*s*sqrt(sqrt(2) + 1)*cos(1/8*s^2)*erf((1/4*I + 1/4)*sqrt(I - 1)*s)*e^(-1/8*s^2) + (1/16*I + 1/1
      6)*sqrt(pi)*s*sqrt(sqrt(2) - 1)*cos(1/8*s^2)*erf((1/4*I + 1/4)*sqrt(I - 1)*s)*e^(-1/8*s^2) - (1/16*I + 1/16)*sqrt(pi)*s*
      sqrt(sqrt(2) + 1)*cos(1/8*s^2)*erf((1/4*I - 1/4)*sqrt(-I - 1)*s)*e^(-1/8*s^2) + (1/16*I - 1/16)*sqrt(pi)*s*sqrt(sqrt(2)
      - 1)*cos(1/8*s^2)*erf((1/4*I - 1/4)*sqrt(-I - 1)*s)*e^(-1/8*s^2) - (1/16*I + 1/16)*sqrt(pi)*s*sqrt(sqrt(2) + 1)*erf((1/4
      *I + 1/4)*sqrt(I - 1)*s)*e^(-1/8*s^2)*sin(1/8*s^2) - (1/16*I - 1/16)*sqrt(pi)*s*sqrt(sqrt(2) - 1)*erf((1/4*I + 1/4)*sqrt
      (I - 1)*s)*e^(-1/8*s^2)*sin(1/8*s^2) - (1/16*I - 1/16)*sqrt(pi)*s*sqrt(sqrt(2) + 1)*erf((1/4*I - 1/4)*sqrt(-I - 1)*s)*e^
      (-1/8*s^2)*sin(1/8*s^2) - (1/16*I + 1/16)*sqrt(pi)*s*sqrt(sqrt(2) - 1)*erf((1/4*I - 1/4)*sqrt(-I - 1)*s)*e^(-1/8*s^2)*si
      n(1/8*s^2) + (1/8*I - 1/8)*sqrt(I - 1)*sqrt(sqrt(2) + 1)*cos(1/8*s^2)*e^(1/8*I*s^2) - (1/8*I + 1/8)*sqrt(I - 1)*sqrt(sqr
      t(2) - 1)*cos(1/8*s^2)*e^(1/8*I*s^2) - (1/8*I + 1/8)*sqrt(-I - 1)*sqrt(sqrt(2) + 1)*cos(1/8*s^2)*e^(-1/8*I*s^2) + (1/8*I
      - 1/8)*sqrt(-I - 1)*sqrt(sqrt(2) - 1)*cos(1/8*s^2)*e^(-1/8*I*s^2) + (1/8*I + 1/8)*sqrt(I - 1)*sqrt(sqrt(2) + 1)*e^(1/8*I
      *s^2)*sin(1/8*s^2) + (1/8*I - 1/8)*sqrt(I - 1)*sqrt(sqrt(2) - 1)*e^(1/8*I*s^2)*sin(1/8*s^2) - (1/8*I - 1/8)*sqrt(-I - 1)
      *sqrt(sqrt(2) + 1)*e^(-1/8*I*s^2)*sin(1/8*s^2) - (1/8*I + 1/8)*sqrt(-I - 1)*sqrt(sqrt(2) - 1)*e^(-1/8*I*s^2)*sin(1/8*s^
      2)
```

You can see here, it is very very complicated function, but you can see Sage is able to find Laplace transform of this kind of functions.

(Refer Slide Time: 10:48)



Laplace transform of derivatives

```
[11]: ## Laplace tranform of 1st derivative
      var('t')
      f = function('f')(t)
      laplace(diff(f,t),t,s)

[11]: s*laplace(f(t), t, s) - f(0)

[12]: ## Laplace tranform of 2nd derivative
      laplace(diff(f,t,2),t,s)

[12]: s^2*laplace(f(t), t, s) - s*f(0) - D[0](f)(0)

[13]: ## Laplace tranform of 3rd derivative
      laplace(diff(f,t,3),t,s)

[13]: s^3*laplace(f(t), t, s) - s^2*f(0) - s*D[0](f)(0) - D[0, 0](f)(0)
```

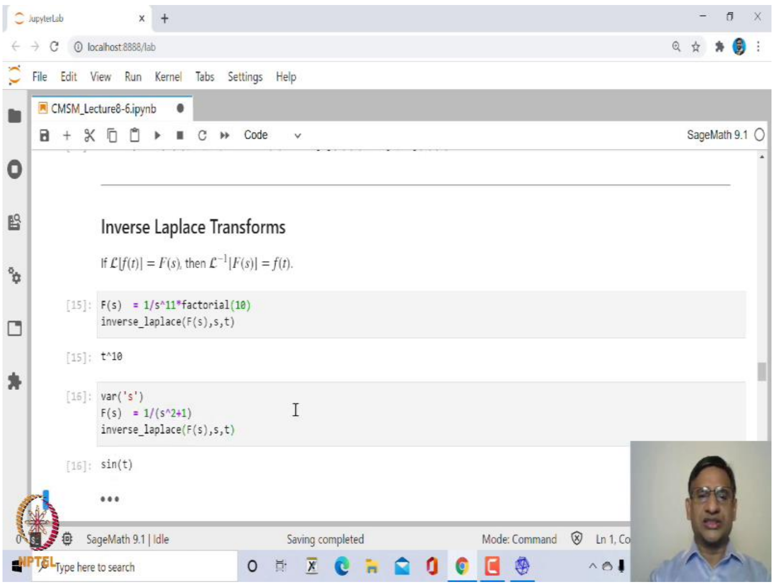
Next, let us look at Laplace transform of derivatives. Actually when you have differential equations, it is an equation in various derivative of the function. So, it should know how to find Laplace transform of derivative, and what is it. So, in case we take f is a function of t and find Laplace transform of 1st derivative, then what we get is s into Laplace transform of f(t) and minus f(0). This can be very easily proved using the first fundamental theorem of integral calculus. That is where this f(0) term is coming. Similarly you can find laplace transform of 2nd derivative of f, and in this case, it is s square into laplace transform of f(t) minus s into f 0 minus D 0 D square bracket 0, stands for f dash at 0.

And you can extend this to Laplace transform of 3rd derivative, and in this case you will get s cube into laplace transform of f minus s square into f0 minus s into f dash at 0 minus minus f double dash at 0. You can very easily extend this to Laplace transform of nth derivative.

If you just try to see this there is a pattern, and if you look at this pattern you can find Laplace transform of nth derivative of the function. What it should be? It should be s to the power n into Laplace transform of f(t) minus s to the power n-1 into f (0) minus s to the power n minus 2 into f dash at 0 and dot dot dot dot, the last one will be n minus 1th derivative of f at 0. So, that is how you can apply Laplace transform to derivative of a function.

So, this is what we will be using in order to solve differential equation using Laplace transform. So, and you can see here in this requires f at 0 f dash at 0 and so on. So, in case you have initial value differential equation all these things are given to you and hence, you will be able to solve this and this is what we will see right.

(Refer Slide Time: 13:11)



Inverse Laplace Transforms

If $\mathcal{L}[f(t)] = F(s)$, then $\mathcal{L}^{-1}[F(s)] = f(t)$.

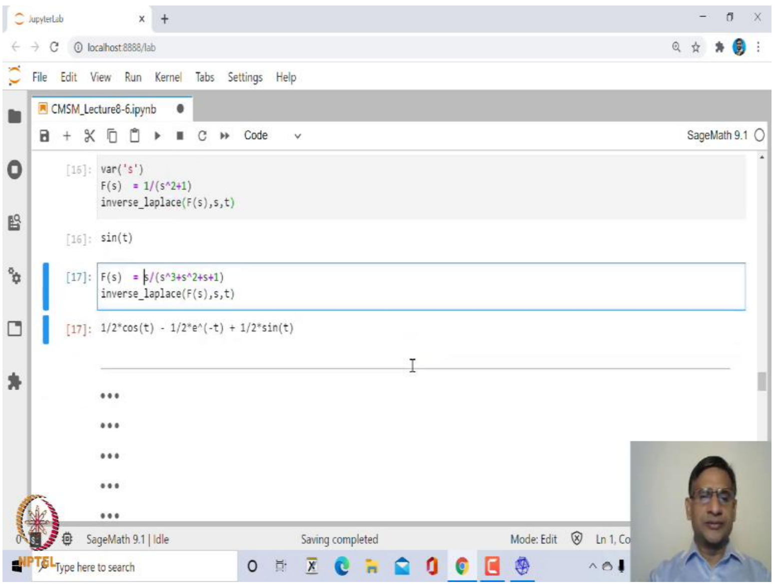
```
[15]: F(s)  = 1/s^11*factorial(10)
      inverse_laplace(F(s),s,t)

[15]: t^10

[16]: var('s')
      F(s)  = 1/(s^2+1)
      inverse_laplace(F(s),s,t)

[16]: sin(t)
```

Next, let us look at how you can find Inverse Laplace Transform. As I mentioned in case Laplace transform of f(t) is capital F(s) then inverse Laplace transform of capital F(s) is f(t). Let us look at few examples. If you want to find the inverse Laplace transform of 1 upon s to the power 11. Now, you must have noticed that Laplace transform of t to the power 10 is going to be 10 factorial divided by s to the power 11, therefore, inverse Laplace transform is going to be some multiple of t to the power 10. Let us find it. It is says that, it is 1 upon 3628800 which is nothing but 10 factorial into Laplace transform of f(t) into t to the power 10.

So, you can find that Laplace transform of this, is going to be 1 upon s to the power 11. For example, if you if you try to multiply this by factorial 10 factorial and then find its inverse Laplace transform, you will t to the power 10 right. So, similarly you can find let us say inverse Laplace transform of 1 upon s square plus 1.

We saw that Laplace transform of sin(t) is 1 upon 1 plus s square, therefore, we should get as sin(t), that is correct.

(Refer Slide Time: 14:40)



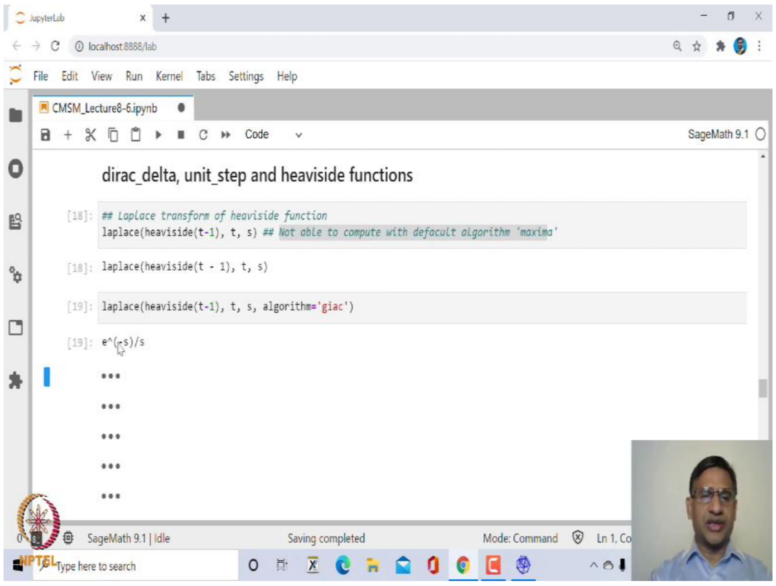
```
[16]: var('s')
      F(s)  = 1/(s^2+1)
      inverse_laplace(F(s),s,t)

[16]: sin(t)

[17]: F(s)  = s/(s^3+s^2+s+1)
      inverse_laplace(F(s),s,t)

[17]: 1/2*cos(t) - 1/2*e^(-t) + 1/2*sin(t)
```

And we can find inverse Laplace transform of even complicated functions.

So, for example, suppose capital F(s) is s upon s to the power 3 plus s square plus s plus 1 and if you find inverse Laplace transform this, it is given by half into cos(t) minus half e to the power minus t plus half sin(t).

You will be able to find Laplace transform of an expression in s, but not every expression will have inverse Laplace transform, sometimes it may become very complicated. Also not for every expression you will have inverse Laplace transform.

(Refer Slide Time: 15:18)


dirac_delta, unit_step and heaviside functions

```
[18]: ## Laplace transform of heaviside function
      laplace(heaviside(t-1), t, s) ## Not able to compute with defacult algorithm 'maxima'

[18]: laplace(heaviside(t - 1), t, s)

[19]: laplace(heaviside(t-1), t, s, algorithm='giac')

[19]: e^(-s)/s
```

Next, let us look at this how we can make use of this dirac delta function, unit step function and heaviside function.

This is used quite frequently in this notion of Laplace transform, many differential equations will have this kind of functions.

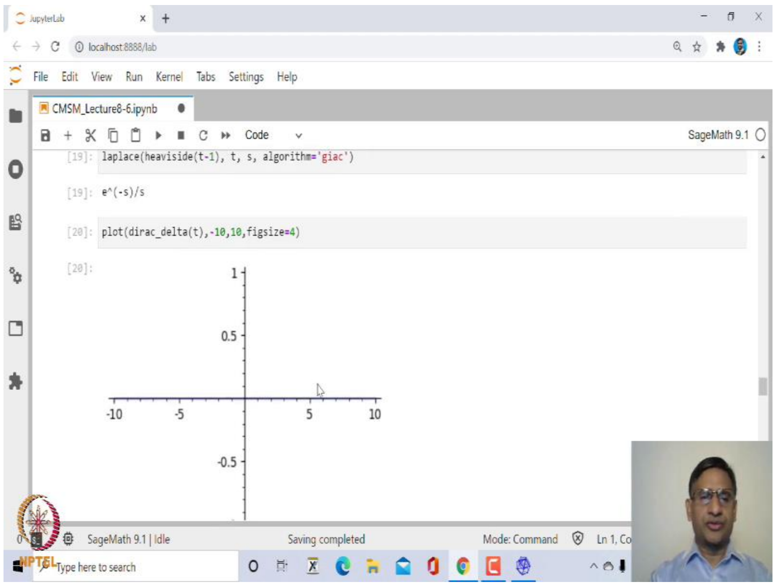
So, let us look at, we can find inverse Laplace transform of heaviside function (t-1). If you look at heaviside function, by definition itself it means that it is one side heavy.

Heaviside(t-1) will be 0 when t is less than 1 and when t is greater than 1 it will be 1. But, if you try to find Laplace transform of heaviside function it does not give you any output, that is because again using default algorithm it is unable to compute this. However, if you use another algorithm which is called ‘giac’ this is an algorithm in SageMath, to find laplace transform, then you will be able to find its Laplace transform.

Let us just wait for a second, it is e to the power minus s upon s. So, laplace transform of this function is e to the power minus s upon s.

You could have also defined this as a piecewise function and obtain its laplace transform like this.

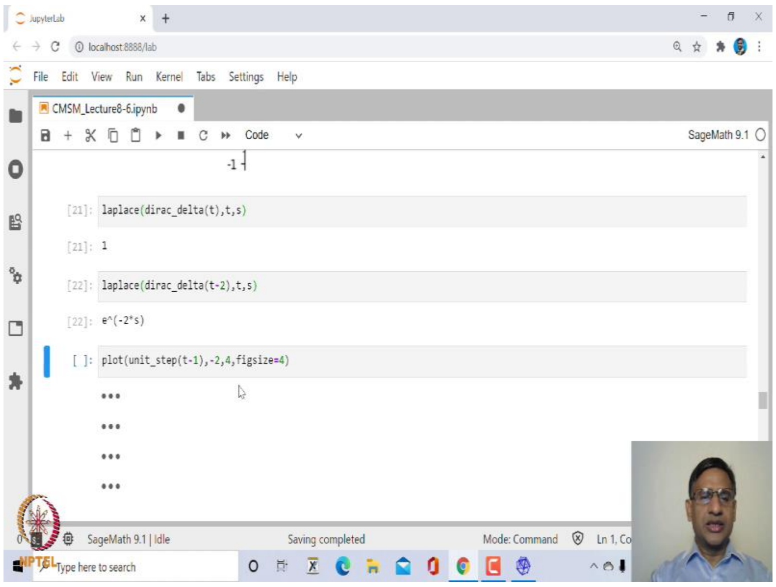
(Refer Slide Time: 16:44)



```
[19]: laplace(heaviside(t-1), t, s, algorithm='giac')

[19]: e^(-s)/s

[20]: plot(dirac_delta(t),-10,10,figsize=4)

[20]:
```

Now let us look at, what is dirac delta function. If you plot dirac delta function between minus 10 and 10, you can see here this is almost like a 0, only at 0 it will be like infinity and as a result, in such a way that the integral of dirac delta function from minus infinity to infinity is equal to 1.

(Refer Slide Time: 17:08)

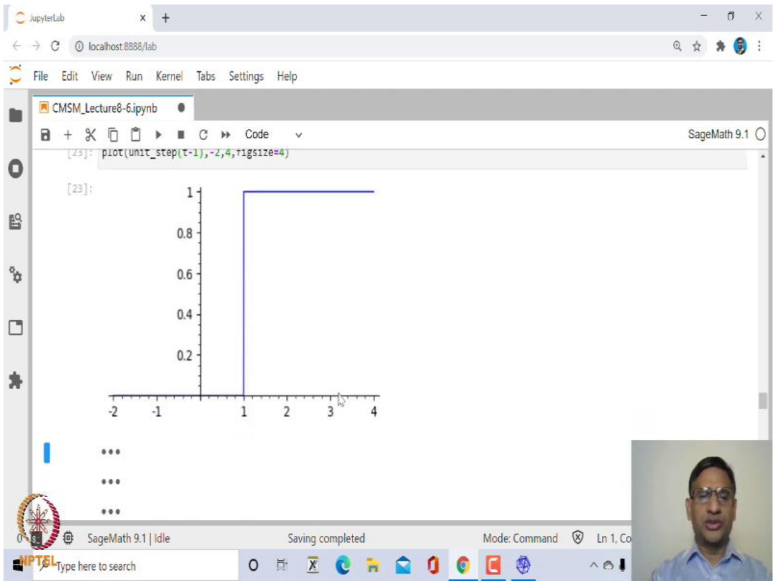


```
[21]: laplace(dirac_delta(t),t,s)

[21]: 1

[22]: laplace(dirac_delta(t-2),t,s)

[22]: e^(-2*s)

[ ]: plot(unit_step(t-1),-2,4,figsize=4)
```

You can find the laplace transform of dirac delta function and it turns out to be it is 1. So, for dirac delta function, the laplace transform is 1. If you look at laplace transform of dirac delta t minus 2, this will be e to the power minus 2s. So, you can find laplace transform of dirac delta functions defined at various t.

Similarly, you can look at unit step function. The unit step function for example, let us plot unit step function (t-1), between minus 2 and 4. This is actually going to be 0 when t is less than 1 and is equal to 1 when t is bigger than 1. So, this is actually a kind of Heaviside function.

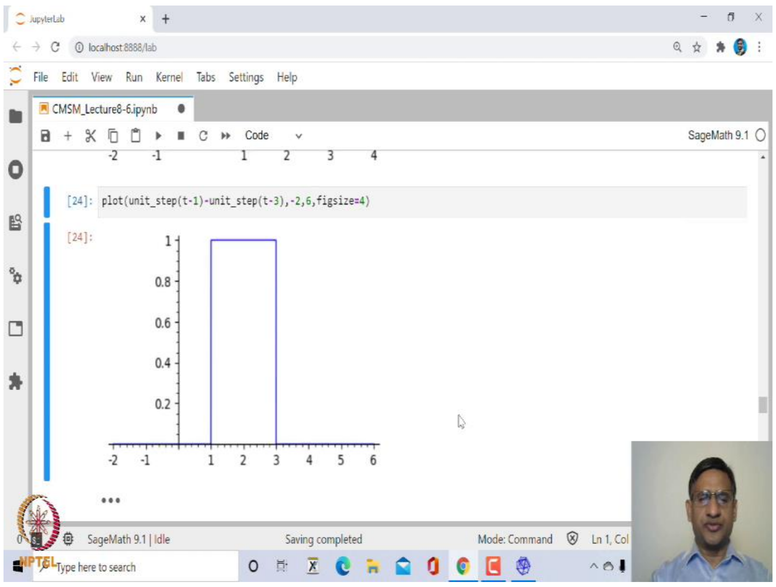
(Refer Slide Time: 17:57)



```
[23]: plot(unit_step(t-1),-2,4,figsize=4)
```

So, this is how the function is defined so, to left of 1 it is 0 and to the right of 1 it is 1. Of course, this vertical line you can ignore, we have seen that.

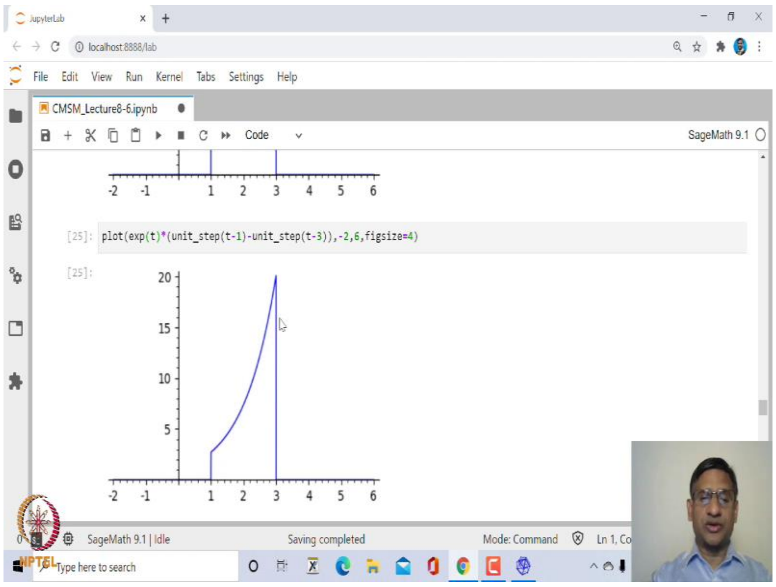
(Refer Slide Time: 18:15)



```
[24]: plot(unit_step(t-1)-unit_step(t-3),-2,6,figsize=4)
```

Next, we can look at for example, if I say unit step (t -1) minus unit step (t-3), then if you plot, this graph you can see here, this is this is how it looks like. So, between 1 and 3 it has become 1 and remaining all the places it is 0. So, you can think of this as a signal, between this and this signal is on otherwise it is off.

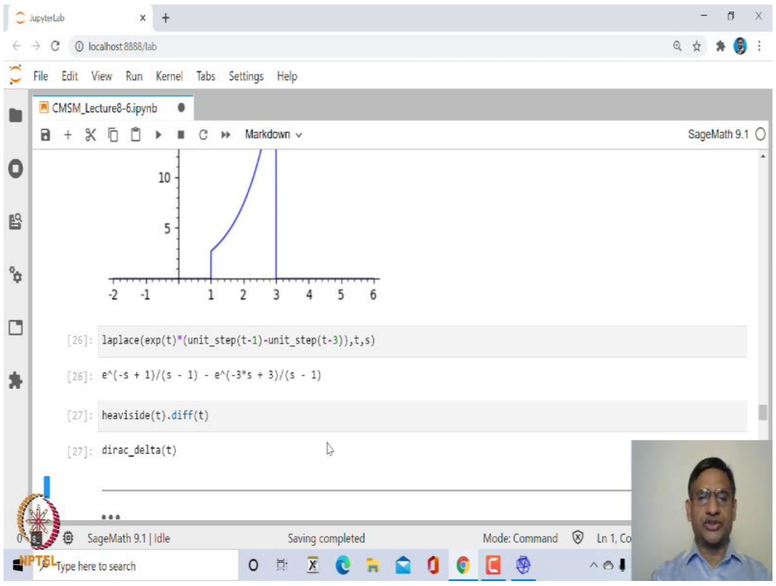
(Refer Slide Time: 18:40)



```
[25]: plot(exp(t)*(unit_step(t-1)-unit_step(t-3)),-2,6,figsize=4)
```

You can even include, some more parts. Let us say, unit step function(t-1) minus 1 minus unit step function (t-3) and multiply this by exponential of t now, you can see here, between 1 and 3 it will become exponential of t. Yeah, between 1 and 3 the function varies as exponential of 3 remaining place exponential of t and remaining places it is 0.

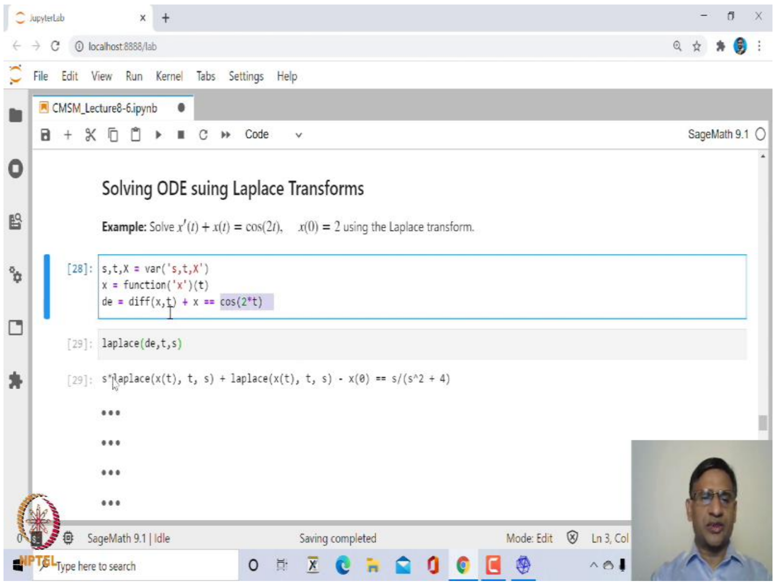
(Refer Slide Time: 19:14)



```
[26]: laplace(exp(t)*(unit_step(t-1)-unit_step(t-3)),t,s)

[26]: e^(-s + 1)/(s - 1) - e^(-3*s + 3)/(s - 1)

[27]: heaviside(t).diff(t)

[27]: dirac_delta(t)
```

Next we can find out even laplace transform of exponential of t into unit step t minus 1 minus unit step t minus 3. You can even find the derivative of heaviside function with respect to t and it is nothing but dirac delta function. So, derivative of unit step heaviside function is dirac delta function.

(Refer Slide Time: 19:45)



Solving ODE suing Laplace Transforms

Example: Solve $x'(t) + x(t) = \cos(2t)$, $x(0) = 2$ using the Laplace transform.

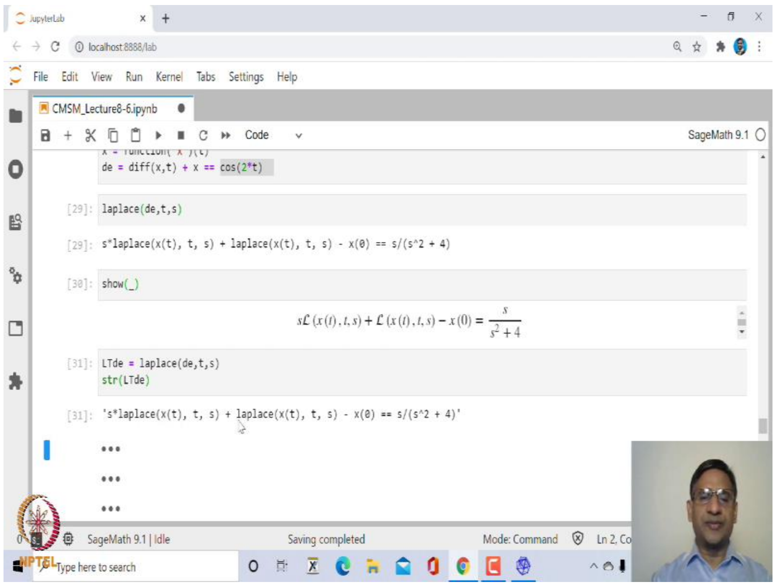
```
[28]: s,t,X = var('s,t,X')
      x = function('x')(t)
      de = diff(x,t) + x == cos(2*t)

[29]: laplace(de,t,s)

[29]: s*laplace(x(t), t, s) + laplace(x(t), t, s) - x(0) == s/(s^2 + 4)
```

So, let us look at how we can make use of this notion of Laplace Transform to solve differential equations. What you saw was a Laplace transform of derivative of function is nothing but a polynomial in s, it is an algebraic expression. So, in case you have a differential equation, let us take an example. We want to solve x dash t is plus x(t) is equal to cos(2t) and where x(0)=2 using the Laplace transform.

What we do is, we take Laplace transform of the equation on both left hand side and right hand side. First let us define this differential equation as de which is derivative of x with respect to t plus x is equal to cos(2t). And then, we can find Laplace transform of this de.

So, what it will do? It will find the Laplace transform of the left hand side and Laplace transform on the right hand side. You can see here the left hand side Laplace transform has become s into laplace transform of x plus laplace transform of x minus x0 and then, this part including and minus x(0) is the Laplace transform of derivative of x, and this is Laplace transform of x and the right hand side is s upon s square plus 4, which is Laplace transform of cos(2t). So, this is an expression in s and this is Laplace transform etcetera, This will be a function f of s.

(Refer Slide Time: 21:22)



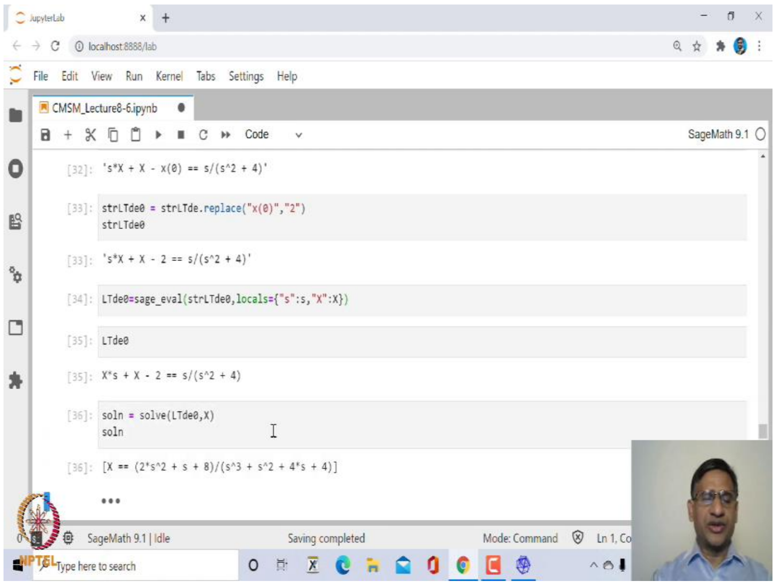
```
de = diff(x,t) + x == cos(2*t)

[29]: laplace(de,t,s)

[29]: s*laplace(x(t), t, s) + laplace(x(t), t, s) - x(0) == s/(s^2 + 4)

[30]: show(_)
```

$$s\mathcal{L}\left(x\left(t\right), t, s\right) + \mathcal{L}\left(x\left(t\right), t, s\right) - x\left(0\right) = \frac{s}{s^2+4}$$

```
[31]: LTde = laplace(de,t,s)
      str(LTde)

[31]: 's*laplace(x(t), t, s) + laplace(x(t), t, s) - x(0) == s/(s^2 + 4)'
```

We can ask it to show what is this. This is what, it is now. What we will do is, let me store this Laplace term of differential equation in LTde, Laplace transform of differential equation and convert it into a string. So, this entire thing is converted into string, each element here will be a string variable.

(Refer Slide Time: 21:47)



```
[32]: 's*X + X - x(0) == s/(s^2 + 4)'

[33]: strLTde0 = strLTde.replace("x(0)","2")
      strLTde0

[33]: 's*X + X - 2 == s/(s^2 + 4)'

[34]: LTde0=sage_eval(strLTde0,locals={"s":s,"X":X})

[35]: LTde0

[35]: X*s + X - 2 == s/(s^2 + 4)

[36]: soln = solve(LTde0,X)
      soln

[36]: [X == (2*s^2 + s + 8)/(s^3 + s^2 + 4*s + 4)]
```

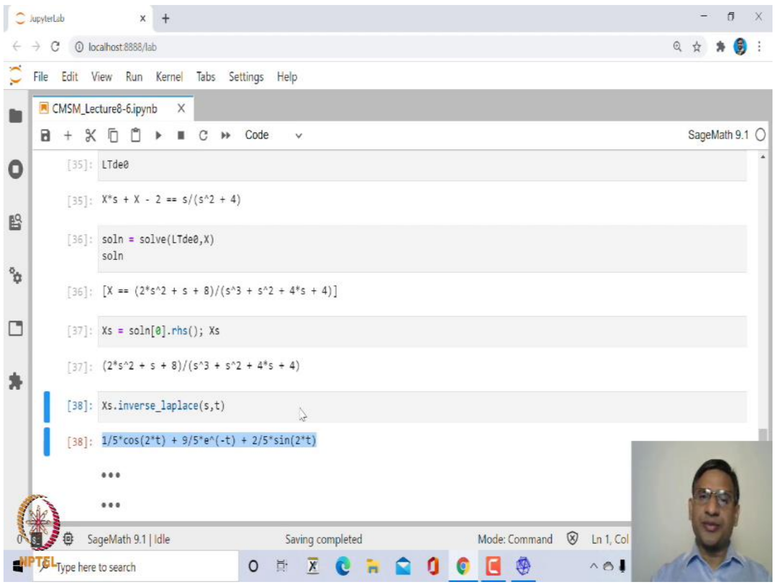
Now what we can do is this, wherever you have laplace(x(t), t s), that let us replace it by capital X. So, Laplace transform of small x, we are defining as capital X. Then let us look at what it is. So, it is now an equation in capital X. We will solve this for capital X So, capital X will be a function of s, and then we will find inverse Laplace transform of capital X, that will give you solution.

Now, this strLTde involves x(0), we know that x(0) is 2. So, let us replace x(0) by y 2. This is strLTd.replace, the string variable x in the bracket 0 by 2 and that let us store in strLTde. This initial value is also inserted, but this is still it is a string. What we will have to do, is we will evaluate this as an expression.

There is an inbuilt function called sage_eval and give the expression equation and then say locals is equal to so, here string is replaced by s and capital X str string variable is replaced by capital X.

So, now let us look at what is this, we have obtained. This is an expression, this is an expression in capital X so, that we can solve it for capital X. When we solve it for capital X, this is what we get. This is the right hand side, is a function of small s. Remember what was capital X, capital X was Laplace transform of small x. Therefore, if I want to get small x, which is a solution of this, we have to find inverse Laplace transform of capital X.

(Refer Slide Time: 23:56)

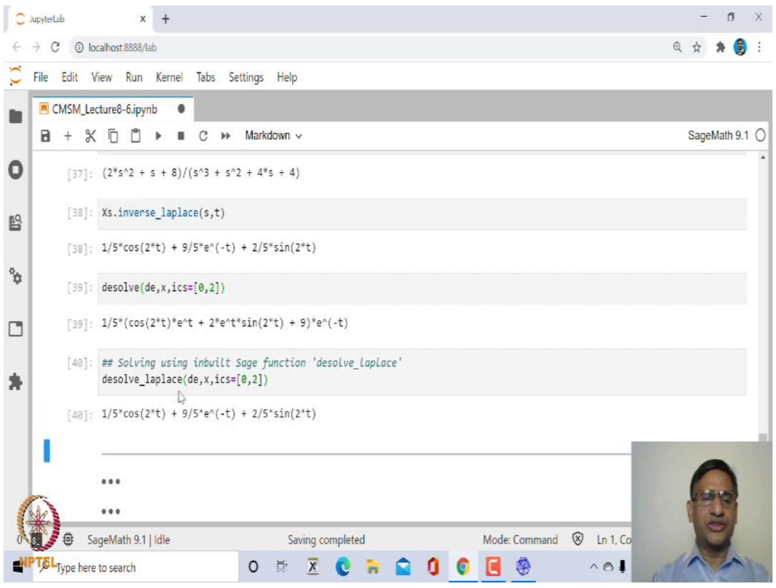


```
[35]: LTde0

[35]: X*s + X - 2 == s/(s^2 + 4)

[36]: soln = solve(LTde0,X)
      soln

[36]: [X == (2*s^2 + s + 8)/(s^3 + s^2 + 4*s + 4)]

[37]: Xs = soln[0].rhs(); Xs

[37]: (2*s^2 + s + 8)/(s^3 + s^2 + 4*s + 4)

[38]: Xs.inverse_laplace(s,t)

[38]: 1/5*cos(2*t) + 9/5*e^(-t) + 2/5*sin(2*t)
```

Let us solve, this solution and let us find right hand side of this, right hand side is this, and let us just find inverse Laplace transform of this X, which is capital X. So, that is what you get. This is the solution of this differential equation.

You can notice here, solving the differential equation by means of Laplace transform is very easy, all we need to do is, apply Laplace transform then, what we get is a polynomial in small s, and it also involve capital X, which is the Laplace transform of small x. Then we just have to solve for capital X and apply inverse Laplace transform. So, it is fairly simple and that is why it is quite useful.

(Refer Slide Time: 24:54)

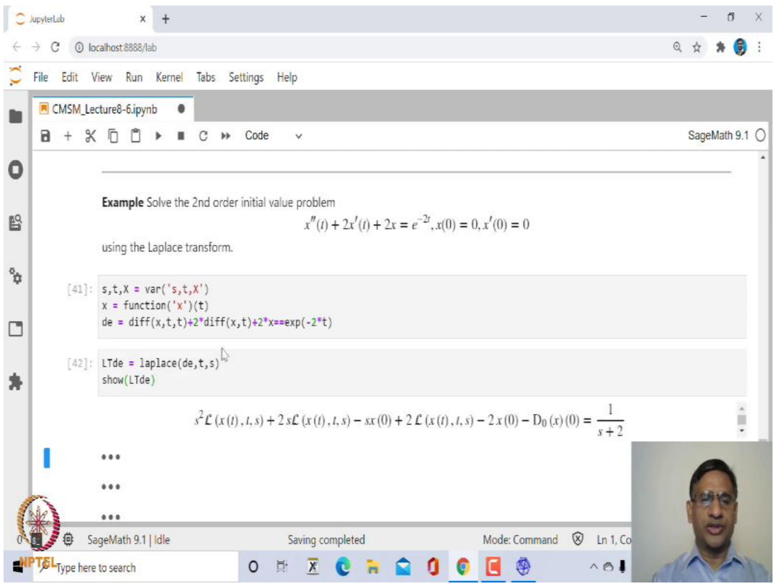


```
[37]: (2*s^2 + s + 8)/(s^3 + s^2 + 4*s + 4)

[38]: Xs.inverse_laplace(s,t)

[38]: 1/5*cos(2*t) + 9/5*e^(-t) + 2/5*sin(2*t)

[39]: desolve(de,x,ics=[0,2])

[39]: 1/5*(cos(2*t)*e^t + 2*e^t*sin(2*t) + 9)*e^(-t)

[40]: ## Solving using inbuilt Sage function 'desolve_laplace'
      desolve_laplace(de,x,ics=[0,2])

[40]: 1/5*cos(2*t) + 9/5*e^(-t) + 2/5*sin(2*t)
```

Now, let us try to verify this. If you use a inbuilt function desolve, then what we get is same as what we obtained using Laplace transform. As I said there is a inbuilt function to solve using Laplace transform in Sage which is desolve_laplace and then mention the differential equation, the variable in which you want to solve, and the initial condition and then, you get the answer which is again same as this.

We have not only solved this step by step, but we also verified this using inbuilt function desolve and also using desolve_laplace.

(Refer Slide Time: 25:36)



Example Solve the 2nd order initial value problem

$$x''(t) + 2x'(t) + 2x = e^{-2t}, x(0) = 0, x'(0) = 0$$

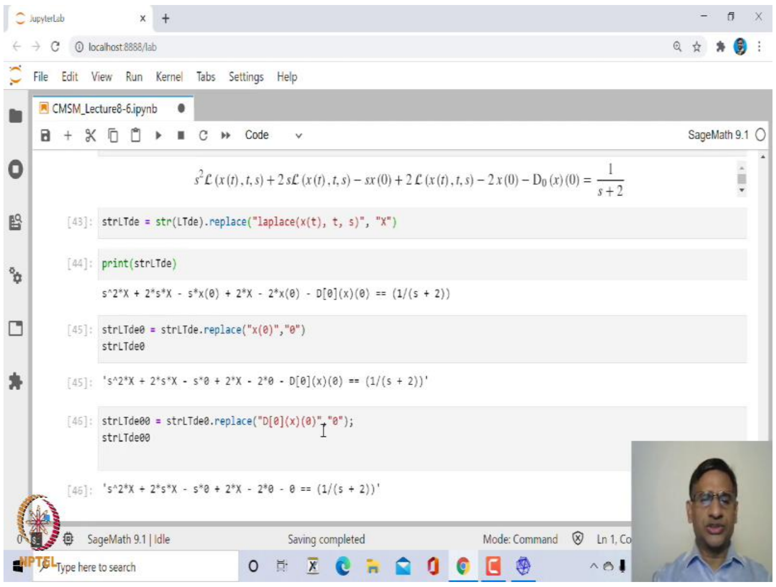
using the Laplace transform.

```
[41]: s,t,X = var('s,t,X')
      x = function('x')(t)
      de = diff(x,t,t)+2*diff(x,t)+2*x==exp(-2*t)

[42]: LTde = laplace(de,t,s)
      show(LTde)
```

$$s^2\mathcal{L}\left(x(t), t, s\right) + 2\,s\mathcal{L}\left(x(t), t, s\right) - sx(0) + 2\,\mathcal{L}\left(x(t), t, s\right) - 2\,x(0) - D_0(x)(0) = \frac{1}{s+2}$$

Now, you can apply this to 2nd order differential equation. Suppose you have a 2nd order differential equation with initial conditions. Then we can do the same thing. Let us solve this using Laplace transform. How do we do? First we define this differential equation then, apply Laplace transform on this differential equation.

(Refer Slide Time: 25:56)



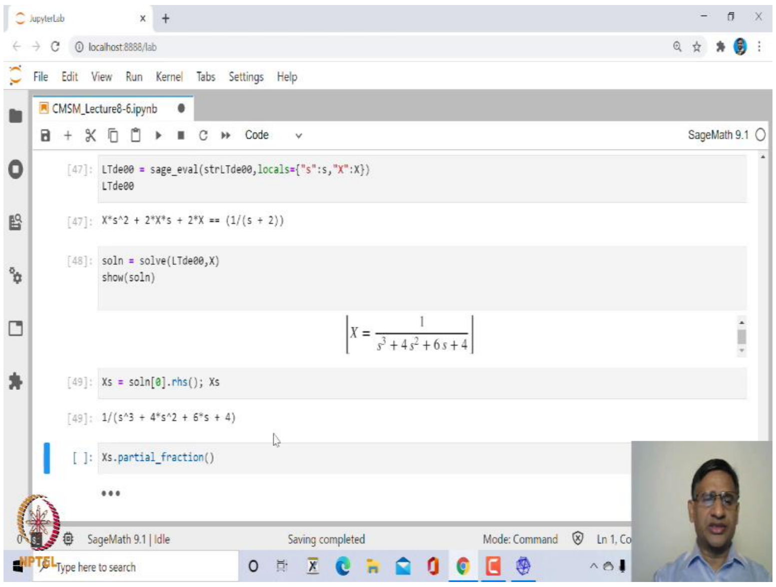
$$s^2\mathcal{L}\left(x(t),t,s\right)+2\,s\mathcal{L}\left(x(t),t,s\right)-sx(0)+2\,\mathcal{L}\left(x(t),t,s\right)-2\,x(0)-\mathrm{D}_0\left(x\right)(0)=\frac{1}{s+2}$$

```
[43]: strLTde = str(LTde).replace("laplace(x(t), t, s)", "X")

[44]: print(strLTde)

      s^2*X + 2*s*X - s*x(0) + 2*X - 2*x(0) - D[0](x)(0) == (1/(s + 2))

[45]: strLTde0 = strLTde.replace("x(0)","0")
      strLTde0

[45]: 's^2*X + 2*s*X - s*0 + 2*X - 2*0 - D[0](x)(0) == (1/(s + 2))'

[46]: strLTde00 = strLTde0.replace("D[0](x)(0)","0");
      strLTde00

[46]: 's^2*X + 2*s*X - s*0 + 2*X - 2*0 - 0 == (1/(s + 2))'
```

Then this is what we get and then what we do? We convert this into string and then replace wherever you have laplace(x(t),t,s) by capital X. So, that is what we do. Once we have converted this into string and it will be now string expressions in capital X small s and x(0) and x’(0).

So, let us print this is what you get, it involves x(0), which is known to us x(0) is 0 and, x’(0) is also 0. So, we will replace x(0) by 0 and D[0]x(0) by 0. Let us do that, we are first replace x(0) by 0 and then next we will replace D[0]x(0) also by 0. So, this is what we will get. This is a now expression in capital X and small s.

(Refer Slide Time: 26:58)



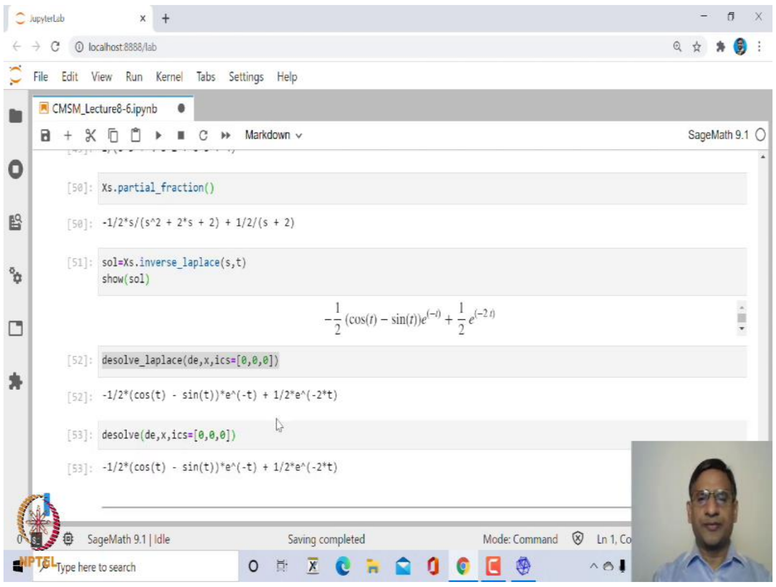
```
[47]: LTde00 = sage_eval(strLTde00,locals={"s":s,"X":X})
      LTde00

[47]: X*s^2 + 2*X*s + 2*X == (1/(s + 2))

[48]: soln = solve(LTde00,X)
      show(soln)
```

$$\left[X = \frac{1}{s^3 + 4s^2 + 6s + 4}\right]$$

```
[49]: Xs = soln[0].rhs(); Xs

[49]: 1/(s^3 + 4*s^2 + 6*s + 4)

[ ]: Xs.partial_fraction()
```

Now, we will evaluate this using sage_eval and substitute wherever you have string s by expression s and capital X by capital X. This is what we get. Now we have to solve this for capital X. When we solve this for capital X, this is how it looks like. Therefore, now we need to find inverse Laplace transform of the right hand side of this solution.

(Refer Slide Time: 27:28)



```
[50]: Xs.partial_fraction()

[50]: -1/2*s/(s^2 + 2*s + 2) + 1/2/(s + 2)

[51]: sol=Xs.inverse_laplace(s,t)
      show(sol)
```

$$-\frac{1}{2}\,(\cos(t) - \sin(t))e^{(-t)} + \frac{1}{2}\,e^{(-2\,t)}$$

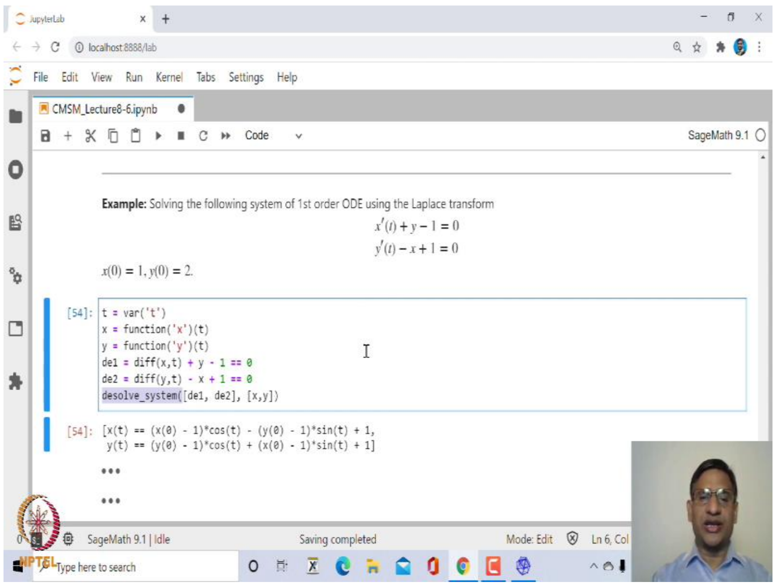
```
[52]: desolve_laplace(de,x,ics=[0,0,0])

[52]: -1/2*(cos(t) - sin(t))*e^(-t) + 1/2*e^(-2*t)

[53]: desolve(de,x,ics=[0,0,0])

[53]: -1/2*(cos(t) - sin(t))*e^(-t) + 1/2*e^(-2*t)
```

So, let us just take right hand side store it in Xs and then find inverse Laplace transform of this. You can directly find the inverse Laplace transform, you can also split this into partial fraction, in that case finding Laplace transform becomes much simpler. So, now let us take inverse Laplace transform of capital Xs this is what you get.

That is the solution of this 2nd order differential equation, and you can verify this using inbuilt function dedesolve_laplace with initial condition x(0) is 0 x'(0) is 0. This is what you can see. Both these things are same, of course, you can simplify this and you will get this expression.

So, both are the same you can also verify this using desolve function. When we say desolve and this differential equation with this initial condition this what you get right.

(Refer Slide Time: 28:20)



Example: Solving the following system of 1st order ODE using the Laplace transform

$$x'(t) + y - 1 = 0$$
$$y'(t) - x + 1 = 0$$

$x(0) = 1, y(0) = 2.$

```
[54]: t = var('t')
      x = function('x')(t)
      y = function('y')(t)
      de1 = diff(x,t) + y - 1 == 0
      de2 = diff(y,t) - x + 1 == 0
      desolve_system([de1, de2], [x,y])

[54]: [x(t) == (x(0) - 1)*cos(t) - (y(0) - 1)*sin(t) + 1,
       y(t) == (y(0) - 1)*cos(t) + (x(0) - 1)*sin(t) + 1]
```

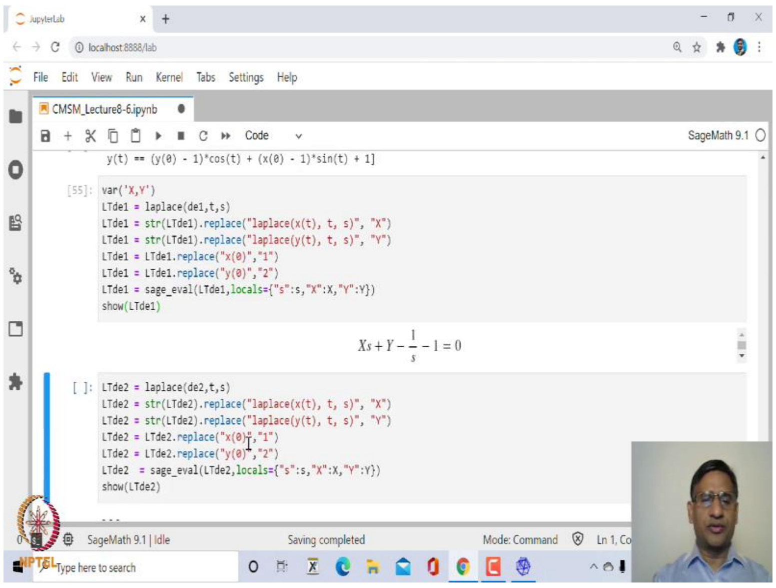
You can also solve system of 1st and 2nd order differential equations.

Here I have taken example of 1st order, but you can apply this to second order as well.

What do we do? First we will define these 2 differential equations in de1 and de2, and once you have defined this then you can solve this using inbuilt function desolve_system.

So, this is the solution using desolve_system. This we have we have seen as an application to eigenvalues eigenvectors.

(Refer Slide Time: 29:01)



```
y(t) == (y(0) - 1)*cos(t) + (x(0) - 1)*sin(t) + 1]
```

```
[55]: var('X,Y')
LTde1 = laplace(de1,t,s)
LTde1 = str(LTde1).replace("laplace(x(t), t, s)", "X")
LTde1 = str(LTde1).replace("laplace(y(t), t, s)", "Y")
LTde1 = LTde1.replace("x(0)","1")
LTde1 = LTde1.replace("y(0)","2")
LTde1 = sage_eval(LTde1,locals={"s":s,"X":X,"Y":Y})
show(LTde1)
```

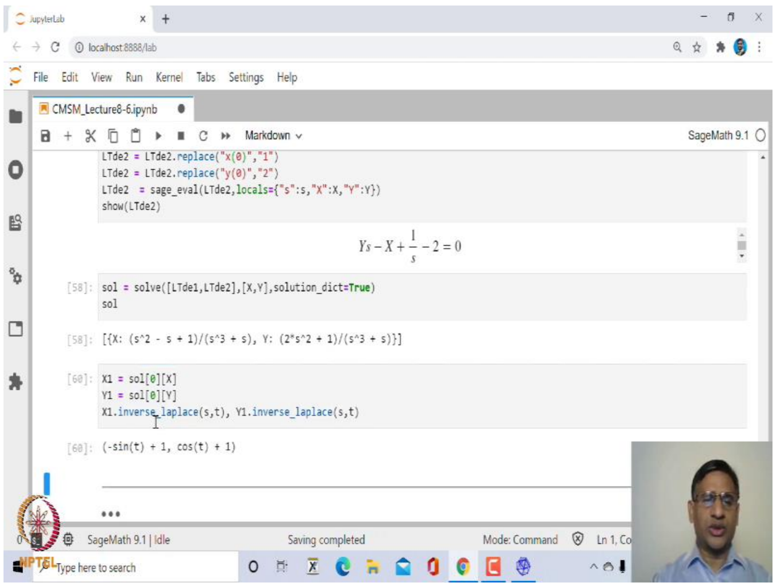
$$Xs + Y - \frac{1}{s} - 1 = 0$$

```
[ ]: LTde2 = laplace(de2,t,s)
LTde2 = str(LTde2).replace("laplace(x(t), t, s)", "X")
LTde2 = str(LTde2).replace("laplace(y(t), t, s)", "Y")
LTde2 = LTde2.replace("x(0)","1")
LTde2 = LTde2.replace("y(0)","2")
LTde2  = sage_eval(LTde2,locals={"s":s,"X":X,"Y":Y})
show(LTde2)
```

Now, how do we solve this using Laplace transform. All we need to do is, again apply this process that we did for each of this equation. Take the Laplace transform of the first equation then, replace this Laplace transform by capital X, then this will involve laplace transform of small y.

So, you replace capital laplace transform of y(t), by capital Y and then, replace this initial condition x(0) and y(0) by whatever the value you are given then evaluate this in as an expression. When you do that, the first differential equation gets transformed into capital X times s plus Y minus 1 upon s minus 1 is equal to 0. Similarly, you can do it with the second equation and that gets transformed into this.

(Refer Slide Time: 29:56)



```
LTde2 = LTde2.replace("x(0)","1")
LTde2 = LTde2.replace("y(0)","2")
LTde2  = sage_eval(LTde2,locals={"s":s,"X":X,"Y":Y})
show(LTde2)
```

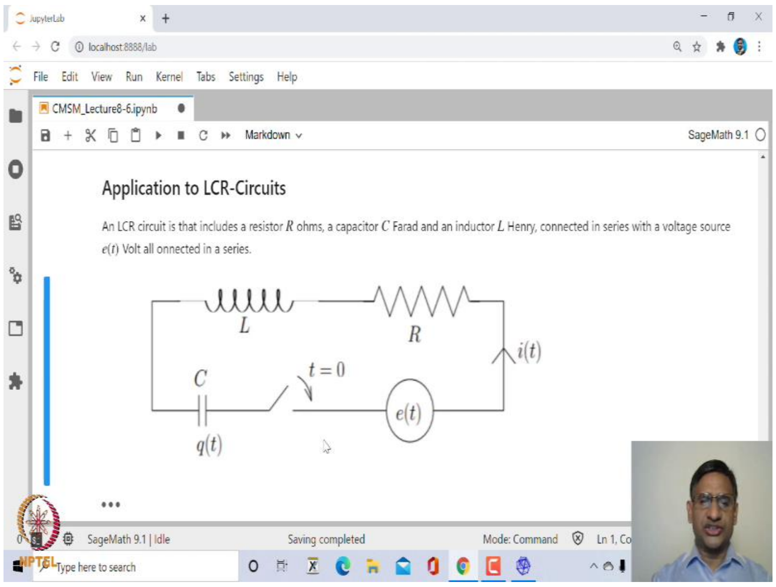
$$Ys - X + \frac{1}{s} - 2 = 0$$

```
[58]: sol = solve([LTde1,LTde2],[X,Y],solution_dict=True)
      sol

[58]: [{X: (s^2 - s + 1)/(s^3 + s), Y: (2*s^2 + 1)/(s^3 + s)}]

[60]: X1 = sol[0][X]
      Y1 = sol[0][Y]
      X1.inverse_laplace(s,t), Y1.inverse_laplace(s,t)

[60]: (-sin(t) + 1, cos(t) + 1)
```

So, now you have to solve these 2 equations simultaneously for capital X and capital Y, and they are linear equations. We can very easily solve this. Solve list of first equation second equation for capital X and Y and this is what we get. So, let us try to show what is this solution? This solution is, x is equal to this, y equal to this, and then we can find out what is capital X1 capital Y1. Extract this the X values and Y values from this solution.

Then you find inverse Laplace transform of this each of this X1 and Y1. You will get this as a solution. The first solution is 1 minus sin(t), first component x(t) is 1 minus sin(t), second component is 1+cos(t), that is very easy to check.

(Refer Slide Time: 30:57)



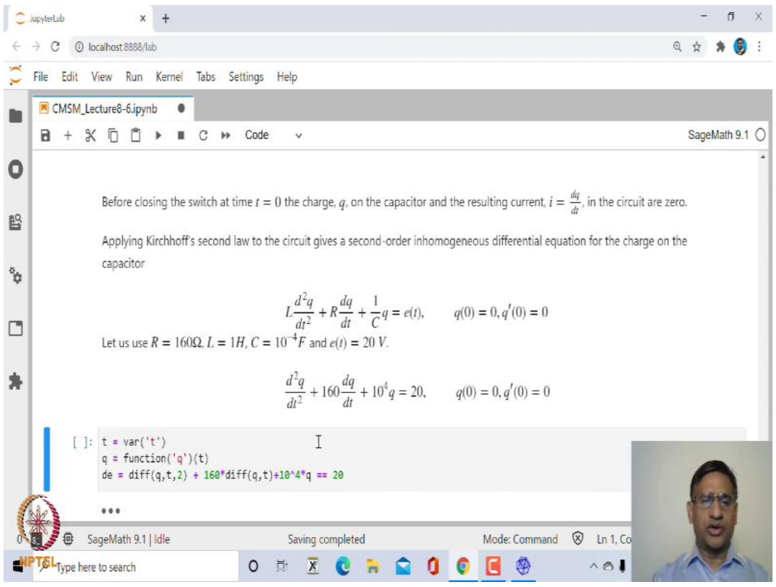
Application to LCR-Circuits

An LCR circuit is that includes a resistor R ohms, a capacitor C Farad and an inductor L Henry, connected in series with a voltage source $e(t)$ Volt all onnected in a series.

Now, let us just look at one more application to this into LR Circuit, actually this Laplace transform etcetera is used quite heavily in this circuit problems.

So, let us take what is called LCR circuit, that includes a resistor R in ohms, capacitor C in Farad and an inductor L in Henry, and it is connected in a series with voltage source, let us say e(t) volt. This is the diagram here, L, R, i(t), this is the current and this is e(t) and this is q(t).

(Refer Slide Time: 31:41)



Before closing the switch at time $t = 0$ the charge, q, on the capacitor and the resulting current, $i = \frac{dq}{dt}$, in the circuit are zero.

Applying Kirchhoff's second law to the circuit gives a second-order inhomogeneous differential equation for the charge on the capacitor

$$L\frac{d^2q}{dt^2} + R\frac{dq}{dt} + \frac{1}{C}q = e(t), \qquad q(0) = 0, q'(0) = 0$$

Let us use $R = 160\Omega$, $L = 1H$, $C = 10^{-4}F$ and $e(t) = 20\ V$.

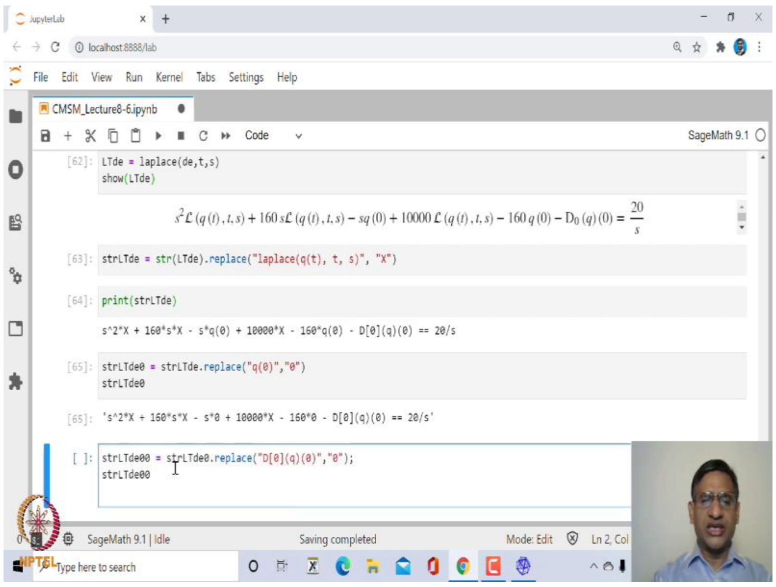
$$\frac{d^2q}{dt^2} + 160\frac{dq}{dt} + 10^4 q = 20, \qquad q(0) = 0, q'(0) = 0$$

```
[ ]: t = var('t')
     q = function('q')(t)
     de = diff(q,t,2) + 160*diff(q,t)+10^4*q == 20
```

What it means is, before closing at time t, the charge q on this capacitor and resulting current here is dq by dt in this circuit is 0. So, and then you can use Kirchhoff second law in this case and then what you get is this particular this particular circuit can be represented by this second order differential equation with initial condition q(0) is 0, q'(0) is 0. Let us take R is equal to 160 ohm, L is equal to 1 Henry, C is equal to 10 to the power minus 4 Faraday and e(t), that is 20 volt.

So, for this particular thing, if you substitute in this particular equation, what you get is this second order differential equation with initial conditions.

(Refer Slide Time: 32:36)



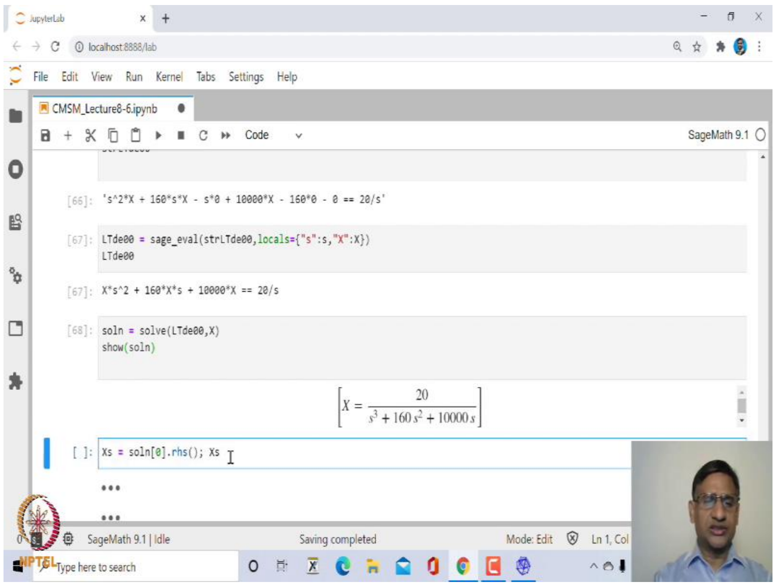
```
[62]: LTde = laplace(de,t,s)
      show(LTde)
```

$$s^2\mathcal{L}\left(q\left(t\right),t,s\right)+160\,s\mathcal{L}\left(q\left(t\right),t,s\right)-sq\left(0\right)+10000\,\mathcal{L}\left(q\left(t\right),t,s\right)-160\,q\left(0\right)-\mathrm{D}_0\left(q\right)\left(0\right)=\frac{20}{s}$$

```
[63]: strLTde = str(LTde).replace("laplace(q(t), t, s)", "X")

[64]: print(strLTde)

      s^2*X + 160*s*X - s*q(0) + 10000*X - 160*q(0) - D[0](q)(0) == 20/s

[65]: strLTde0 = strLTde.replace("q(0)","0")
      strLTde0

[65]: 's^2*X + 160*s*X - s*0 + 10000*X - 160*0 - D[0](q)(0) == 20/s'

[ ]: strLTde00 = strLTde0.replace("D[0](q)(0)","0");
     strLTde00
```

Now, we in order to solve this, first we will just define this differential equation, apply Laplace transform to this differential equation. This is what we get, and then we substitute Laplace transform of q by capital X, we could have substituted by capital Q, it would have been better, but it does not matter anyway it is a variable.

(Refer Slide Time: 33:07)



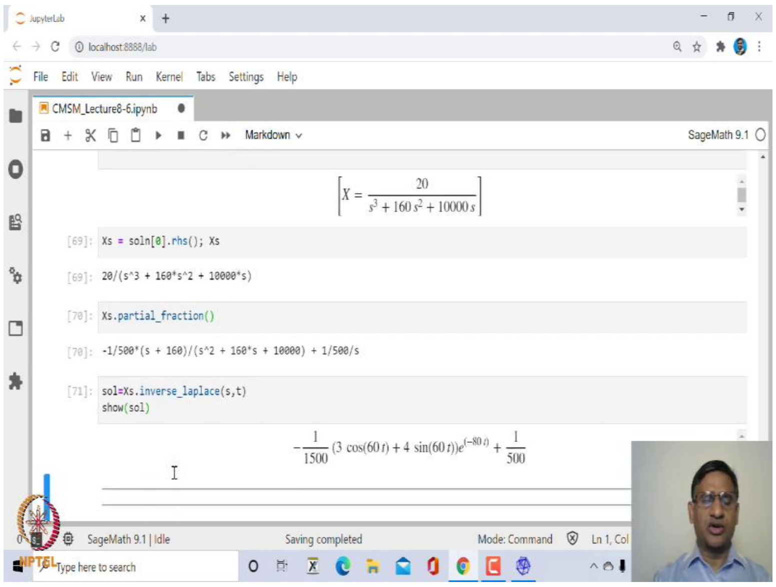
```
[66]: 's^2*X + 160*s*X - s*0 + 10000*X - 160*0 - 0 == 20/s'

[67]: LTde00 = sage_eval(strLTde00,locals={"s":s,"X":X})
      LTde00

[67]: X*s^2 + 160*X*s + 10000*X == 20/s

[68]: soln = solve(LTde00,X)
      show(soln)
```

$$\left[X = \frac{20}{s^3 + 160\,s^2 + 10000\,s}\right]$$

```
[ ]: Xs = soln[0].rhs(); Xs
```

Then let us print this string, which is laplace transform of this differential equation. And then, replace q(0) by 0 and similarly replace q’(0) by 0 and then, evaluate this, small s by small s, string replace it by expression s and capital X by expression capital X. So, this is what we get. Now, we have to solve this, for let us solve this for capital X. This is what you get. That is the right hand side and when we take inverse Laplace transform of the right hand side we will get the solution.

(Refer Slide Time: 33:40)



$$\left[X = \frac{20}{s^3 + 160\,s^2 + 10000\,s}\right]$$

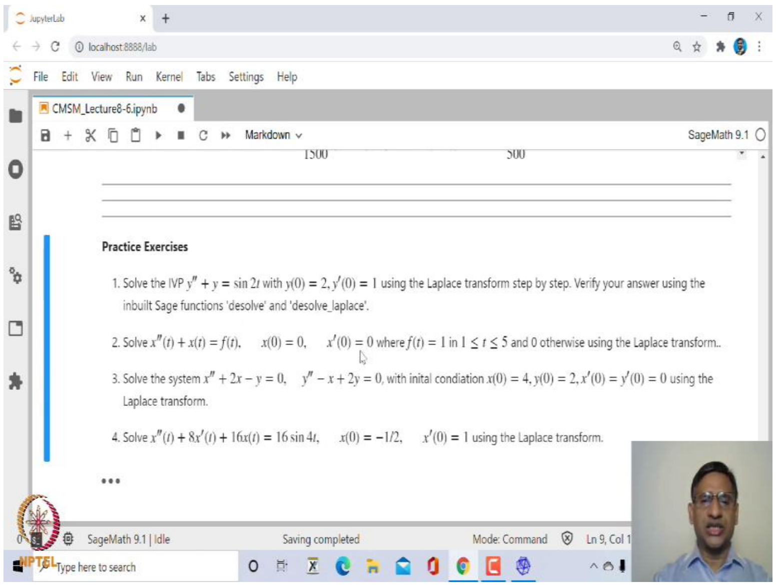
```
[69]: Xs = soln[0].rhs(); Xs

[69]: 20/(s^3 + 160*s^2 + 10000*s)

[70]: Xs.partial_fraction()

[70]: -1/500*(s + 160)/(s^2 + 160*s + 10000) + 1/500/s

[71]: sol=Xs.inverse_laplace(s,t)
      show(sol)
```

$$-\frac{1}{1500}\left(3\,\cos(60\,t) + 4\,\sin(60\,t)\right)e^{(-80\,t)} + \frac{1}{500}$$

So, let us store this into capital Xs and find find the partial fraction which is not necessary you can take the inverse Laplace transform of this. This is what you get as a solution. So, this is a solution of this differential equation which is associated to this particular LCR circuit.

So, this is one application, but you can explore, there are many applications of Laplace transform in physics and in engineering. What I gave you is just one such, but there are many.

(Refer Slide Time: 34:20)



Practice Exercises

1. Solve the IVP $y'' + y = \sin 2t$ with $y(0) = 2, y'(0) = 1$ using the Laplace transform step by step. Verify your answer using the inbuilt Sage functions 'desolve' and 'desolve_laplace'.
2. Solve $x''(t) + x(t) = f(t), \quad x(0) = 0, \quad x'(0) = 0$ where $f(t) = 1$ in $1 \leq t \leq 5$ and 0 otherwise using the Laplace transform..
3. Solve the system $x'' + 2x - y = 0, \quad y'' - x + 2y = 0$, with inital condiation $x(0) = 4, y(0) = 2, x'(0) = y'(0) = 0$ using the Laplace transform.
4. Solve $x''(t) + 8x'(t) + 16x(t) = 16 \sin 4t, \quad x(0) = -1/2, \quad x'(0) = 1$ using the Laplace transform.

Now, let me leave you with a few practice exercises. These are the simple exercises of solving differential equations, the initial value problems using Laplace transform.

I expect all of you to solve this, 1st solve this step by step and verify this using inbuilt function desolve and desolve_laplace.

So, these are very simple except that the 2nd one is here right hand side is f(t), f(t), is actually a unit step function, which is equal to 1 when t lies between 0 and 5 1 and 5 and 0 otherwise.

We saw that such a function, we can write in terms of unit step function.

So, let me stop here actually this was the last lecture in this the course. In next lecture, I will tell you briefly after going through this course how you can explore SageMath further.