

# Lecture 11

Solving ordinary differential equations [ODEs] with the **Fourier** series.

Let's first review what we already know from previous lectures. Ordinary differential equations are equations that state the relation between a function  $y(t)$  and its derivatives  $y'(t), y''(t), \dots$ . The function depends only on one independent variable  $t$ . An example of a second-order equation would be

$$\frac{d^2}{dt^2}y(t) + 4y(t) = f(t)$$

The right hand side  $f(t)$  is assumed to be a known function given by the problem we are solving. Sometimes it is called the forcing term and represents an outside influence to the system, like for example the source voltage applied to drive a circuit. If it is zero the ODE is called homogenous, otherwise inhomogenous.

The goal is to use the differential equation to find the function  $y(t)$ , given the driving function and an initial state, for example  $y(t_0)$  at time  $t_0$ . Since its a second order system we also need to specify  $y'(t_0)$

The solution of the example above in the homogenous case  $f(t) = 0$  can be found by looking for solutions for  $y(t) = e^{at}$  which gives the roots

$$a^2 + 4 = 0, \quad a = \pm 2i$$

thus we are looking for a solution of the form  $y(t) = c_1e^{2it} + c_2e^{-2it}$ . Lets simplify further and require

$$y(0) = 1$$
$$y'(0) = 0$$

Thus we get

$$c_1 + c_2 = 1 \text{ and } 2ic_1 - 2ic_2 = 0 \text{ yielding } c_1 = c_2 = \frac{1}{2} \text{ so that we get}$$

$$y(T) = \frac{e^{2it} + e^{-2it}}{2} = \cos(2t)$$

Using Matlabs symbolic calculator we can confirm

```
syms y(T) T0 Y0
dY = diff(y,T);
ode = diff(diff(y,T),T)+4*y == 0;
cond = [y(T0) == 1 dY(T0) == 0]
```

cond =

$$\left( y(T_0) = 1 \quad \frac{\partial}{\partial T_0} y(T_0) = 0 \right)$$

```
subs(dsolve(ode,cond),T0,0)
```

$$\text{ans} = \cos(2T)$$

In Lecture 6 we introduced the **Laplace** transform  $L\{y(t)\} = Y(s)$  as another tool to solve ODEs. It is defined as the integral transformation

$$Y(s) = \int_0^{\infty} e^{-st}y(t)dt$$

Applying the Laplace transform had the effect of projecting the original differential ODE into simple algebraic equations for the transformed function  $Y(s)$  in "s" space. After finding the solutions for  $Y(s)$  applying the **inverse Laplace** transformation yields the solution for  $y(t)$ .

Laplace transforming the simple homogenous example above yields

$$L\left\{\frac{d}{dt}y(t)\right\} + 4L\{y(t)\} = 0$$

using the derivative property of the Laplace transform

$$s^2L\{y(t)\} - sY(0) - L\{y'(0)\} + 4L\{y(t)\} = 0$$

$$s^2Y(s) - sY(0) + 4Y(s) = 0$$

$$Y(s) = \frac{s}{s^2 + 4} Y(0)$$

Applying the **inverse Laplace** transform

$$L^{-1}\{Y(s)\} = y(t) = Y(0) L^{-1}\left\{\frac{1}{s+4}\right\} = Y(0)\cos(2t)$$

where we used the particular inverse Laplace transform for the  $1/s$  term. Thus the problem of solving the ODEs simplified to algebraic equations but now relies on being able to perform the inverse Laplace transform.

Using Matlab to perform both transformations steps yields

```
syms s t y(t)
laplace(diff(diff(y,t))+4*y(t),t,s)
```

ans =

$$s^2 \text{laplace}(y(t), t, s) - s y(0) - \left(\left(\frac{\partial}{\partial t} y(t)\right)\Big|_{t=0}\right) + 4 \text{laplace}(y(t), t, s)$$

```
ilaplace(s/(s^2+4),s,t)
```

ans =  $\cos(2t)$

In this lecture we will use the **Fourier series** to transform the ODE into frequency space. Just like in the Laplace transform case the ODE translates into an algebraic equation for the coefficients.

We introduced the Fourier series in the context of decomposing functions by their frequency spectrum using the Fast Fourier transform algorithm in Lecture 8. In the Fourier series we represent that function as a series of periodic trigonometric functions

$$y(t) = \frac{a_0}{2} + \sum_{n=1 \dots \infty} a_n \cos(nt) + b_n \sin(nt)$$

with

$$a_n = \frac{1}{\pi} \int_{-\pi}^{\pi} y(t) \cos(nt) dt$$

$$b_n = \frac{1}{\pi} \int_{-\pi}^{\pi} y(t) \sin(nt) dt$$

Taking derivatives we find

$$y'(t) = \sum_{n=1 \dots \infty} b_n n \cos(nt) - a_n n \sin(nt)$$

$$y''(t) = \sum_{n=1 \dots \infty} -a_n n^2 \cos(nt) - b_n n^2 \sin(nt)$$

The coefficients  $a_n$  and  $b_n$  for frequencies  $n$  play the role of the roots of the first method above or the function  $Y(s)$  in the Laplace case. In the Laplace case the integration domain was set to a specific starting point at  $t_0$  into the future. This is most appropriate if we consider external driving functions of this form. For example like the heavyside function stepping up from 0 to 1 at time 0. In the Fourier case the trigonometric basis functions are defined for all times and the integration domain includes both past and future times.

Solving a ODE by means of the Fourier series requires substituting the derivatives with the expansion above and matching like terms for the trigonometric functions. For the simple ODE example above

$$\frac{d^2}{dt^2} y(t) + 4 y(t) = 0$$

we get by substituting the Fourier series

$$\sum_{n=1 \dots \infty} -a_n n^2 \cos(nt) - b_n n^2 \sin(nt) + 4 \frac{a_0}{2} + 4 \sum_{n=1 \dots \infty} a_n \cos(nt) + b_n \sin(nt) = 0$$

and grouping frequency terms

$$4 \frac{a_0}{2} + \sum_{n=1 \dots \infty} a_n (4 - n^2) \cos(nt) + b_n (4 - n^2) \sin(nt) = 0$$

Since the right hand side is zero all independent frequency terms must be individually zero as well.

$$2a_0 = 0$$

$$a_n (4 - n^2) = 0$$

$$b_n(4 - n^2) = 0$$

This implies that that only the coefficients  $a_2$  and  $b_2$  are non zero

$$y(t) = a_2 \cos(2t) + b_2 \sin(2t)$$

For initial conditions  $t=0$  we get require  $y(0) = 1$  and  $y'(0) = 0$  getting the final answer

$$y(t) = \cos(2t)$$

## Fourier Series applied to Inhomogenous ODEs

We not consider the case of an inhomogeneous differential equation with the right hand side  $f(t) \neq 0$ . The type of driving function is critical if one chooses to apply either a Laplace or Fourier transform technique, as a large part of the work will applying the transformation to it. If  $f(t)$  is already a periodic or even a sinusoidal function the Fourier transform will be easier. A typical example is applying the technique if the driving function is a square wave. Consider

$$\frac{d^2}{dt^2} y(t) + 4 y(t) = f(t)$$

with

$$f(t) = -1 \text{ for } -\pi < t < 0$$

$$f(t) = +1 \text{ for } 0 < t < \pi$$

In order to transform the ODE into frequency space, we need to transcribe  $f(t)$  in terms of

$$f(t) = \frac{a_0}{2} + \sum_{n=1 \dots \infty} a_n \cos(nt) + b_n \sin(nt)$$

using

$$a_0 = \frac{1}{\pi} \int_{-\pi}^{\pi} f(t) dt$$

$$a_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(t) \cos(nt) dt$$

$$b_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(t) \sin(nt) dt$$

Since the particular choice of  $f(t)$  is a odd function in  $t$  and the  $\cos$  function is even, all  $a$  coefficients are zero!

For the  $\sin$  functions we get

$$b_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(t) \sin(nt) dt = \frac{2}{\pi} \int_0^{\pi} 1 \sin(nt) dt = \frac{-2}{n\pi} (\cos(n\pi) - \cos(0)) = \frac{-2}{n\pi} (\cos(n\pi) - 1)$$

$$b_n = \frac{4}{n\pi} \text{ for } n = 1, 3, 5 \dots \text{odd}$$

$$b_n = 0 \text{ for } n = 2, 4, 6 \dots \text{even}$$

$$b_n = \frac{2}{n\pi} (1 - (-1)^n)$$

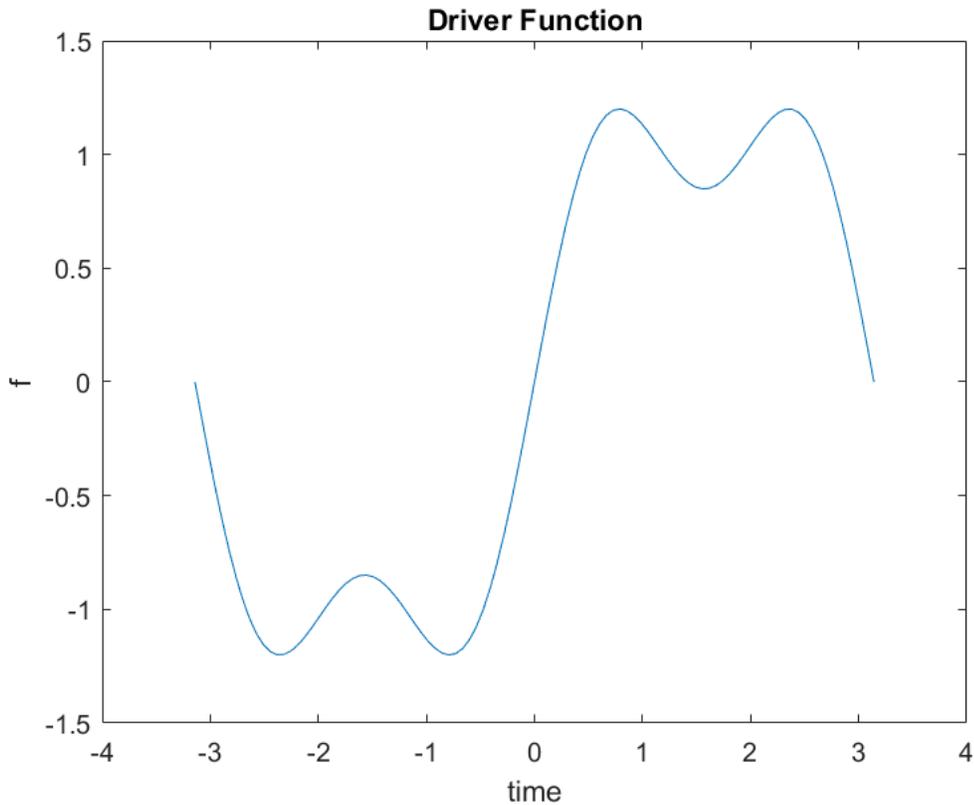
## Using Matlab

```
syms n t
assume(n,{'positive','integer'})
int(2*sin(n*t),0,pi)/pi
```

ans =

$$\frac{4 \sin\left(\frac{\pi n}{2}\right)^2}{n \pi}$$

```
bn = @(n) (2/(pi*n)*(1-(-1)^n));
n_array = 1:2:3;
sin_vec = @(t) ( arrayfun(@sin,t.*n_array) );
bn_vec = arrayfun(bn,n_array);
y = @(t) ((bn_vec*sin_vec(t)') );
tarray = linspace(-pi,pi,100);
yarray = arrayfun(y,tarray);
plot(tarray,yarray)
title('Driver Function')
xlabel('time')
ylabel('f')
```



Considering the full ODE from above, the grouping of coefficient term becomes

$$4\frac{a_0}{2} + \sum_{n=1, \dots, \infty} a_n(4 - n^2)\cos(nt) + b_n(4 - n^2)\sin(nt) = \frac{2}{n\pi}(1 - (-1)^n)\sin(nt)$$

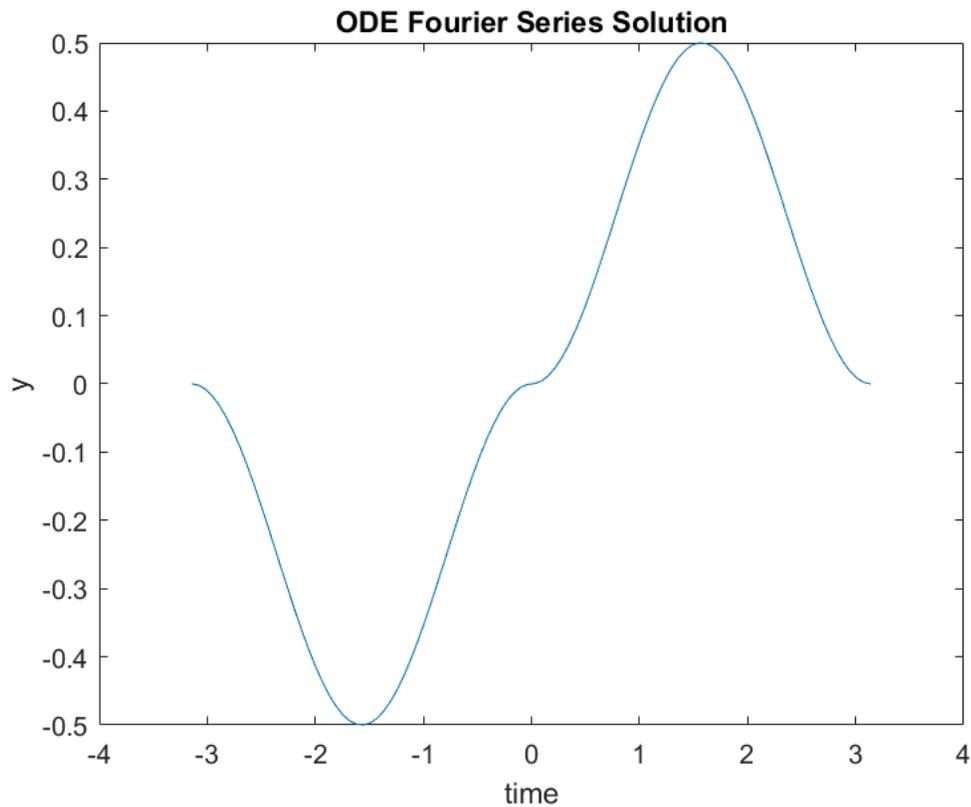
This times only the sin terms survive

$$b_n = \frac{2}{n\pi}(1 - (-1)^n)\frac{1}{4 - n^2}$$

So we get for the inhomogenous solution of the ODE

$$y(t) = \sum_{n=1, \dots, \infty} \frac{2}{n\pi(4 - n^2)}(1 - (-1)^n)\sin(nt)$$

```
bn = @(n) (2/(pi*n*(4-n^2))*(1-(-1)^n));
n_array = 1:2:101;
sin_vec = @(t) ( arrayfun(@sin,t.*n_array) );
bn_vec = arrayfun(bn,n_array);
y = @(t) ((bn_vec*sin_vec(t)') );
tarray = linspace(-pi,pi,100);
yarray = arrayfun(y,tarray);
plot(tarray,yarray)
title('ODE Fourier Series Solution')
xlabel('time')
ylabel('y')
```

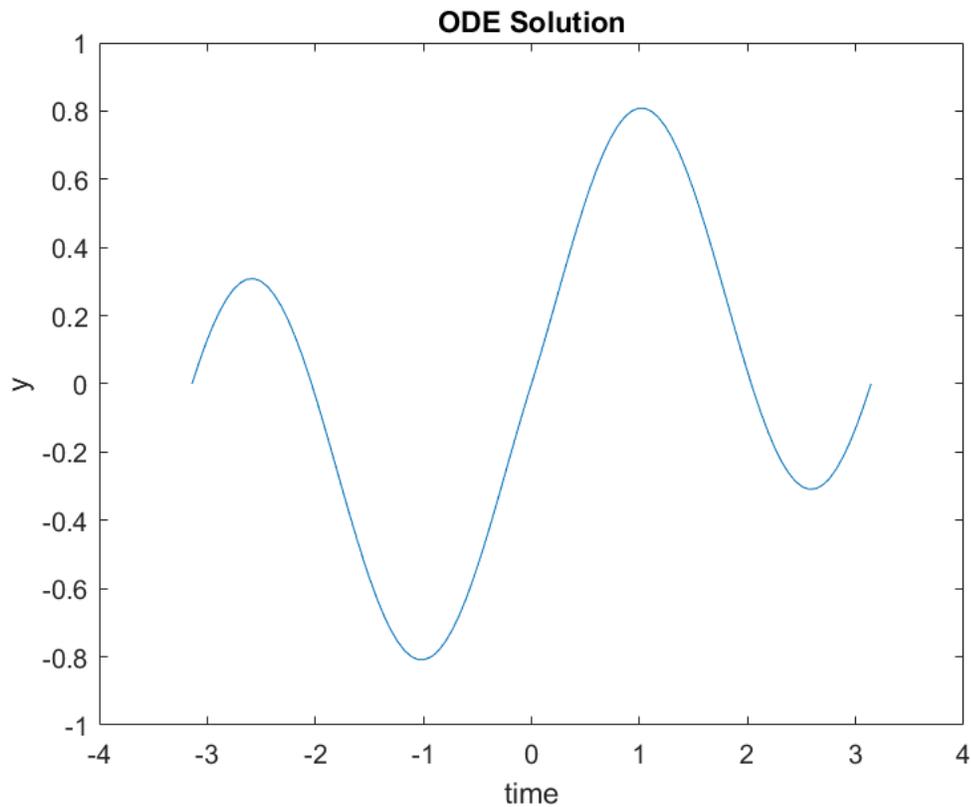


Comparing the Fourier series solution to using the symbolic solver DSOLVE

```

syms y(t) t
dy = diff(y,t);
ddy = diff(dy,t);
f(t) = heaviside(t)-heaviside(-t);
cond =[y(0)==0 dy(0)==1];
ysol(t) = dsolve(ddy+4*y==f,cond);
tarray = linspace(-pi,pi,100);
yarray = arrayfun(@(t)(double(ysol(t))),tarray);
plot(tarray,yarray)
title('ODE Solution')
xlabel('time')
ylabel('y')

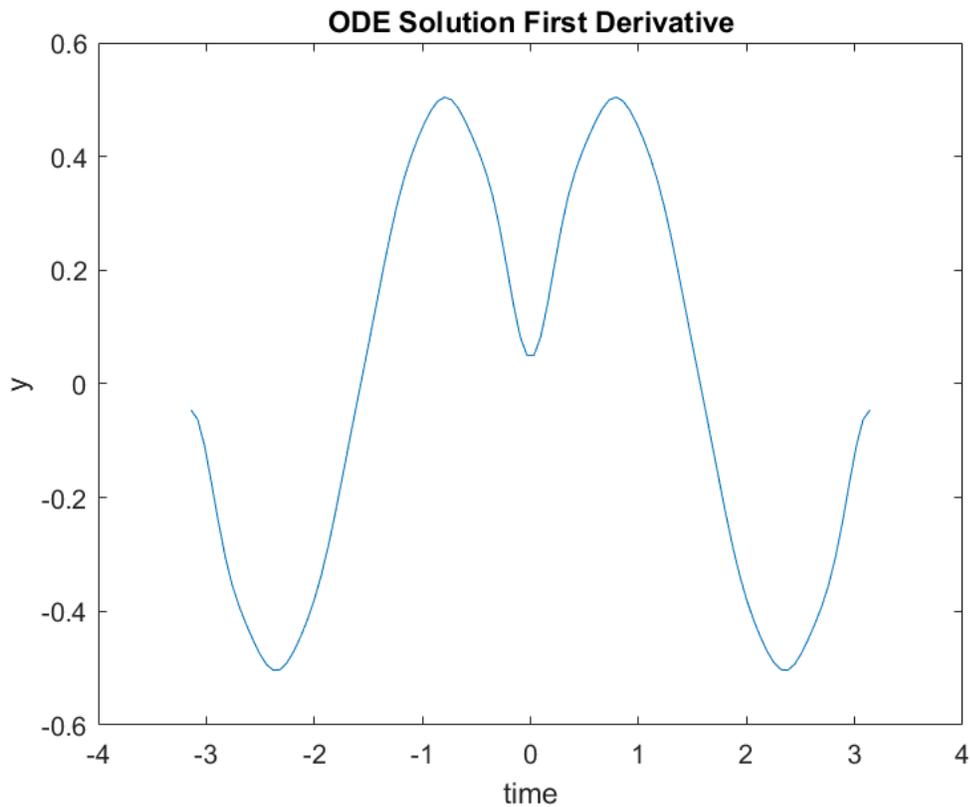
```



## TASK

1. Copy and modify the code that plots the value of the Fourier ODE solution to calculate the first derivative  $dy/dt$  of  $y$  instead of the function value. Hint, taking the derivative requires only a small modification to the terms of the Fourier series.

```
% bn =      ENTER YOUR CODE HERE
% n_array = 1:2:13;
% sin_vec = ENTER YOUR CODE HERE
% bn_vec = arrayfun(bn,n_array);
% y = @(t) ((bn_vec*sin_vec(t)') );
% tarray = linspace(-pi,pi,100);
% yarray = arrayfun(y,tarray);
% plot(tarray,yarray)
% title('ODE Solution First Derivative')
% xlabel('time')
% ylabel('y')
```



## TASK

2. With the slope calculated above, record its value for time  $t=0$ . Use this value as initial condition for the DSOLVE ODE solution. Copy the modified code to plot the DSOLVE solution below, do you see a better match?

## TASK

3. Use the integration-by-part technique to derive the integrals for  $x \cdot \sin(x)$ . Use the equation editor (INSERT/Equation) to type up your calculation steps. Compare your result to doing the symbolic derivation with Matlab using the "int" function.

## TASK

4. Solve the second order ODE from above using  $f(t) = t$  as driver function using the Fourier series technique. For this you need to recalculate the Fourier coefficients. Plot the answer. Then make another plot comparing your solution with a solution using Matlabs DSOLVE function.