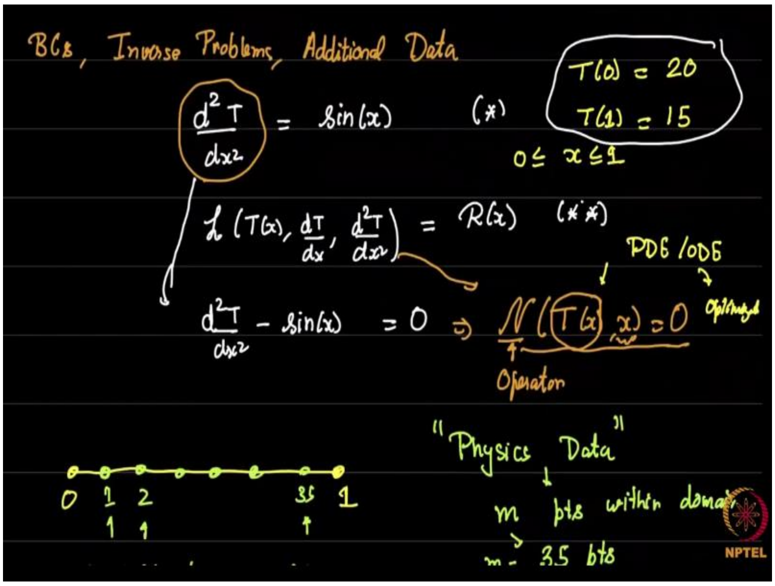
Inverse Methods in Heat Transfer
Prof. Balaji Srinivasan
Department of Mechanical Engineering
Indian Institute of Technology – Madras

Lecture - 59
Physics Informed Neural Networks – BC Incorporation

(Refer Slide Time: 00:19)



We are in week eleven of inverse methods in heat transfer. in the last couple of videos, we discussed a basic introduction to physics informed neural networks also known as PINNs. I had discussed various aspects of this particularly I had given you a sort of semi-intuitive explanation in the previous video. but we did leave out a few things I did not talk about how we apply specific boundary conditions nor did we discuss how we solve inverse problems using PINNs nor did we really talk about how we can incorporate additional data. so can we do with a mix of experimental data as well as existing neural existing physics information can we combine all these and we will look at these three aspects within this video.

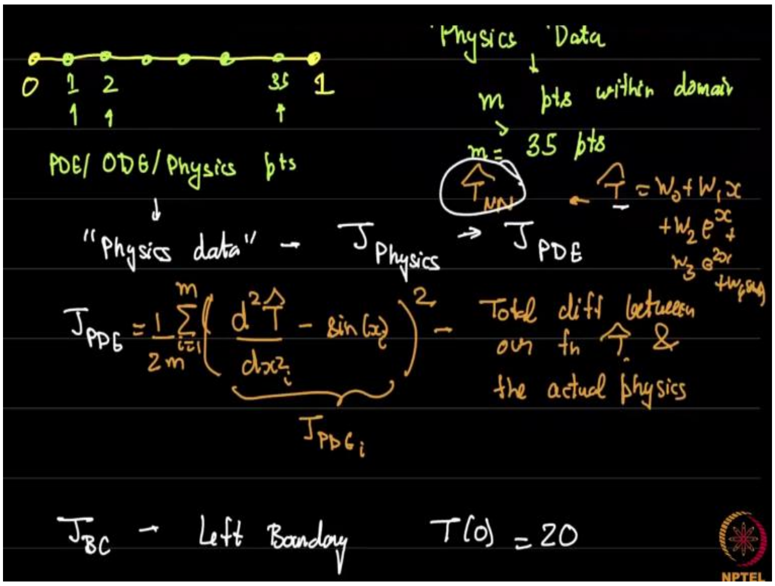
So, what we saw in the last couple of videos was given a PDE or an ODE, so let us take we have an ode of the form $\frac{d^2T}{dx^2}$ equal to let me just make up something sin (x). So, let us say this is an example ode I am going to call this to be of the form L(x) = R(x), so left hand side of x is some right-hand side of x now more precisely L(x) is an operator which is acting on T and on the right-hand side you have a function which could be a function of T or it could be a function of x. If you want to be even more precise the reason, I am doing it in three steps is when we write it this way it can get confusing so you could have a function of T, $\frac{dT}{dx}, \frac{d^2T}{dx^2}$ and you have something on the right-hand side let us call this some R(x) now all put together.

You basically do not write it in the form some left hand side equal to some right-hand side. but you transform this to the form $\frac{d^2T}{dx^2}$ just like we did with newtons method something equal to zero okay. So, we will write this as $\frac{d^2T}{dx^2}$ - $\sin(x) = 0$ and more precisely from here and the equivalent form from here would be something like $\aleph(T(x), x) = 0$, what this N means ultimately, we want to talk about Navier Stokes equations or some thermal equation. but this is some operator an operator means there could be a derivative or the second order of the derivative like here which acts on t and you have some terms that depend on x and at the end this is equal to zero.

So, this should remind you of What we did with Newton Raphson also okay. now the question is how do you transfer this equation which is simply a PDE or an ODE into an optimization problem okay? now obviously we already looked at a little bit of this in the last couple of videos but before we move further we know that in addition to this we have some boundary conditions so the boundary conditions could be let us just say T(0)=0 and T(1)=15 and let me change this also to some more identifiable value let us say it is 20 and right hand side is 15 and x belongs to the domain zero one okay so this could be a slab could be a temperature problem or t could be any other variable I am just making up some ODE or PDE okay.

Now what we know is all We do is we take this domain zero to one and decide on getting some physics data okay now just imagine a parallel problem where you are doing some housing price prediction or something of that sort if we are doing some prediction of some engineering problem or we are trying to predict some stock price, we would sample it at a few points. similarly, here we are sampling the physics at few points let us say we are sampling it at m points within the domain okay and these m points could be let us say just for the sake of argument let me just give some value let us say thirty-five points just so that we have some unique number here so m equal to thirty-five points I am sampling one two three four etcetera up till thirty-five.

(Refer Slide Time: 05:23)

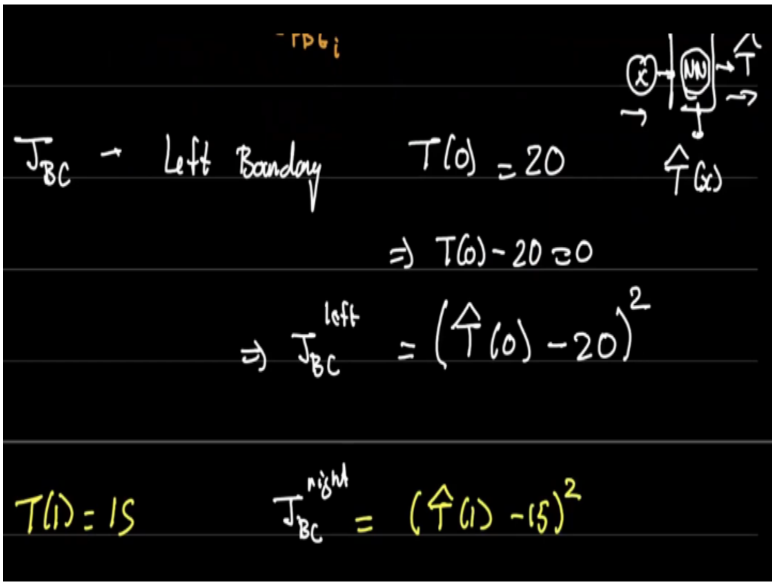


Now add these thirty-five these we will call the ode or the physics points or if it is a PDE we will call it the PDE box okay so these points serve only one purpose. We are going to take the physics data from here and that is going to go into the loss function as J physics or typically in the literature we call it J_{PDE} or J_{ODE} or J differential equation so let us just call it J_{PDE} here so just I am going to call it J_{PDE} although this is actually an ordinary differential equation so what is J_{PDE}? J_{PDE} is whatever you predicted so here obviously just like we did before we need a hypothesis function and the hypothesis function is $\hat{T}$ some function that we decide on minus $\sin(x)$ at these points whichever points we have these squared and summed up and if you wish we can have a one by two factor or not and one by two the number of points the summation is from i equal to one to the number of points.

This denotes the total difference or square difference between our function $\hat{T}$ and the actual physics of the problem so actual physics of the problem says this should be exactly zero whereas our $\hat{T}$ suppose I make up a function $\hat{T}$ is some function of w_0 and w_1 . let us say $w_0 + w_1 x$ if I plug that in here then I will get $\frac{d^2T}{dx^2} = 0$ because it is a linear function that will get zero and that will not match this so you want to say that okay w zero plus w one x is not a great function now you might add something else let us say $w_2 e^x + w_3 e^{2x} + w_4 \sin 2x + \ldots$ so you might if may add all these other hypothesis functions

We plug all the all of those in here and just compare What the $\frac{d^2T}{dx^2}$ does versus what it should do which is equal to signs. So, once we do that, we actually have our hypothesis function and we have our J_{PDE} individually at this i okay. now of course instead of $\hat{T}$ being this $\hat{T}$ could also be a neural network. So, it could be a neural network and that would also give some hypothesis function which depends on w so it really does not matter what we use here okay. this is all fine but there is no way that any t we choose like that will satisfy these two conditions.

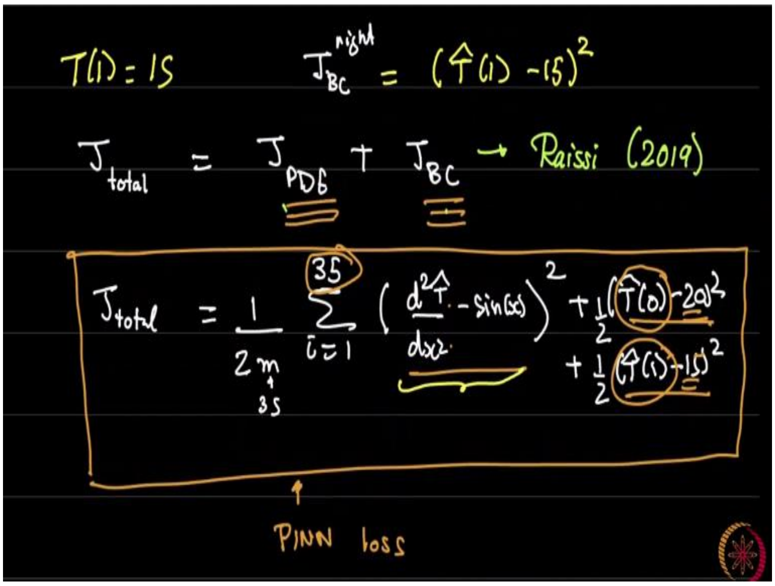
(Refer Slide Time: 08:28)



So, just like we had a J_{PDE} we should have a J boundary condition and if it is an initial value problem, we should have a J initial condition I will show you how to do a J boundary condition right now. So, J boundary condition is going to be let us first take the left boundary. So, the left boundary says that $\hat{T}$ should be I took some value twenty. so of course what you want is to say that $\hat{T}(0)$-20 $= 0$ so what we do is J_{BC} from the left will be you actually find out what $\hat{T}(0)$ is so remember you have a hypothesis function which is a neural network this neural network would take in x, let me call this NN and it will give out a T. so neural network simply is a substitute for what this function T is okay so instead of saying T(x) I can say neural network the value at x equal to zero it will give a value of course I give an input will give a value out it should be twenty.

But I cannot match this exactly you know that neural networks will not in general match this exactly so we will add a loss which is $(\hat{T}(0)\text{-}20)^2$ okay now similarly you can have a J_{BC} which is at the right and our right boundary condition was that t of one equal to i think it was fifteen yes $T(1) = 15$. So, I will simply say that J_{BC} at the right is $(\hat{T}(1)\text{-}15)^2$, how does this help?

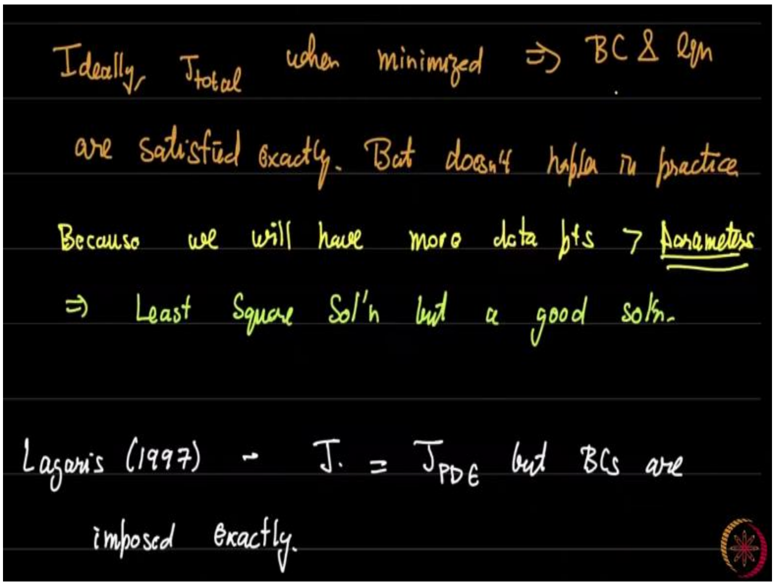
(Refer Slide Time: 10:30)



We will basically say a simple thing J_{total} or J_{PINN} is J_{PDE} plus J at the boundary condition so which comes to if you see one by two 1 by times however the number of points were how many ever were the number of points so let us say there are thirty five points so then m will be thirty five then we will simply sum up d square T-hat dx square minus sine x the whole square plus t of T-hat of zero minus twenty square plus one by two you have a half everywhere T-hat at one minus fifteen square.

So, this then some people put an additional factor of one by two for just the left and right boundary conditions this total loss is the PINN loss okay so the PINN loss is made up of the PDE loss and the boundary condition what does this do so when you try to minimize the PINN loss it will be ideally minimized then this is zero this is zero and this is 0 zero and the ideal minimization will be obviously d square t dx square equal to sine x that is thirty five 35 points and t at the left is exactly twenty and t at the right is exactly fifteen.

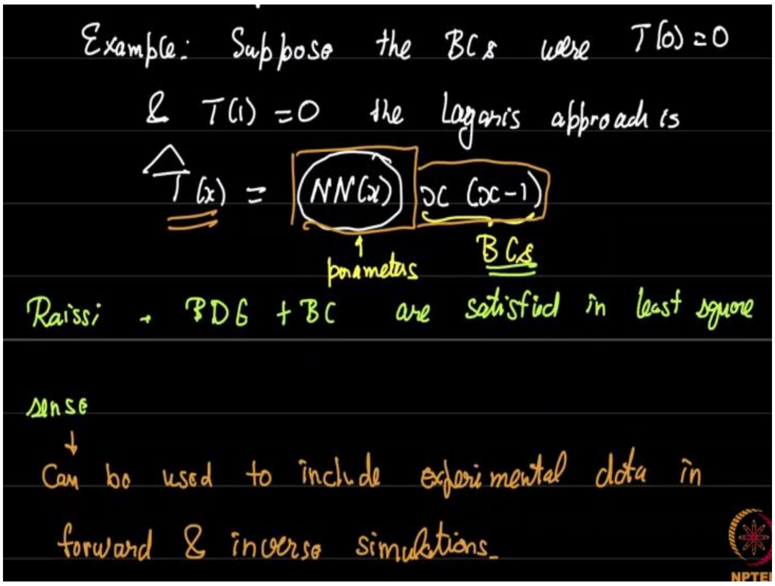
(Refer Slide Time: 12:10)



Unfortunately, obviously in practice ideally J total When minimized, implies that BC and equation are satisfied exactly. but this minimum but does not happen in practice okay so typically you will have more problems than this you will not be able to satisfy these exactly. that is because you will have more equations than unknowns, because we will have typically more data points compared to the number of parameters. So, this is an important thing to consider while doing PINNs this does not get often discussed. the number of parameters and PINNs in the neural network that you use should be fewer than the number of points where we apply these conditions. just so that it is effectively a least square solution that occurs. but even though this does not occur we tend to get a least square solution but a good solution.

Now this idea of adding J_{PDE} plus J_{BC} was a particular contribution of the Raissi and Karniadakis twenty nineteen paper I will shortly show it to you. there was another idea which was by Lagaris this is the original idea from nineteen ninety-seven. this idea was that J is just equal to J_{PDE} but boundary conditions are imposed exactly. now how do we do that. there are various ways of doing it let me give you a simple example.

(Refer Slide Time: 14:36)



The example would be something like you satisfy let me just show you suppose I am just giving you a simple example suppose the BCs were t of zero is zero and t of one equal to zero then the Lagaris approach is T-hat equal to neural network of x multiplied by x into x minus one now how does this help?

This helps in making the solution arbitrary so that you have number of parameters so all the parameters are here this satisfies the boundary conditions how so when you put t of one you get x minus one so this will be zero when you put t of this is x equal to zero then still it will be zero. everywhere else it does not have to be zero and you will basically satisfy the parameters. So, you basically manipulate the expression for the neural network so that the boundary conditions are satisfied exactly.

On the other hand, in the Raissi approach both PDE which is the physics as well as the boundary conditions are satisfied in the least square sense the Raissi approach has a few disadvantages of course the boundary condition is not satisfied exactly and you need to play with certain parameters.

I am not discussing that here maybe I will discuss it next week if time permits but it has the major advantage that can be used to include experimental data in forward simulations, turns out the same thing also helps in inverse simulations. now I will show how this is possible in a short while but for now I just would like you to remember the following things the temperature or any variable we have is basically represented by a neural network in the Lagaris approach we add some extra terms just so that the boundary condition is satisfied otherwise we have very simple and very beautiful form which is there in the Raissi and Karniadakis form which is you just add the PDE loss to the boundary condition loss and it just works in Practical problems.

So, what I would like you to what I would like you to see now is some excerpts out of these two seminal papers. I will show you some of these results from the papers and then we will come back and discuss some extra important aspects of these. So, I will show you these in the next video.