Presented by


# Taiki Miyagawa

**Independent researcher, Japan**

(NEC Corporation)

# Takeru Yokota

**RIKEN iTHEMS & RQC, Japan**

“Physics-informed Neural Networks for **Functional Differential Equations**: Cylindrical Approximation and Its Convergence Guarantees”

[NeurIPS 2024]

**in 5 minutes**

# Summary

We developed the first solver for

general functional differential equations,

a game changer in functional analysis!

# Summary

We developed the first solver for

general functional differential equations,

a game changer in functional analysis!

# Summary

A **functional** is a function of a function.

We developed the first solver for

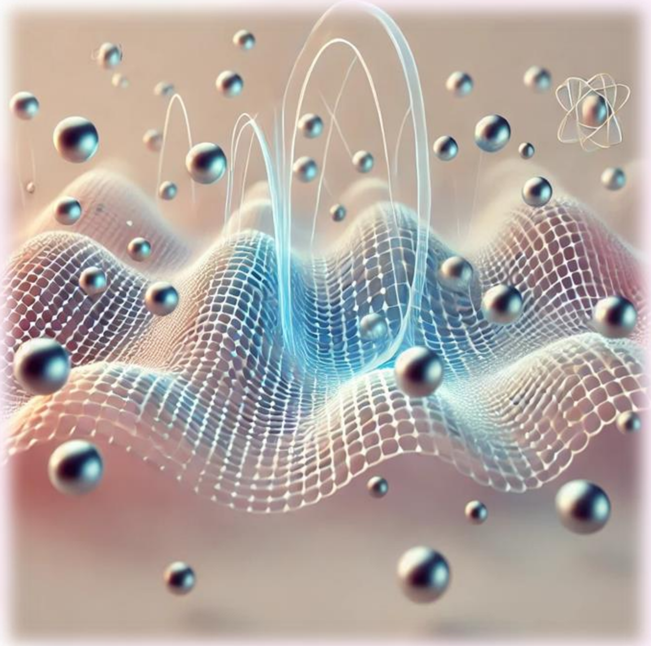
general **functional** differential equations,

a game changer in functional analysis!

# A **functional** is a function of a function.

Energy functional

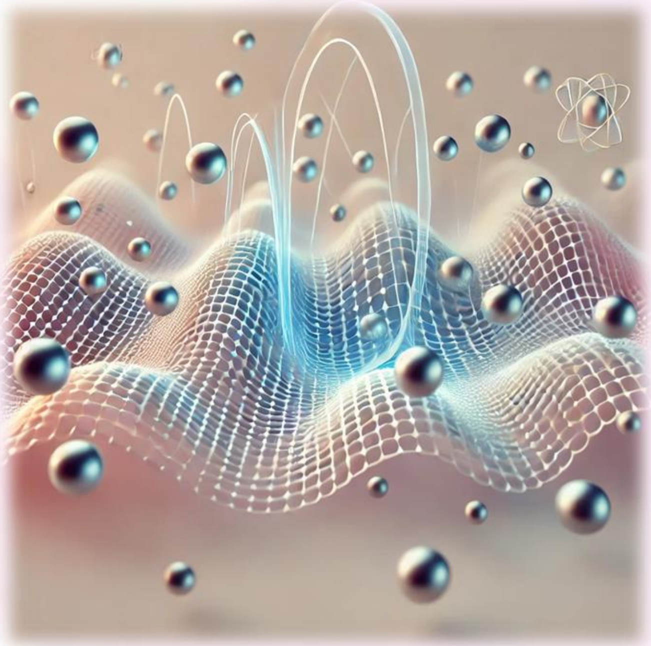
$$E([\rho])$$



# A **functional** is a function of a function.

Energy functional

$E([\rho])$



Characteristic functional

$\Phi([\theta], t)$

A **functional** is a function of a function.

## Summary

A functional is a function of a function.

We developed the first solver for

general functional differential equations,

a game changer in functional analysis!

# Summary

A functional differential equation is a differential equation involving functionals and functional derivatives.

We developed the first solver for

general functional differential equations,

a game changer in functional analysis!

# A **functional differential equation** is a differential equation involving functionals and functional derivatives.

Navier-Stokes-Hopf equation

$$\frac{\partial \Phi([\theta],t)}{\partial t} = \int dx\, \theta(x) \left( \frac{i}{2} \frac{\partial}{\partial x} \frac{\delta^2 \Phi([\theta],t)}{\delta\theta(x)\delta\theta(x)} + \nu \frac{\partial^2}{\partial x^2} \frac{\delta \Phi([\theta],t)}{\delta\theta(x)} \right)$$

# A **functional differential equation** is a differential equation involving functionals and **functional derivatives**.

$$\frac{\partial \Phi([\theta],t)}{\partial t} = \int dx\, \theta(x) \left( \frac{i}{2} \frac{\partial}{\partial x} \frac{\delta^2 \Phi([\theta],t)}{\delta\theta(x)\delta\theta(x)} + \nu \frac{\partial^2}{\partial x^2} \frac{\delta \Phi([\theta],t)}{\delta\theta(x)} \right)$$

# A functional differential equation is a differential equation involving functionals and functional derivatives.

$$\frac{\partial \Phi([\theta],t)}{\partial t} = \int dx\, \theta(x) \left( \frac{i}{2} \frac{\partial}{\partial x} \frac{\delta^2 \Phi([\theta],t)}{\delta\theta(x)\delta\theta(x)} + \nu \frac{\partial^2}{\partial x^2} \frac{\delta \Phi([\theta],t)}{\delta\theta(x)} \right)$$

$$\coloneqq \lim_{\epsilon\to 0} \left(\Phi([\theta(y)+\epsilon\delta(x-y)],t) - \Phi([\theta(y)],t)\right)/\epsilon$$

A **functional differential equation** is a differential equation involving functionals and **functional derivatives**.

$$\frac{\delta\Phi([\theta],t)}{\delta\theta(x)}$$

$$:= \lim_{\epsilon\to 0}(\Phi([\theta(y)+\epsilon\delta(x-y)],t)-\Phi([\theta(y)]),t)/\epsilon$$

# Summary

A functional differential equation is a differential equation involving functionals and functional derivatives.

We developed the first solver for

general functional differential equations,

a game changer in functional analysis!

# Summary

## Our solver

We developed the first solver for

1. Physics-informed Neural Network

general functional differential equations,

2. Cylindrical Approximation

a game changer in functional analysis!

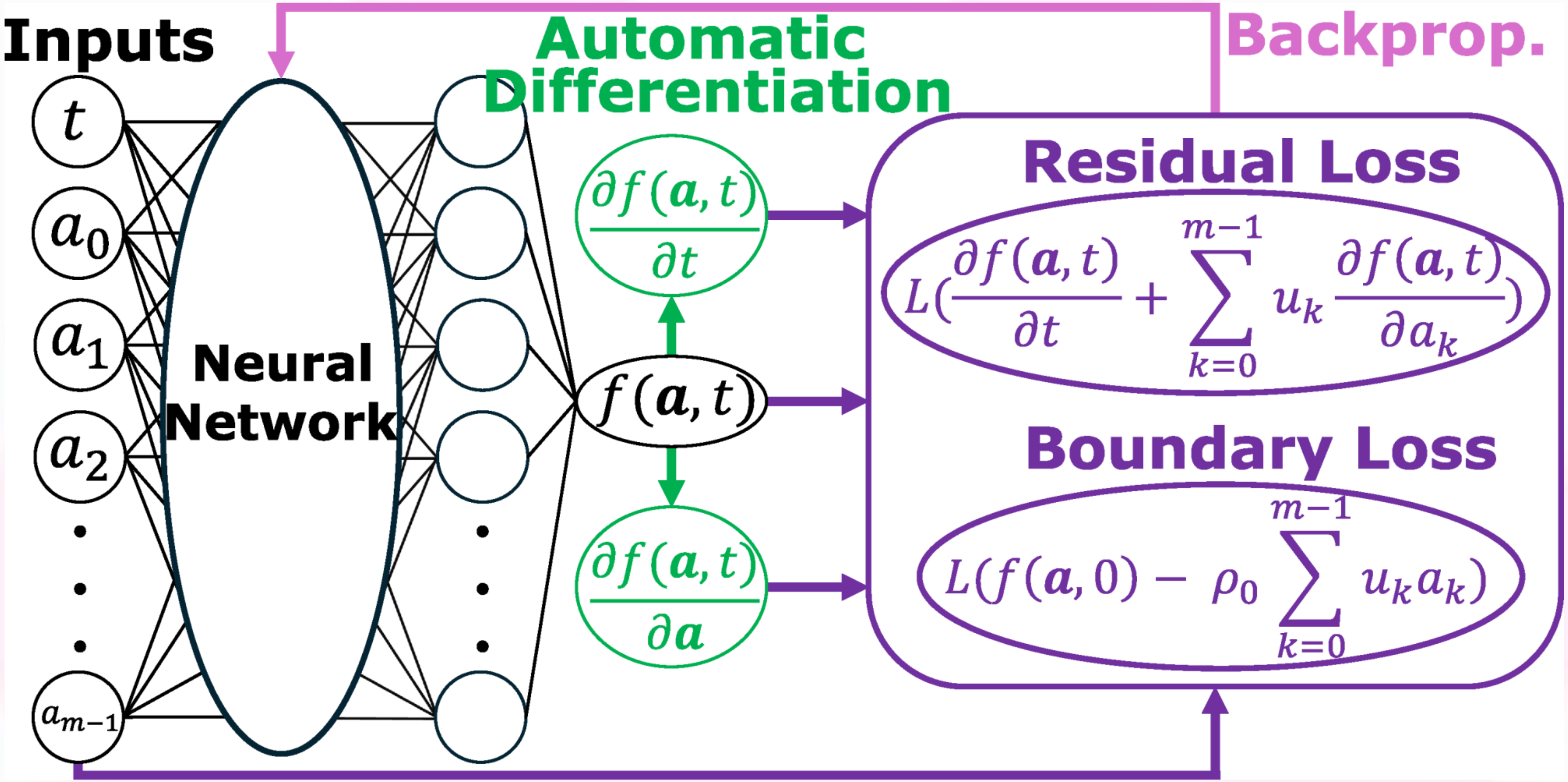
## Our solver

1. Physics-informed Neural Network

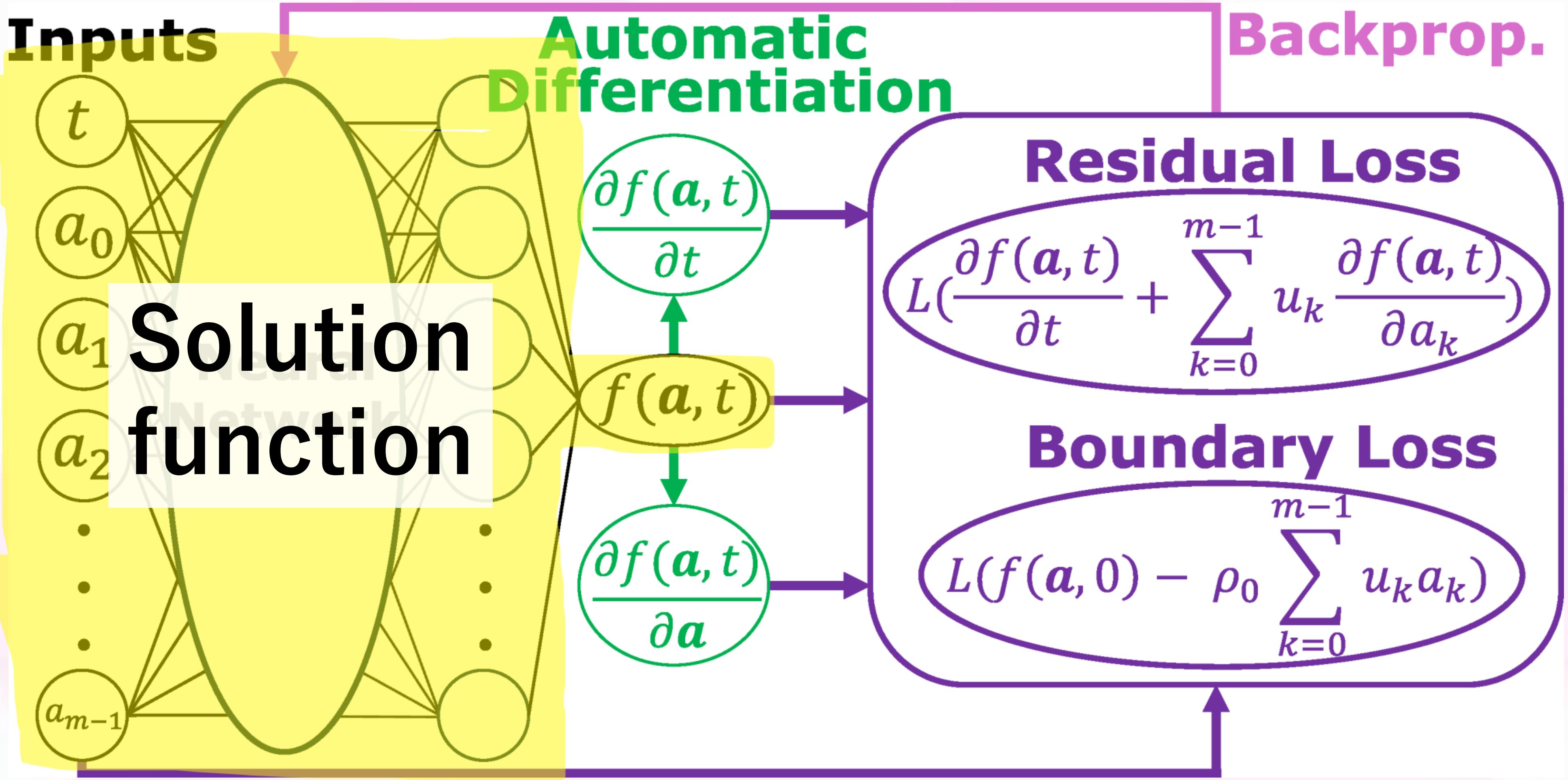
2. Cylindrical Approximation

# Our solver

# 1. Physics-informed Neural Network



# Our solver
# 1. Physics-informed Neural Network



# Our solver

1. Physics-informed Neural Network
2. Cylindrical Approximation

Our solver

1. Physics-informed Neural Network

2. Cylindrical Approximation

# Our solver

## 2. Cylindrical Approximation

$$\theta(x) \approx \sum_{k=0}^{m-1} \underbrace{a_k}_{\text{Variable coeff.}} \ \underbrace{\phi_k(x)}_{\text{Given basis}}$$

↓

$$F([\theta]) \approx f(\boldsymbol{a})$$

$$\frac{\delta F([\theta])}{\delta \theta(x)} \approx \sum_{k=0}^{m-1} \frac{\partial f(\boldsymbol{a})}{\partial a_k} \phi_k(x)$$

# Our solver

## 2. Cylindrical Approximation

$$\theta(x) \approx \sum_{k=0}^{m-1} \underset{\text{Variable coeff.}}{a_k} \; \underset{\text{Given basis}}{\phi_k(x)}$$

$$F([\theta]) \approx f(\boldsymbol{a})$$

$$\frac{\delta F([\theta])}{\delta\theta(x)} \approx \sum_{k=0}^{m-1} \frac{\partial f(\boldsymbol{a})}{\partial a_k} \phi_k(x)$$

FDE → High-dimensional PDE

# Our solver

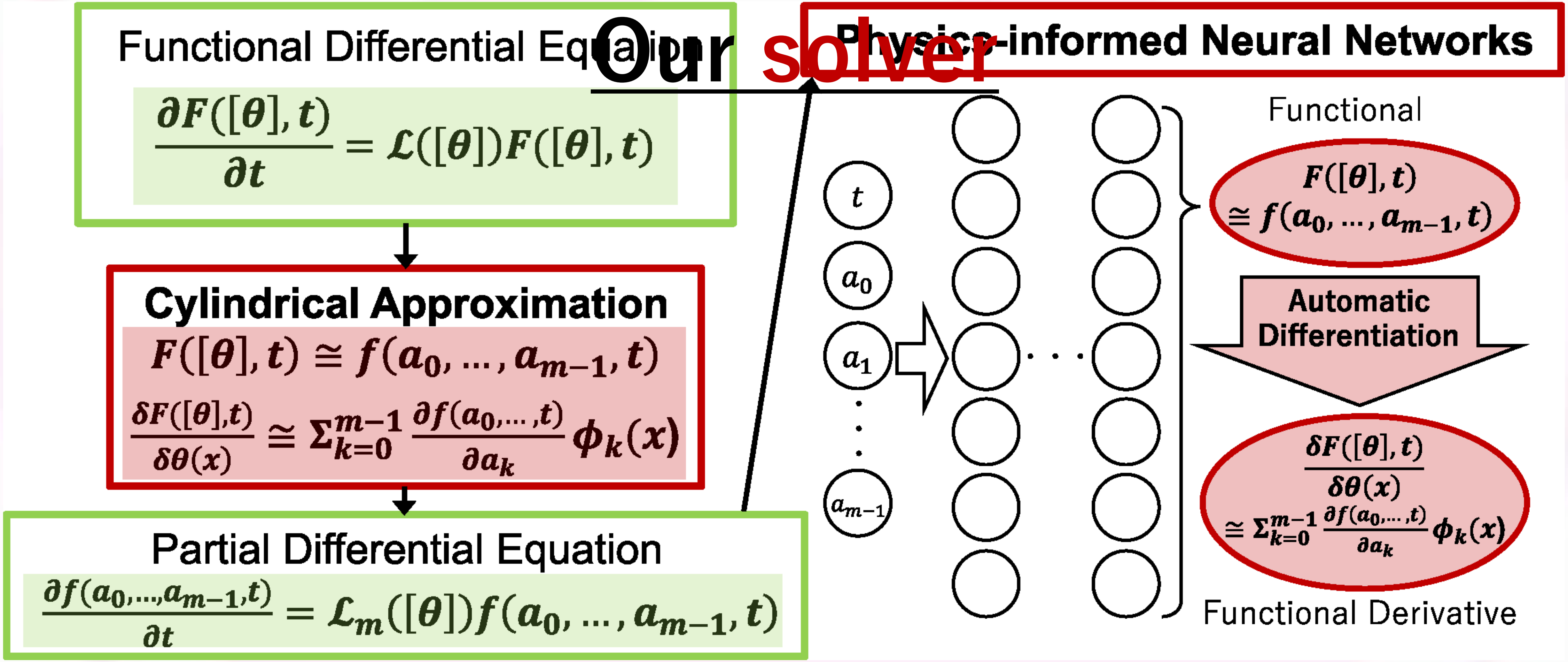
1. Physics-informed Neural Network

2. Cylindrical Approximation

# Our solver

1. Physics-informed Neural Network

2. Cylindrical Approximation



Our solver
Functional Differential Equation
$\frac{\partial F([\theta],t)}{\partial t} = \mathcal{L}([\theta])F([\theta],t)$
Cylindrical Approximation
$F([\theta],t) \cong f(a_0,\dots,a_{m-1},t)$
$\frac{\delta F([\theta],t)}{\delta\theta(x)} \cong \Sigma_{k=0}^{m-1}\frac{\partial f(a_0,\dots,t)}{\partial a_k}\phi_k(x)$
Partial Differential Equation
$\frac{\partial f(a_0,\dots,a_{m-1},t)}{\partial t} = \mathcal{L}_m([\theta])f(a_0,\dots,a_{m-1},t)$
Physics-informed Neural Networks
$t$
$a_0$
$a_1$
$a_{m-1}$
Functional
$F([\theta],t) \cong f(a_0,\dots,a_{m-1},t)$
Automatic Differentiation
$\frac{\delta F([\theta],t)}{\delta\theta(x)} \cong \Sigma_{k=0}^{m-1}\frac{\partial f(a_0,\dots,t)}{\partial a_k}\phi_k(x)$
Functional Derivative

# Our solver

**Functional Differential Equation**

$$\frac{\partial F([\theta],t)}{\partial t} = \mathcal{L}([\theta])F([\theta],t)$$

**Cylindrical Approximation**

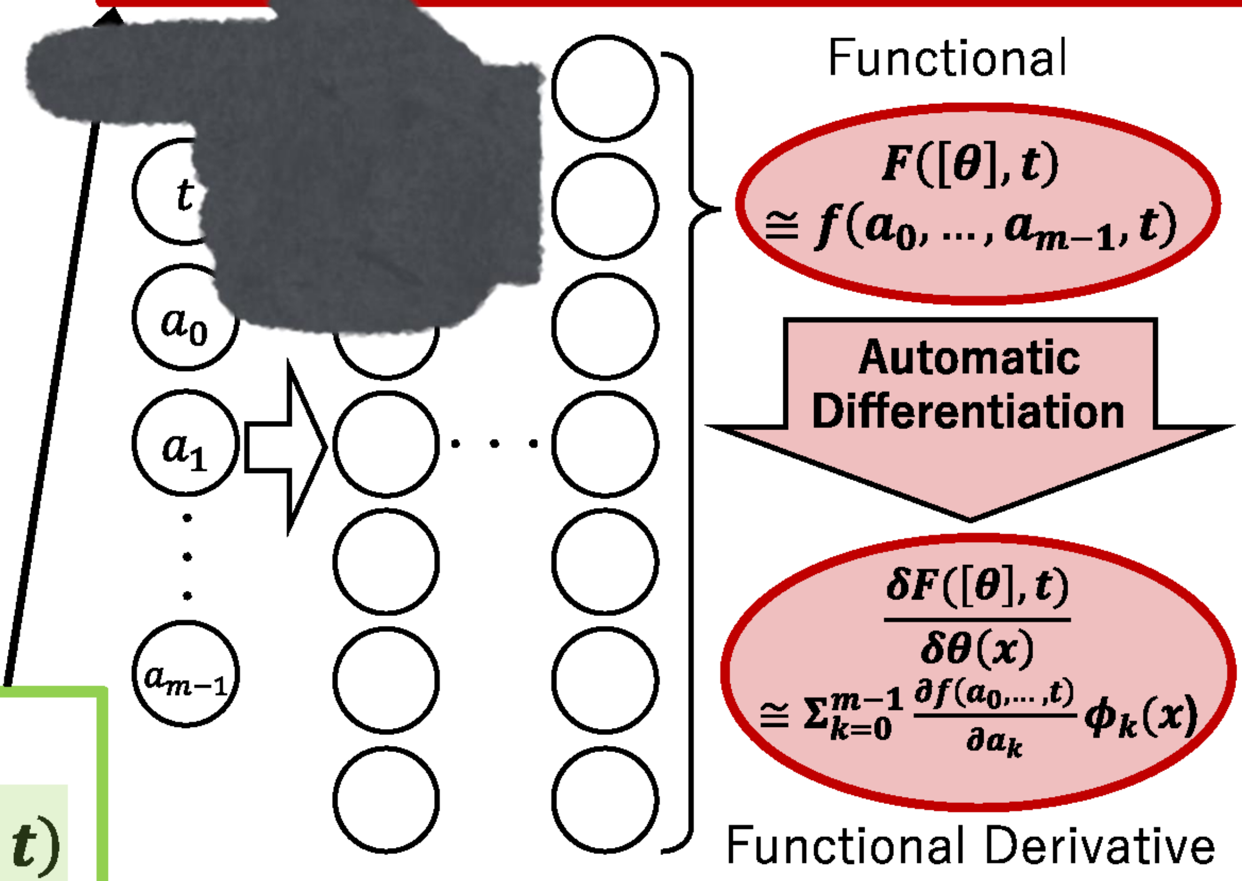
$$F([\theta],t) \cong f(a_0, \ldots, a_{m-1}, t)$$

$$\frac{\delta F([\theta],t)}{\delta \theta(x)} \cong \Sigma_{k=0}^{m-1} \frac{\partial f(a_0, \ldots, t)}{\partial a_k} \phi_k(x)$$

**Partial Differential Equation**

$$\frac{\partial f(a_0, \ldots, a_{m-1}, t)}{\partial t} = \mathcal{L}_m([\theta]) f(a_0, \ldots, a_{m-1}, t)$$

**Physics-informed Neural Networks**



# Our solver

**Functional Differential Equation**

$$\frac{\partial F([\theta],t)}{\partial t} = \mathcal{L}([\theta])F([\theta],t)$$

**Cylindrical Approximation**

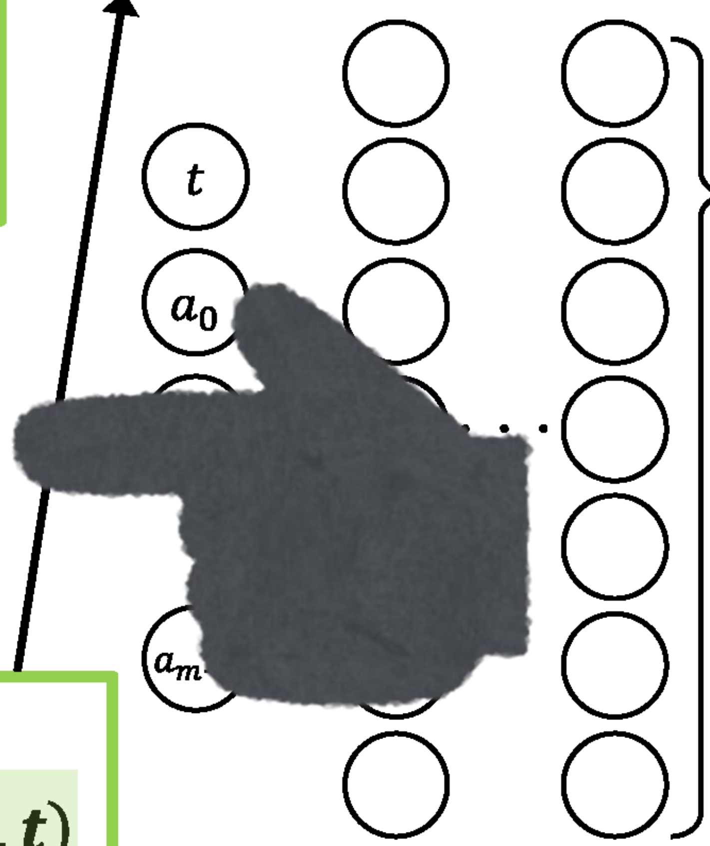
$$F([\theta],t) \cong f(a_0,\dots,a_{m-1},t)$$

$$\frac{\delta F([\theta],t)}{\delta\theta(x)} \cong \Sigma_{k=0}^{m-1}\frac{\partial f(a_0,\dots,t)}{\partial a_k}\phi_k(x)$$

**Partial Differential Equation**

$$\frac{\partial f(a_0,\dots,a_{m-1},t)}{\partial t} = \mathcal{L}_m([\theta])f(a_0,\dots,a_{m-1},t)$$

**Physics-informed Neural Networks**




Functional

$$F([\theta],t) \cong f(a_0,\dots,a_{m-1},t)$$

Automatic Differentiation

$$\frac{\delta F([\theta],t)}{\delta\theta(x)} \cong \Sigma_{k=0}^{m-1}\frac{\partial f(a_0,\dots,t)}{\partial a_k}\phi_k(x)$$

Functional Derivative

# Our solver

**Functional Differential Equation**

$$\frac{\partial F([\theta],t)}{\partial t} = \mathcal{L}([\theta])F([\theta],t)$$

**Cylindrical Approximation**

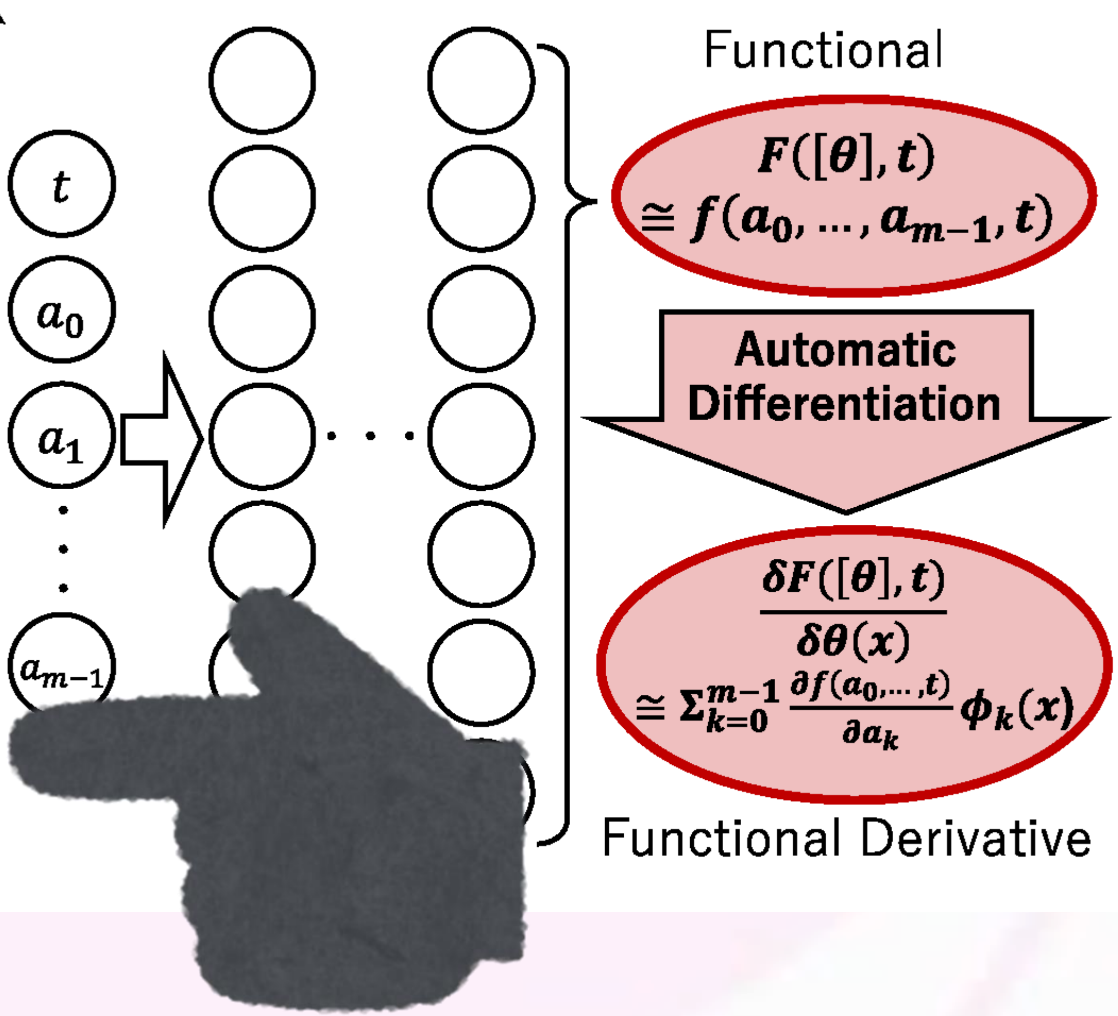
$$F([\theta],t) \cong f(a_0,\dots,a_{m-1},t)$$

$$\frac{\delta F([\theta],t)}{\delta\theta(x)} \cong \Sigma_{k=0}^{m-1}\frac{\partial f(a_0,\dots,t)}{\partial a_k}\phi_k(x)$$

**Partial Differential Equation**

$$\frac{\partial f(a_0,\dots,a_{m-1},t)}{\partial t} = \mathcal{L}_m([\theta])f(a_0,\dots,a_{m-1},t)$$

**Physics-informed Neural Networks**



# Our solver

**Functional Differe**

$$\frac{\partial F([\theta],t)}{\partial t} = \mathcal{L}(\lbrack$$

↓

**Cylindrical Approximation**

$$F([\theta],t) \cong f(a_0,\dots,a_{m-1},t)$$

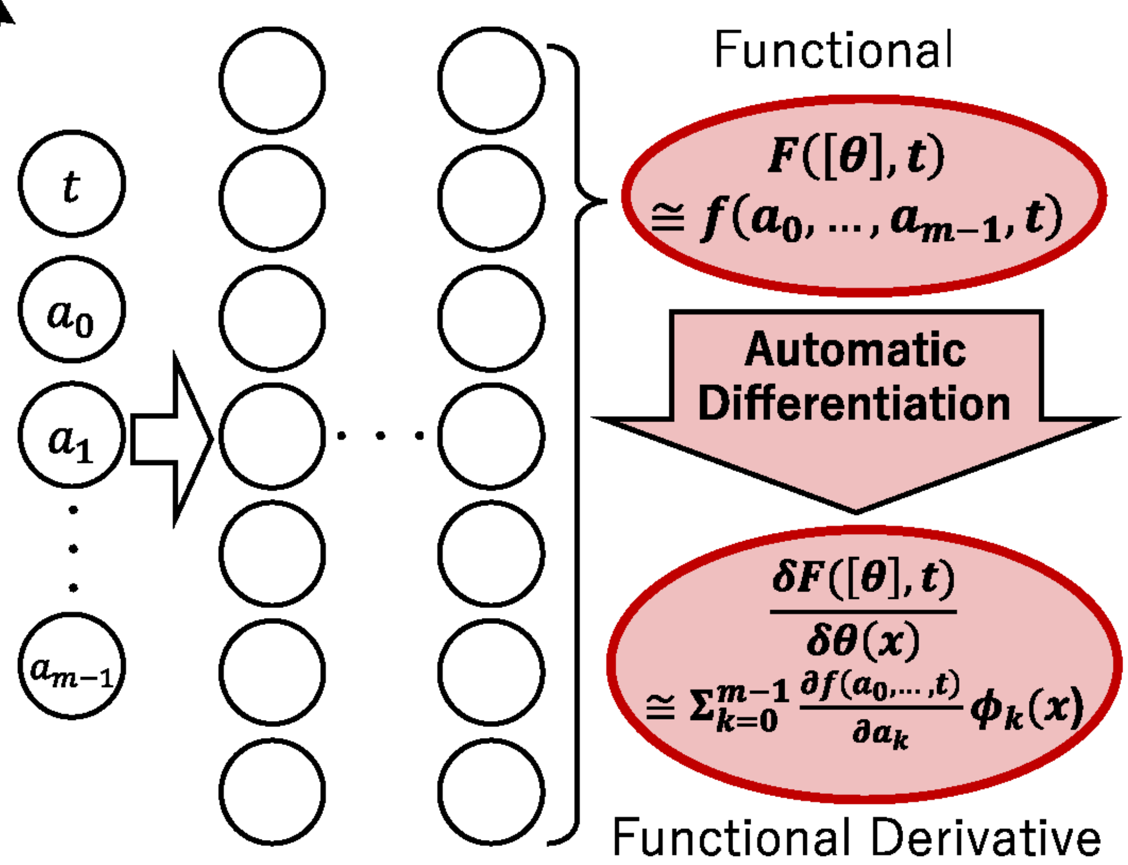
$$\frac{\delta F([\theta],t)}{\delta\theta(x)} \cong \Sigma_{k=0}^{m-1}\frac{\partial f(a_0,\dots,t)}{\partial a_k}\phi_k(x)$$

↓

**Partial Differential Equation**

$$\frac{\partial f(a_0,\dots,a_{m-1},t)}{\partial t} = \mathcal{L}_m([\theta])f(a_0,\dots,a_{m-1},t)$$

**Physics-informed Neural Networks**



# Our solver

**Functional Differential Equation**

$$\frac{\partial F([\theta],t)}{\partial t} = \mathcal{L}([\theta])F([\theta],t)$$

↓

**Cylindrical Approximation**

$$F([\theta],t) \cong f(a_0,\dots,a_{m-1},t)$$

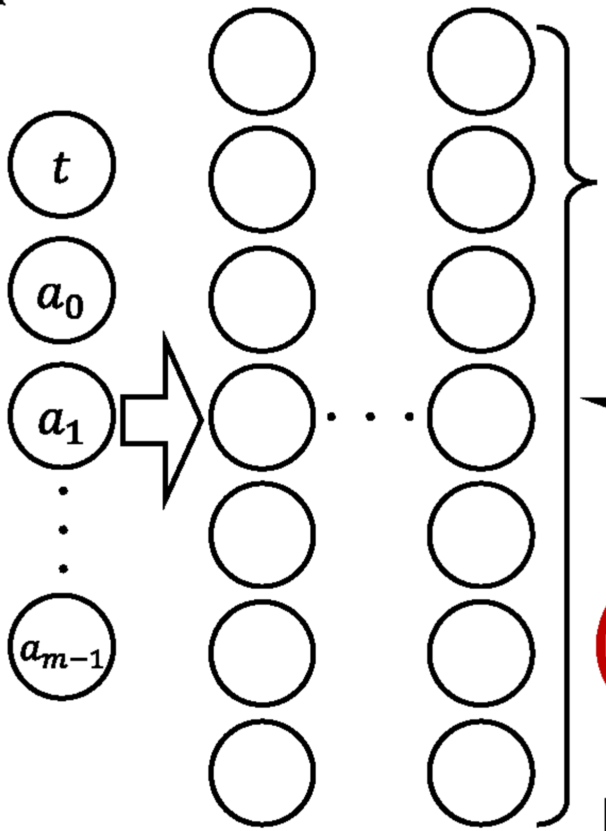
$$\frac{\delta F([\theta],t)}{\delta\theta(x)} \cong \Sigma_{k=0}^{m-1}\frac{\partial f(a_0,\dots,t)}{\partial a_k}\phi_k(x)$$

↓

**Partial Differential Equation**

$$\frac{\partial f(a_0,\dots,a_{m-1},t)}{\partial t} = \mathcal{L}_m([\theta])f(a_0,\dots,a_{m-1},t)$$

**Physics-informed Neural Networks**




Functional

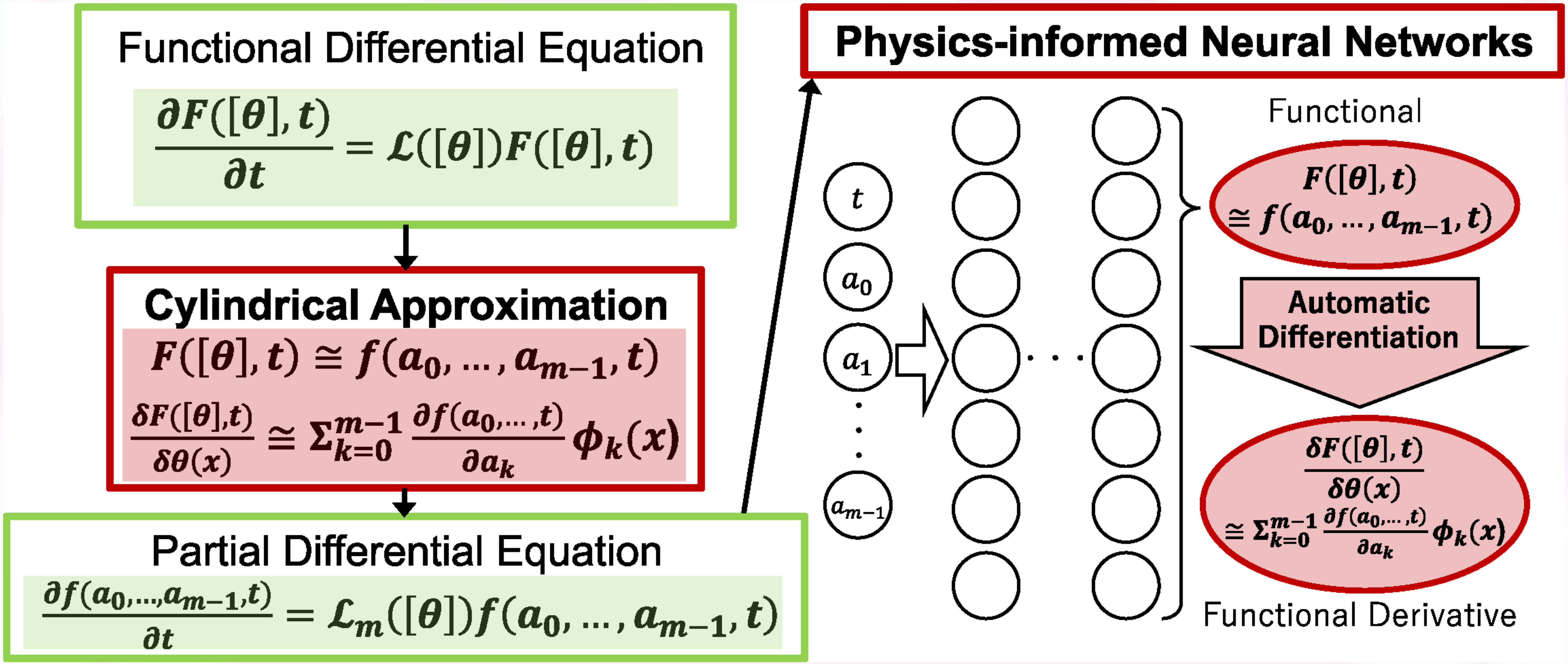
$$F([\theta],t) \cong f(a_0,\dots,a_{m-1},t)$$

Automatic Differentiation

$$\frac{\delta F([\theta],t)}{\delta\theta(x)} \cong \Sigma_{k=0}^{m-1}\frac{\partial f(a_0,\dots,t)}{\partial a_k}\phi_k(x)$$

Functional Derivative

# Functional PINN



# Summary

We developed the first solver for general functional differential equations, a game changer in functional analysis!

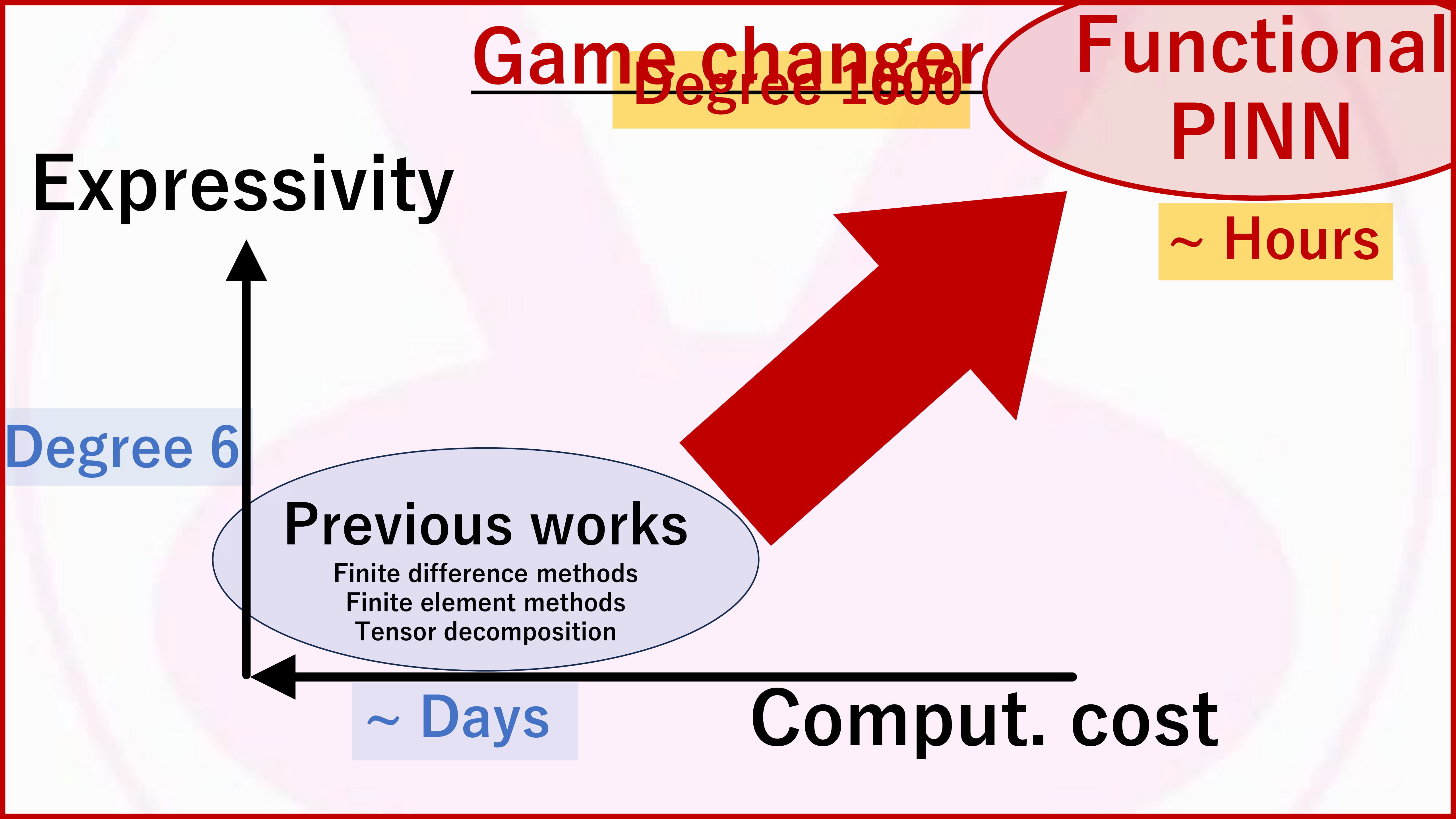
# Game changer

# Summary

We developed the first solver for

general functional differential equations,

a Game changer in functional analysis!



Game changer
Degree 1000
Functional PINN
~ Hours
Expressivity
Degree 6
Previous works
Finite difference methods
Finite element methods
Tensor decomposition
~ Days
Comput. cost

# Potential applications

Aerospace engineering

Environmental science

Medical science

Civil engineering

Meteorology

Architecture

Turbulence theory

Density functional theory

Quantum field theory

Mean-field game theory

Solid-state physics

Materials science

Catalysis and surface science

Bioinorganic chemistry

Radiation effects

# Thank you for watching <3

# Summary

We propose **the first learning scheme for functional differential equations** (FDEs) to address the significant computational complexity and limited approximation ability.

Our model, Functional PINN, **exponentially extends the class of input functions and functional derivatives (from polynomials of degree from 6 to 1000) and reduces computational costs (from ~days to ~hours), a game changer in functional analysis!**

Functional PINN consists of two key ideas: **the physics-informed neural network (a universal PDE solver) and the cylindrical approximation (spectrum decomposition of the input functions)**.

**We prove the convergence** of the approximated functional derivatives and FDE solutions, ensuring the cylindrical approximation to be safely applied to FDEs.

Our experimental results show that our model accurately approximates not only the FDE solutions but also their functional derivatives, achieving $L^1$ relative error of $\sim 10^{-3}$ on the functional transport equation and the Burgers-Hopf equation.