Physics-informed neural networks

What is a PINN

- NN constraints by physical laws
  Instead of fitting the input-output relationship **empirically**, the PINN learning is **guided by physics.**

- Utilize the NN’s representation power (a universal function approximator)

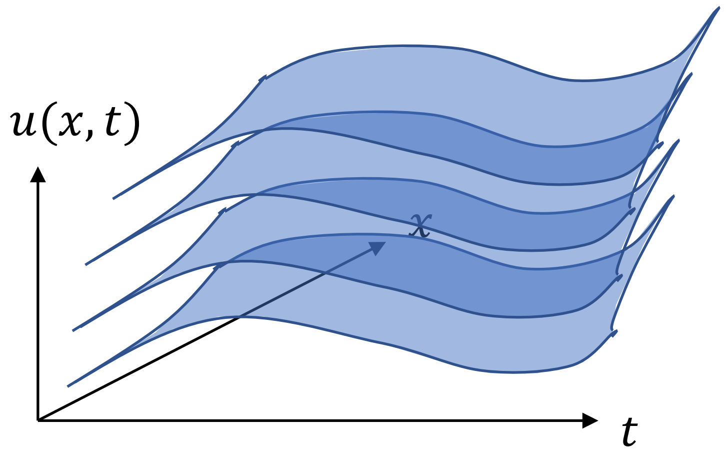
- Derivatives of u(x,t) w.r.t x and t can be calculated analytically because u(x,t) is exact!

Example of solving PDEs: Diffusion equation

$$\frac{\partial u}{\partial t} = a\frac{\partial^2 u}{\partial x^2}$$

1st order derivative in time t
2nd order derivatives in space x $\longrightarrow$ 1 initial condition
2 boundary conditions

$\longrightarrow$ $\boxed{GOAL:\ solve\ for\ u(x,t)}$



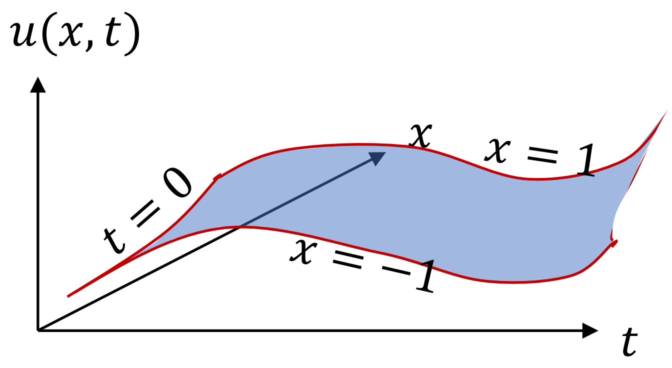
Need to give boundary and initial conditions to properly find a unique surface u(x,t)

Example of solving PDEs: Diffusion equation

$$\frac{\partial u}{\partial t} = a\frac{\partial^2 u}{\partial x^2}$$

1st order derivative in time t
2nd order derivatives in space x → 1 initial condition
2 boundary conditions

→ $\boxed{GOAL:\ solve\ for\ u(x,t)}$



Need to give boundary and initial conditions to properly find a unique surface u(x,t)

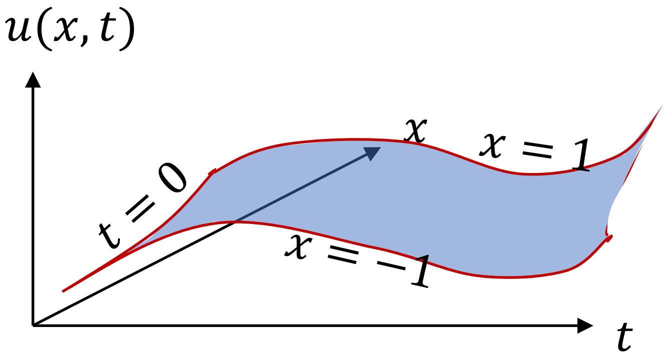
Example of solving PDEs: Diffusion equation

$$\frac{\partial u}{\partial t} = a\frac{\partial^2 u}{\partial x^2}$$

$Initial\ condition{:}\quad u(t=0,x) = f(x)$

$Boundary\ conditions{:}\quad u(t,x=-1) = u(t,x=1) = 0$

$\longrightarrow$ $GOAL{:}\ solve\ for\ u(x,t)$



Need to give boundary and initial conditions to properly find a unique surface u(x,t)

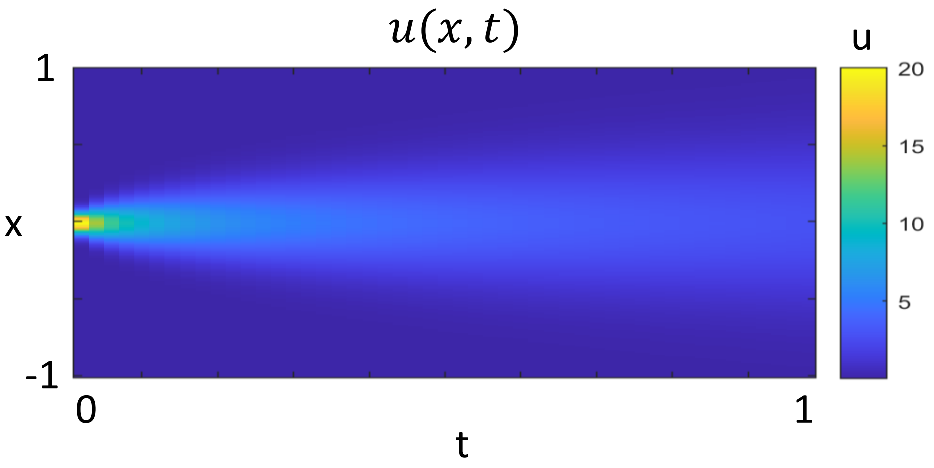
Solve it numerically

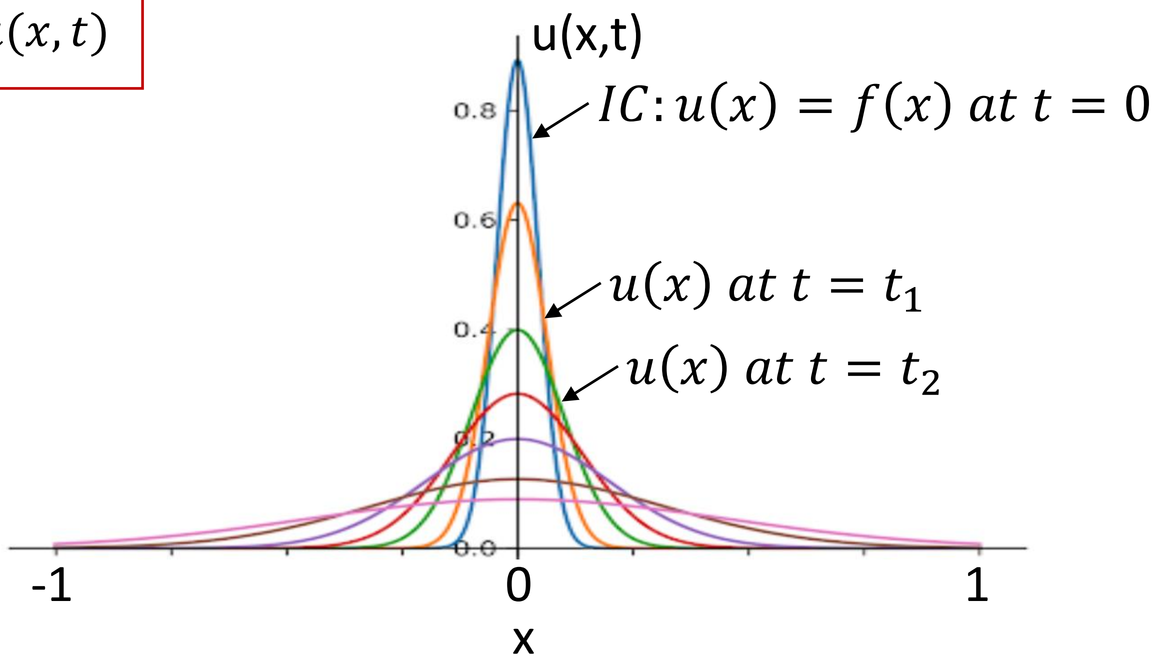
$$\frac{\partial u}{\partial t} = a\frac{\partial^2 u}{\partial x^2}$$

Initial condition: $u(t=0,x) = f(x)$
Boundary conditions: $u(t,x=-1) = u(t,x=1) = 0$

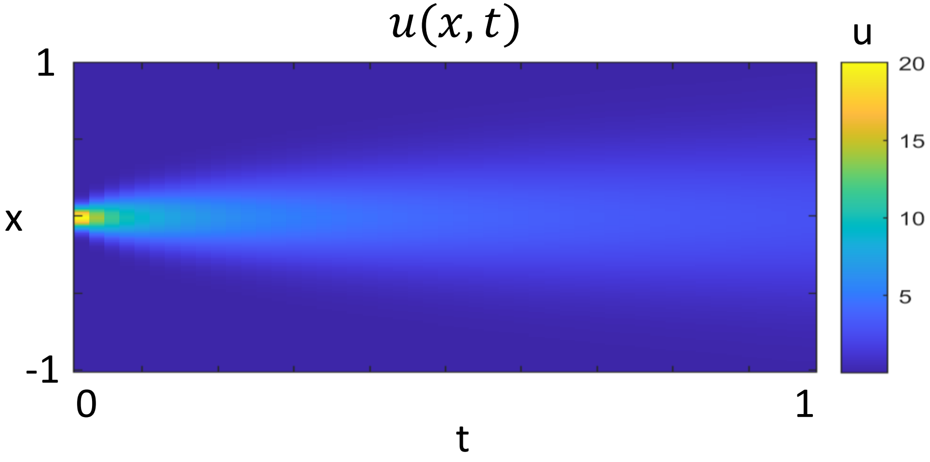
$GOAL: \ solve \ for \ u(x,t)$

Solved numerically





What do we need if we want to find u(x,t) with a NN?



1. Learn u empirically with data

   NN can fit any smooth and continuous functions!

   Give some sample data points and fit the surface u(x,t)

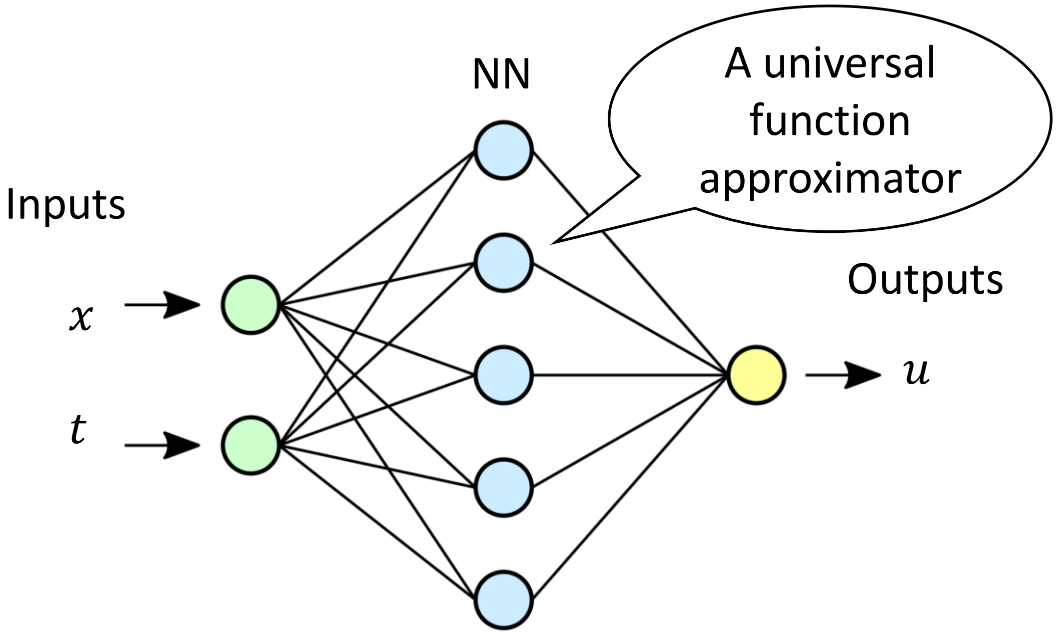
2. Learn u with physics equation + ICs + BCs

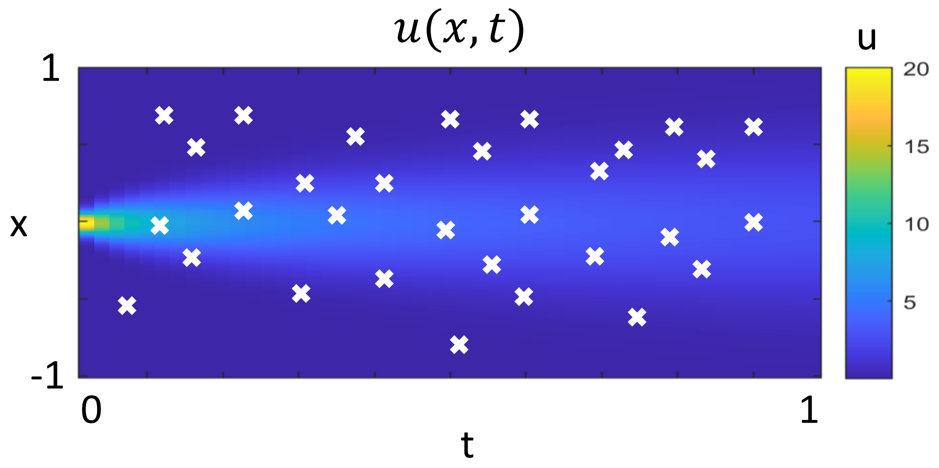
$$\frac{\partial u}{\partial t} = a\frac{\partial^2 u}{\partial x^2}$$

$Initial\ condition$: $u(t=0,x) = f(x)$

$Boundary\ conditions$: $u(t,x=-1) = u(t,x=1) = 0$

Learn u empirically





Training data (ground truth):

input: $t_i, x_i,\quad i = 1 \dots N$

output: $u_i,\quad i = 1 \dots N$

NN Prediction:

input: $t_i, x_i,\quad i = 1 \dots N$

output: $u_{pred}(t_i, x_i),\quad i = 1 \dots N$

What would be the loss function?

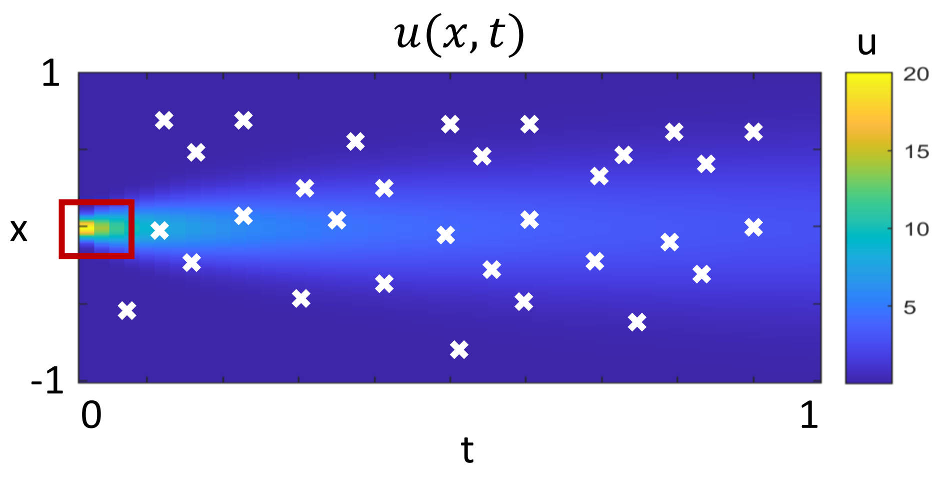
Data loss:

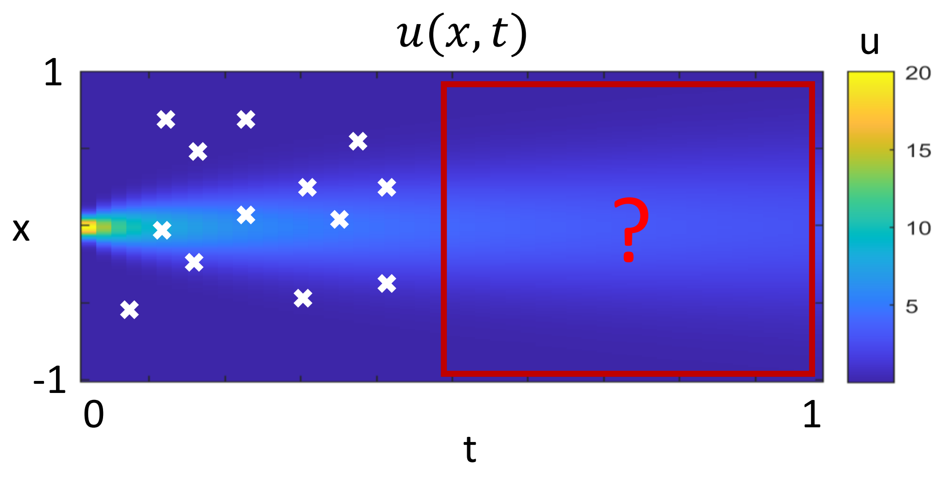
minimize $$\frac{1}{N}\sum_{i}^{N}\left|u_i - u_{pred}(t_i, x_i)\right|^2$$

GOAL: Find NN that minimizes the difference between groud truth and prediction (the loss function)

Problems of empirical learning

- NN will need lots of u training data to well approximate u(x,t).
- The learnt u(x,t) cannot be generalized to new t and x domains!





Problems of empirical learning

- NN will need lots of u training data to well approximate u(x,t).
- The learnt u(x,t) cannot be generalized to new t and x domains!
- Can we use **physical constraints** to reduce the training data needed to make NN approximate u(x,t)?
- Can we use **physical constraints** to help NN generalize to new t and x domains?

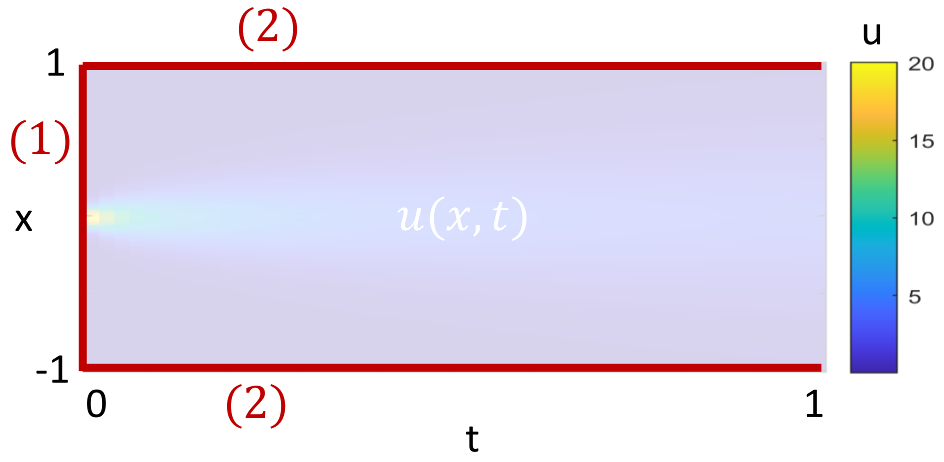
Learn u with physics equation + ICs + BCs

(3) $\frac{\partial u}{\partial t} = a\frac{\partial^2 u}{\partial x^2}$

(1) $Initial\ condition:\quad u(t=0,x) = f(x)$

(2) $Boundary\ conditions:\quad u(t,x=-1) = u(t,x=1) = 0$

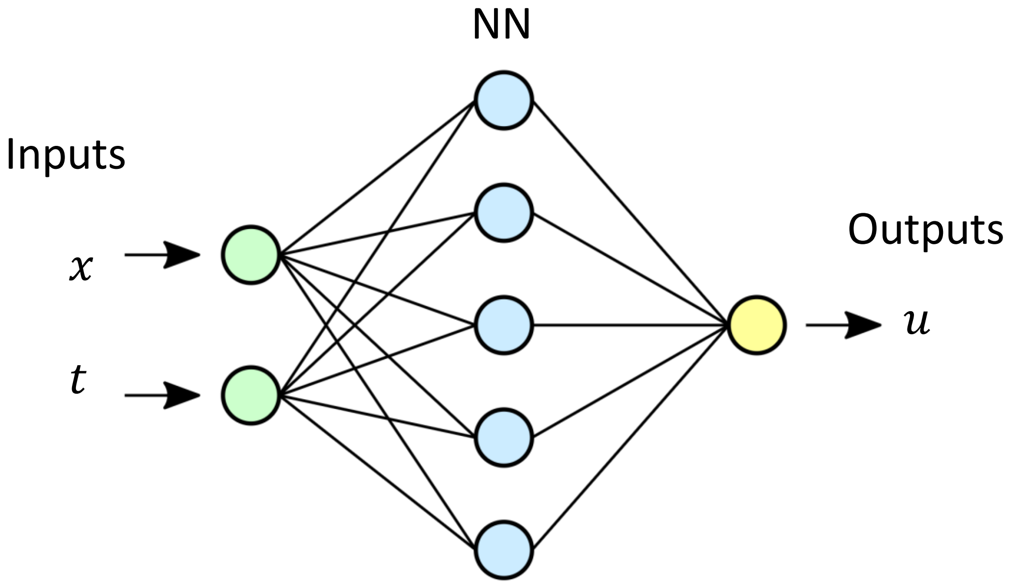
$\longrightarrow$ $\boxed{GOAL:\ solve\ for\ u(x,t)}$

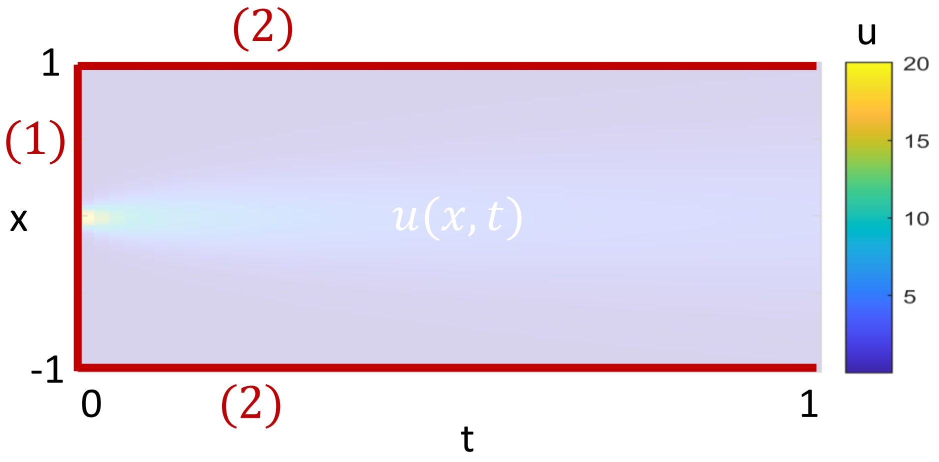


Mathematically, the information we need to get u(x,t) is **(1) ICs + (2) BCs** + **(3) physics equation**.

Instead of traditional numerical solver, can we use a NN to predict u(x,t) using (1), (2), (3)?

Learn u with physics equation + ICs + BCs





Use NN to search for a surface that satisfies (1) ICs + (2) BCs + (3) physics equation.

1. Initial NN(x,t) = u(x,t). x,t are inputs, u is output

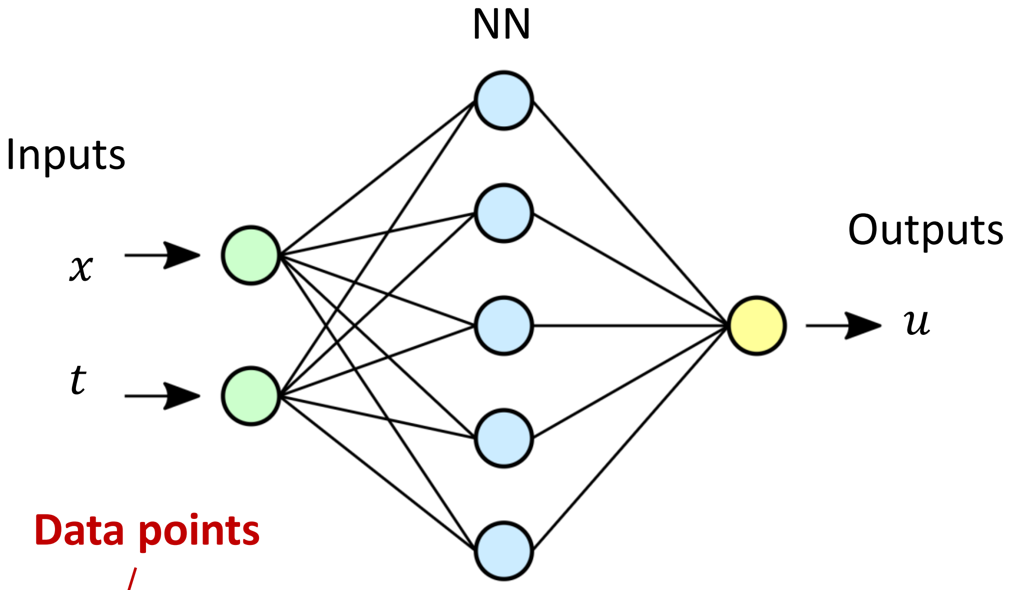
2. For a NN, all derivatives of u w.r.t x and t can be calculated analytically because u(x,t) is exact!

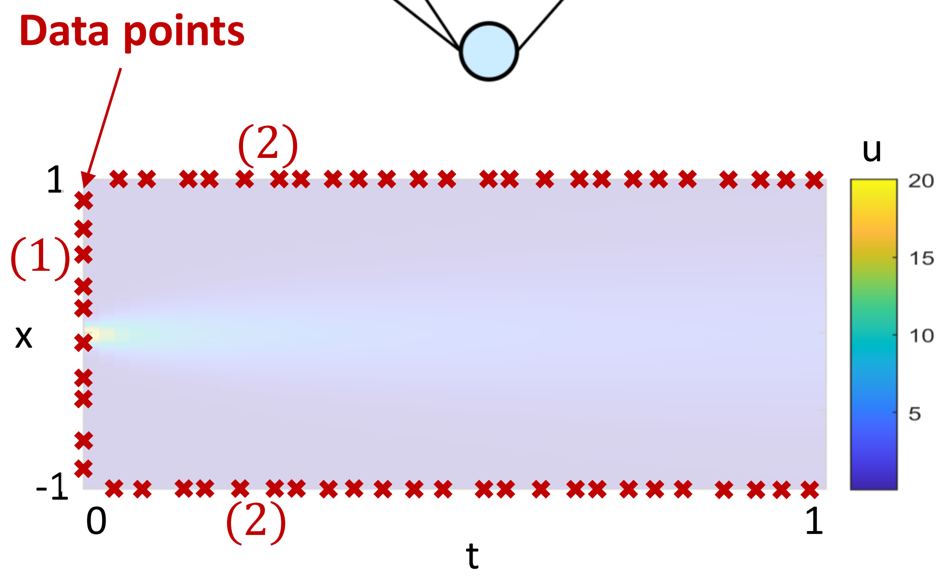
3. We can calculate $\frac{\partial u}{\partial t}, \frac{\partial^2 u}{\partial x^2}$

4. In the cost function, minimize $\left[\frac{\partial u}{\partial t} - a\frac{\partial^2 u}{\partial x^2}\right]^2$

Turn the problem into an optimization problem!

Physics-informed NN





Training data (ground truth):

What would be the loss function?

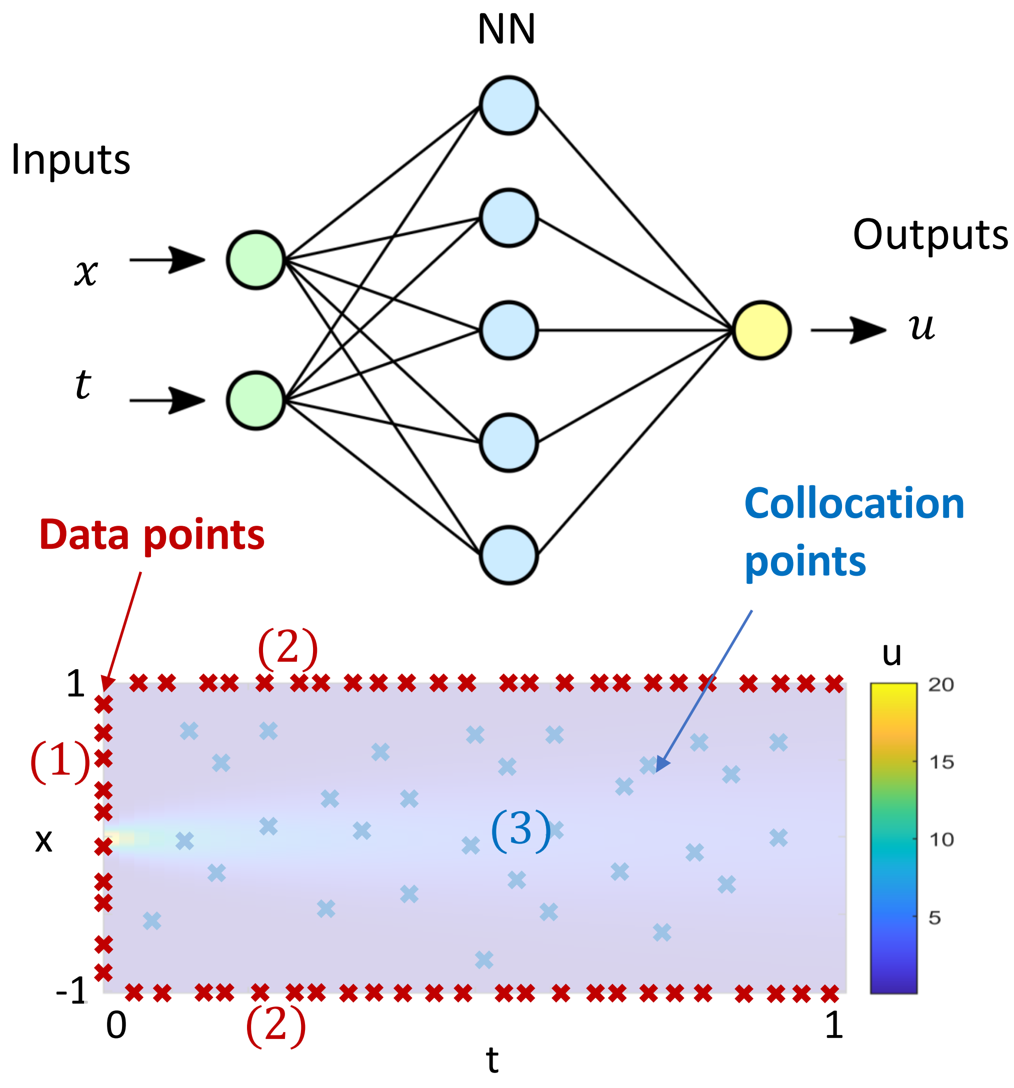
minimize $\frac{1}{N}\sum_{i}^{N}\left|u_0^i - u_{pred}(t=0, x_i)\right|^2$ (1) IC

minimize $\frac{1}{M}\sum_{j}^{M}\left|u_{lb}^j - u_{pred}(t_j, x=-1)\right|^2$

$+\frac{1}{M}\sum_{j}^{M}\left|u_{ub}^j - u_{pred}(t_j, x=1)\right|^2$ (2) BC

GOAL: Find NN that minimizes the loss function

Physics-informed NN



Q: How to incorporate physics equation in the loss function?

$$(3)\quad \frac{\partial u}{\partial t} = a\frac{\partial^2 u}{\partial x^2} \longrightarrow \frac{\partial u}{\partial t} - a\frac{\partial^2 u}{\partial x^2} = 0$$

Recall: NN(x,t) = u(x,t) is a smooth, analytical function

$\frac{\partial u}{\partial t}, \frac{\partial^2 u}{\partial x^2}$ can be directly calculated at the collocation points.

What would be the loss function?

Equation loss:

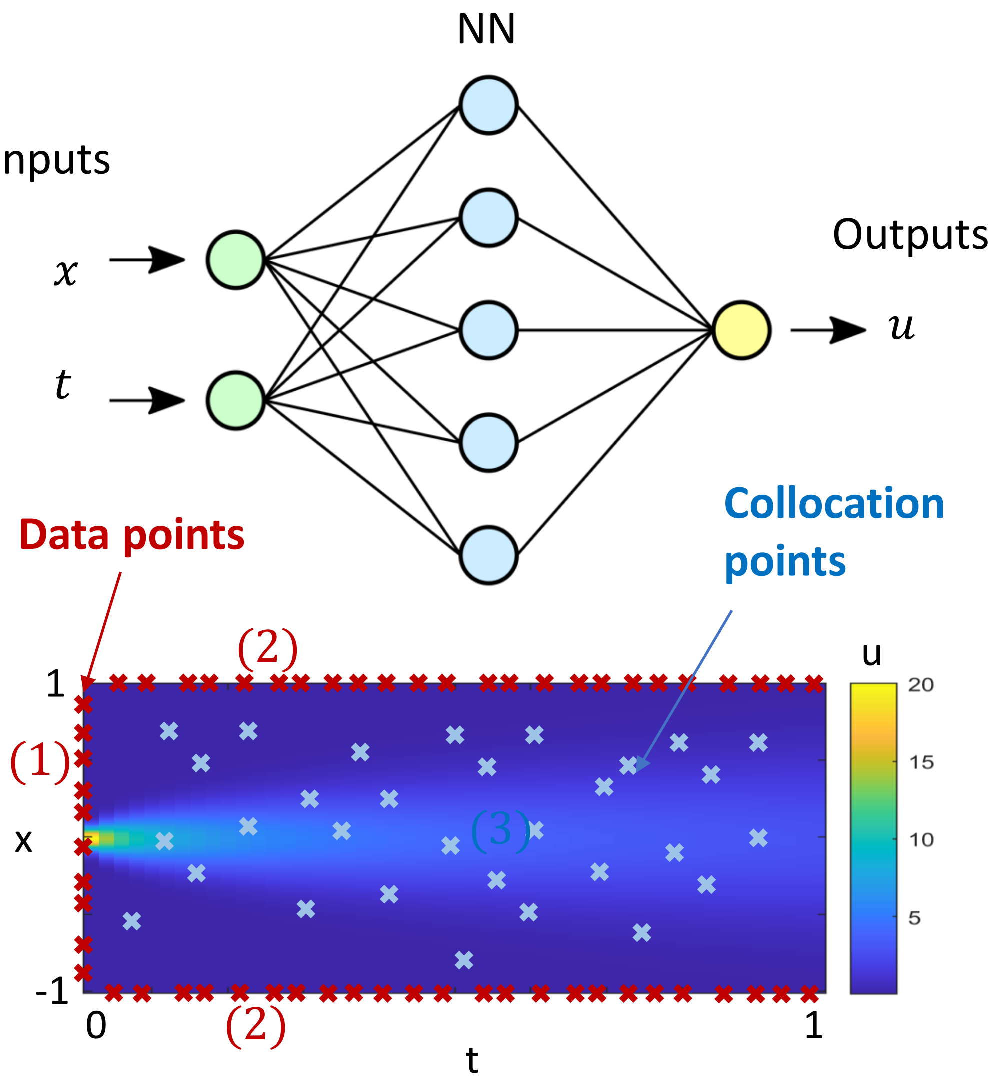
$$\text{minimize} \quad \frac{1}{N_f}\sum_{k}^{N_f}\left|\frac{\partial u_{pred}^k}{\partial t} - a\frac{\partial^2 u_{pred}^k}{\partial x^2}\right|^2 \quad \text{(3) Eqn}$$

$GOAL: Find\ NN\ that\ minimizes\ the\ loss\ function$

$\rightarrow solving\ for\ u(x,t)\ satisfying$ **(1) ICs + (2) BCs + (3) Eqn**

Raissi et al (2019)

Physics-informed NN



Given a partial differential equation of a general form:

$$u_t + \mathcal{N}[u] = 0,\ x \in \Omega,\ t \in [0, T]$$

where $\mathcal{N}[\cdot]$ is a nonlinear differential operator.
Define equation residue f as

$$f := u_t + \mathcal{N}[u]$$

Cost function: (MSE: mean squared error)

$$MSE = \underbrace{MSE_u}_{\text{Data loss}} + \underbrace{MSE_f}_{\text{Equation loss}},$$

Data loss

$$\frac{1}{N_u}\sum_{i=1}^{N_u} |u(t_u^i, x_u^i) - u^i|^2,$$

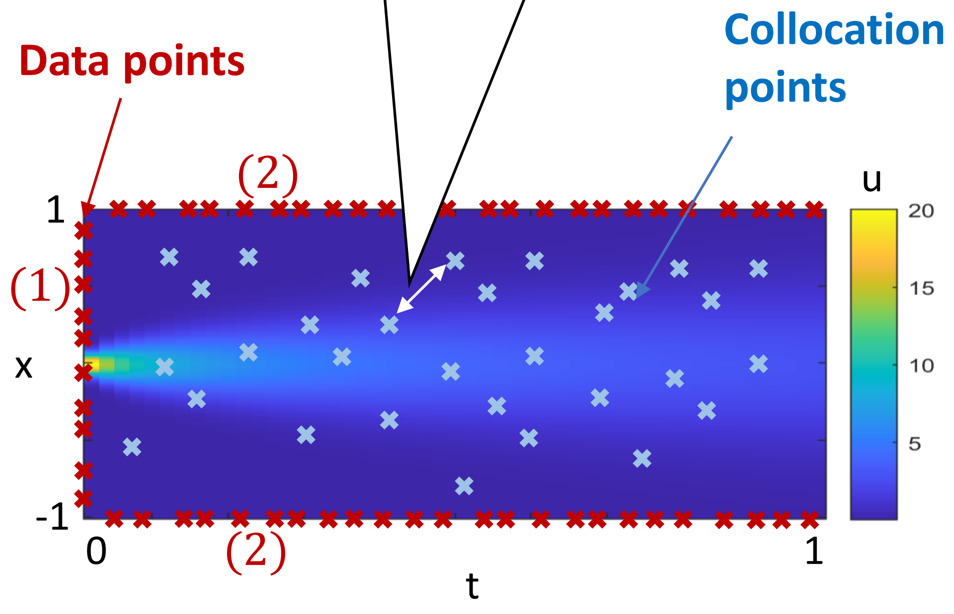
Data points

Equation loss

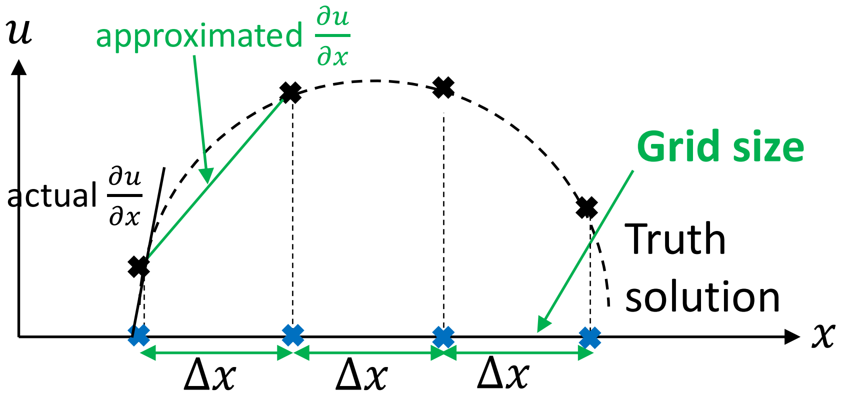
$$\frac{1}{N_f}\sum_{i=1}^{N_f} |f(t_f^i, x_f^i)|^2$$

Collocation points

Q: Wouldn't $\frac{\partial u}{\partial x}$ be poorly approximated between the collocation points that are far apart?



Finite difference: $\frac{\partial u}{\partial x} \approx \frac{u(x+\Delta x e_1)-u(x)}{\Delta x}$

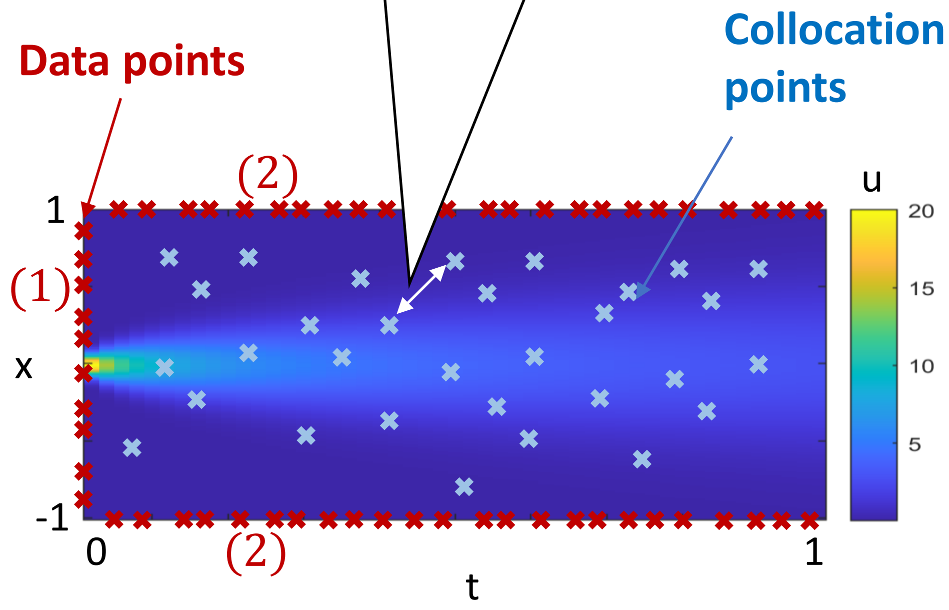


Drawbacks:

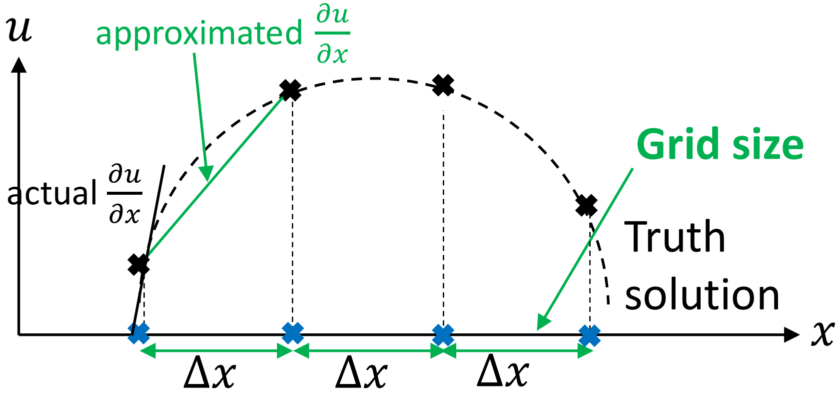
1. Truncation error: Δx needs to be sufficiently small

2. If we calculate derivatives of $u(x,t)$ w.r.t. $e_1, e_2, e_3 \ldots. e_n$, it requires $O(n)$ evaluations

Q: Wouldn't $\frac{\partial u}{\partial x}$ be poorly approximated between the collocation points that are far apart?



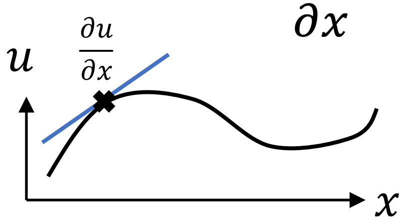
Finite difference: $\frac{\partial u}{\partial x} \approx \frac{u(x+\Delta x e_1)-u(x)}{\Delta x}$



Automatic differentiation: differentiate NN output with respect to their input coordinates.

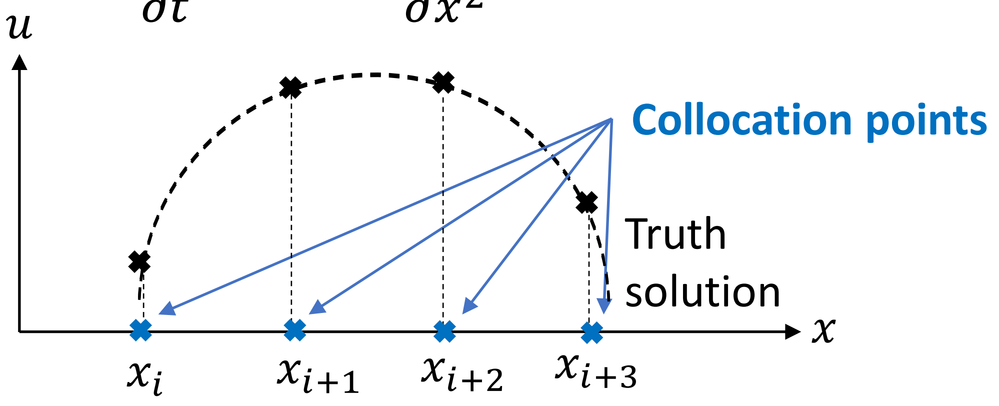
e.g. $\mathrm{NN(x)} = u = \sigma(w_2\sigma(w_1 x + b_1) + b_2)$,

$$\frac{\partial u}{\partial x} =$$



PINN: searches for a curve that satisfies

$$\frac{\partial u(x_i)}{\partial t} - a\frac{\partial^2 u(x_i)}{\partial x^2} \approx 0 \text{ at the collocation pts}$$

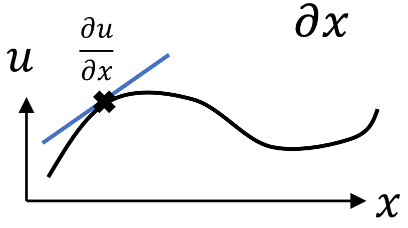


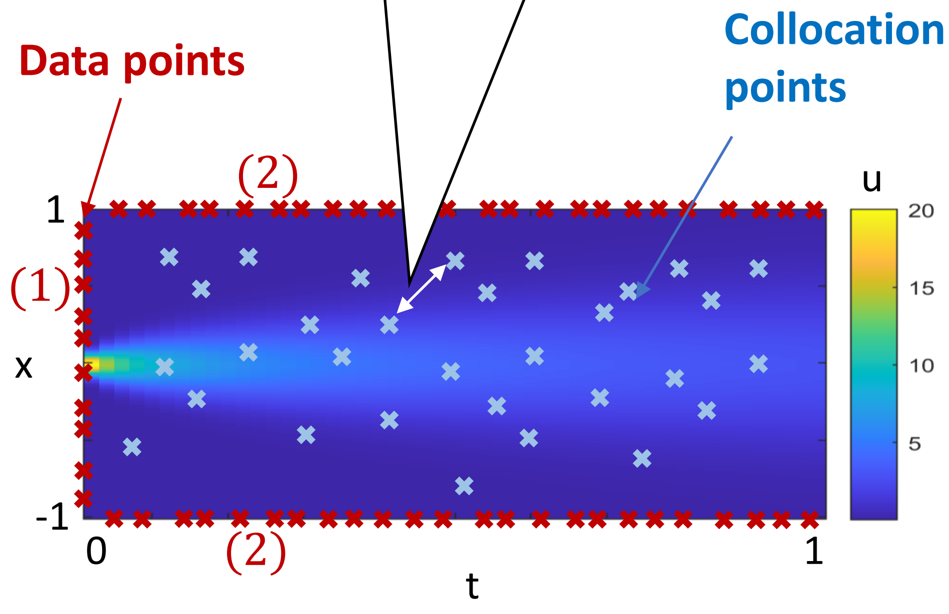
Q: Wouldn't $\frac{\partial u}{\partial x}$ be poorly approximated between the collocation points that are far apart?

Automatic differentiation: differentiate NN output with respect to their input coordinates.

e.g. $\mathrm{NN(x)} = u = \sigma(w_2\sigma(w_1 x + b_1) + b_2)$,

$$\frac{\partial u}{\partial x} = \sigma'(z_2)w_2\sigma'(z_1)w_1$$





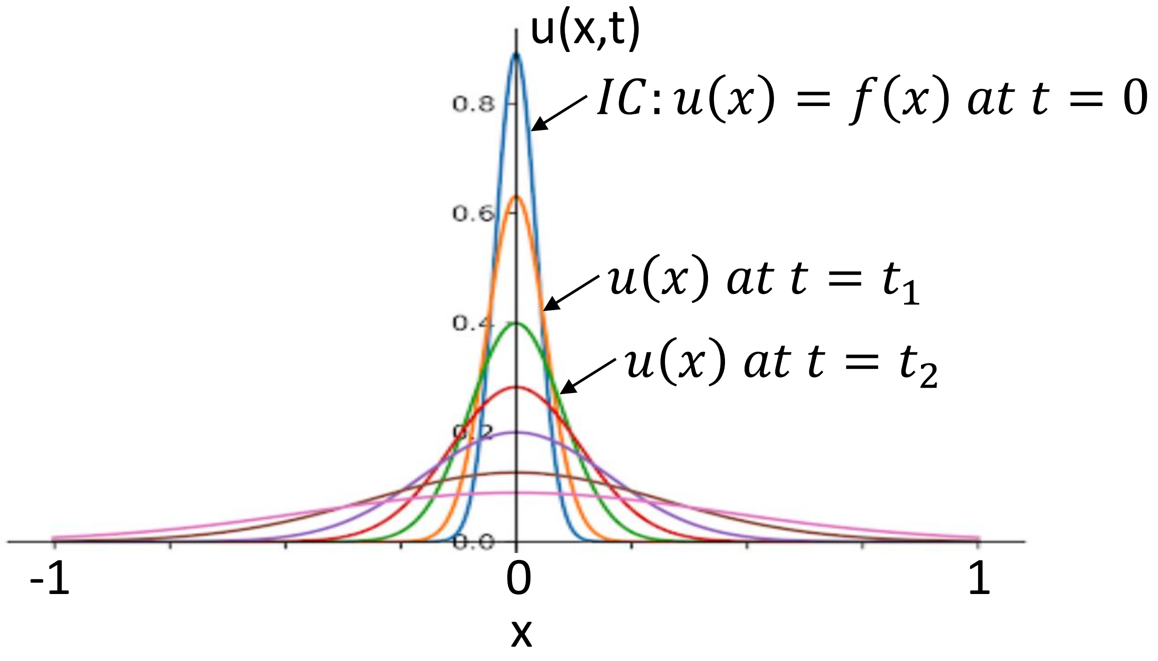
Comparison

(3) $\frac{\partial u}{\partial t} = a \frac{\partial^2 u}{\partial x^2}$

(1) $Initial\ condition: \quad u(t=0, x) = f(x)$

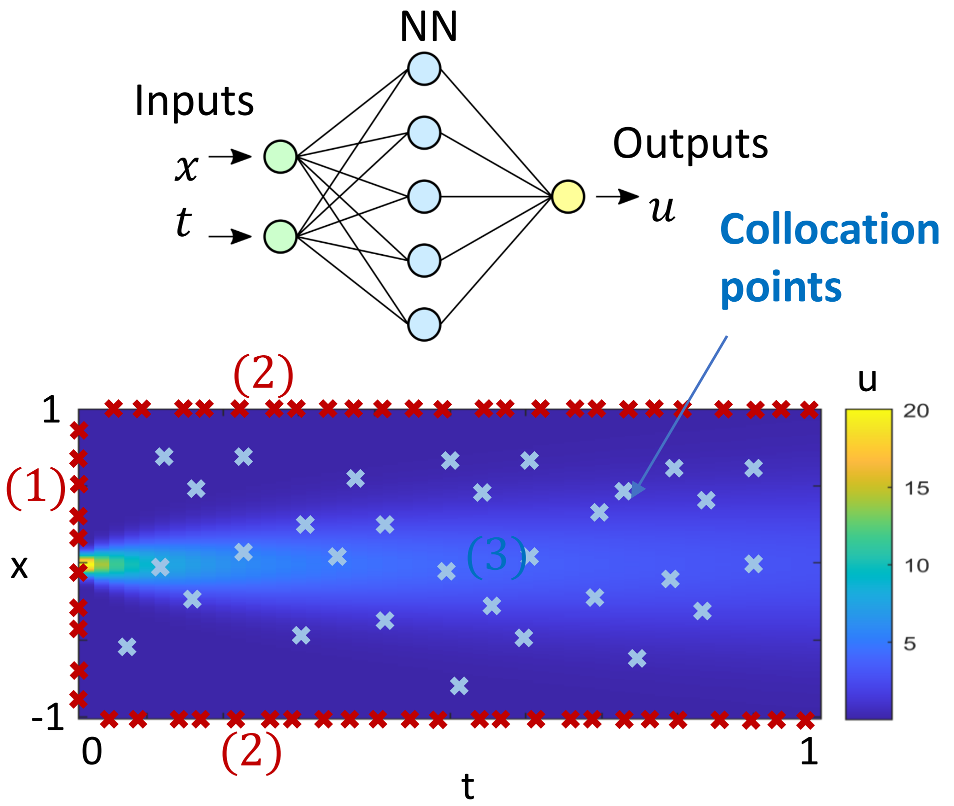
(2) $Boundary\ conditions: \quad u(t, x=-1) = u(t, x=1) = 0$

Finite-difference methods: Starting with IC, calculate discretized spatial derivative, updating the solution with a discretized timestep.



PINN:

Search for a surface that satisfies physics constraints (equation + ICs + BCs).



Pause and Ponder

1. What would happen if we tune the weights between eqn and data loss?

$$MSE = \underbrace{\alpha MSE_u}_{\text{Data loss}} + \underbrace{\gamma MSE_f}_{\text{Equation loss}}$$

2. What would happen to PINN if we use ReLU activation function?