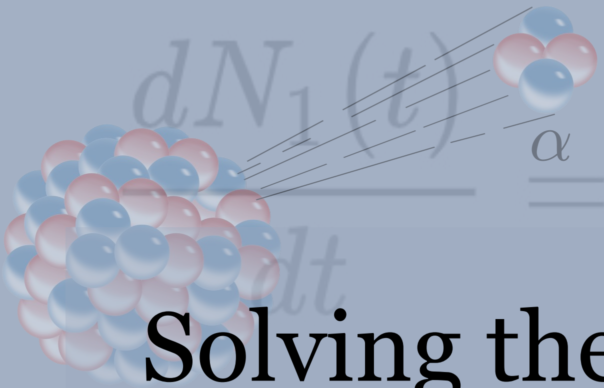
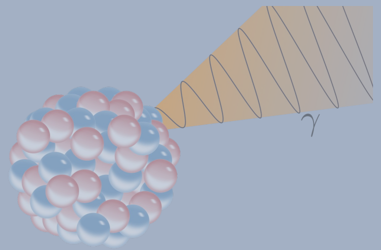


Solving the Bateman Equations Using Physics Informed Neural Networks (PINN)

Presenter: Genevieve Alpar



PAUL SCHERRER INSTITUT PSI

What are the Bateman Equations?

- Mathematical model describing the abundances and activities in decay chains of radioactive isotopes
- Used in fields of nuclear physics, radiochemistry, nuclear medicine, etc.

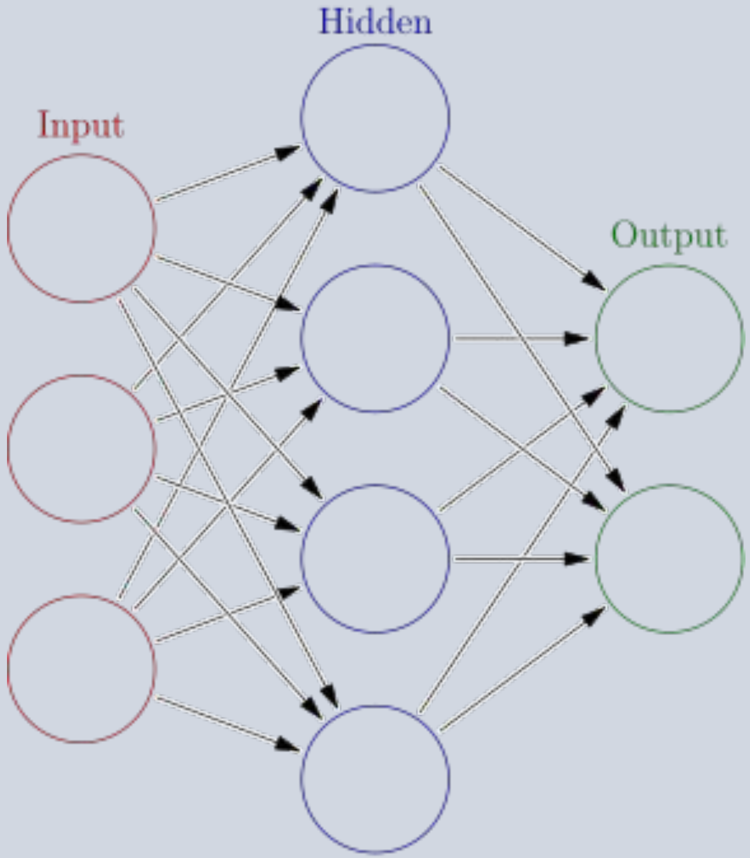
- $\vec{N}(t)$: vector representing concentrations of various nuclides at time t
- $\mathbb{A}$: the transmutation matrix, detailing decay rates and pathways between nuclides
- $\vec{N}_0$: the initial concentration vector

$$\frac{d\vec{N}(t)}{dt} = \mathbb{A}\vec{N}(t), \text{with } \vec{N}(t=0) = \vec{N}_0$$

PAUL SCHERRER INSTITUT PSI

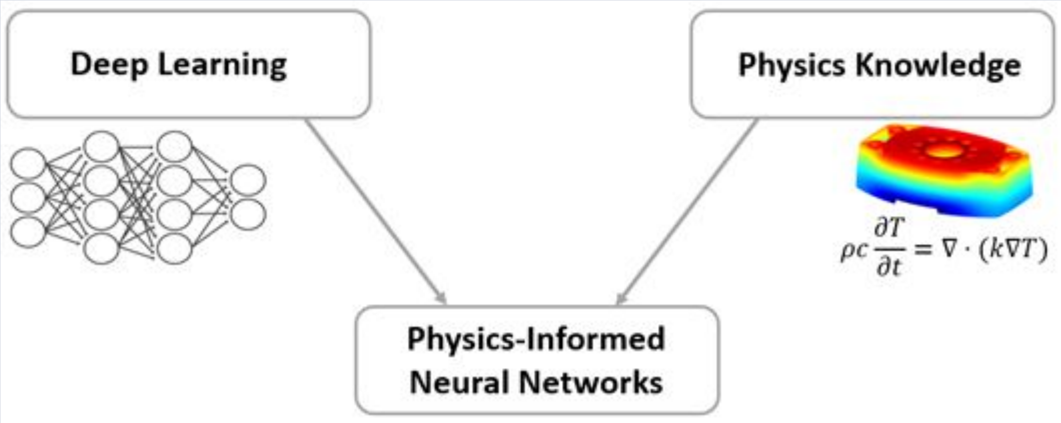
What are Neural Networks?

- Computational model inspired by the structure and functions of biological neural networks
- Each neuron receives input, processes it through mathematical functions, and passes the output to the next layer
- Can be much more accurate interpretations of complex patterns in data



What are Physics Informed Neural Networks (PINNs)?

- Class of neural networks
- Incorporate physical laws as differential equations
- Trained to satisfy both data and governing physical equations



PAUL SCHERRER INSTITUT PSI

How do PINNs solve the Bateman Equation?

(1) Define a neural network
(2) Compute the derivative wrt time + compute the residual of the Bateman equation at a set of time points
(3) Construct the Loss Function
(4) Train the neural network
(5) Evaluate and validate

PROBLEM

$$\frac{d\vec{N}(t)}{dt} = \mathbb{A}\vec{N}(t), \text{with } \vec{N}(t=0) = \vec{N}_0$$

$$\hat{N}(t;\theta)$$

$$\downarrow$$

$$\frac{d\hat{N}(t)}{dt}$$

$$\downarrow$$

$$\mathcal{R}(t_j) = \frac{d\hat{N}(t_j)}{dt} - \mathbb{A}\hat{N}(t_j)$$

$$\downarrow$$

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{phys}} + \lambda_{IC}\mathcal{L}_{IC} + \lambda_{\text{data}}\mathcal{L}_{\text{data}}$$

$$\downarrow$$

$$\textbf{minimize } \mathcal{L}_{\text{total}} \textbf{ to adjust } \theta$$

$$\downarrow$$

validate with analytical or numerical solutions

PAUL SCHERRER INSTITUT PSI

Real world example: $^{88}Y \rightarrow {}^{88}Sr$

$$\frac{dN_Y}{dt} = -\lambda_Y N_Y$$

$$\frac{dN_{Sr}}{dt} = \lambda_Y N_Y$$

NEED TO KNOW:

$$\lambda_Y = \frac{\ln(2)}{106.6} \approx 0.0065 \text{ days}^{-1}$$

STEP 1: Defining a neural network

Input: $t \in [0, T]$ (e.g. days)

Output: $\hat{N}_Y(t),\ \hat{N}_{Sr}(t)$

Architecture:

- Input: 1 neuron (time)
- Output: 2 neurons
- Hidden layers: 3 layers with 20 neurons each (for example)
- Activation: **tanh**

Real world example: $^{88}Y \rightarrow ^{88}Sr$

STEP 2: Defining Physics Residuals

Use automatic differentiation to compute derivatives from the network then define residuals:

$$\mathcal{R}_Y(t) = \frac{d\hat{N}_Y}{dt} + \lambda_Y \hat{N}_Y(t)$$

$$\mathcal{R}_{Sr}(t) = \frac{d\hat{N}_{Sr}}{dt} - \lambda_Y \hat{N}_Y(t)$$

Real world example: $^{88}Y \rightarrow ^{88} Sr$

PAUL SCHERRER INSTITUT

STEP 3: Defining Loss Function

$$\mathcal{L} = \mathrm{MSE}_{\mathrm{residual}} + \mathrm{MSE}_{\mathrm{initial}}$$

physics

data

start with 1
mole 88Y and
0 mole 88Sr

$$\mathrm{MSE}_{\mathrm{residual}} = \frac{1}{N}\sum_{i=1}^{N}\left(\mathcal{R}_Y(t_i)^2 + \mathcal{R}_{Sr}(t_i)^2\right)$$

$$\mathrm{MSE}_{\mathrm{initial}} = \left(\hat{N}_Y(0) - 1\right)^2 + \left(\hat{N}_{Sr}(0) - 0\right)^2$$

Real world example: $^{88}Y \rightarrow ^{88}Sr$

STEP 4: Training

(1) Learning the decay rate: the model learns the correct decay rate by minimizing error between the predicted decay and accumulation and the true data

(2) Loss function: the MSE between predictions and actual values guides the model in adjusting its parameters

(3) Backpropagation: gradients are used to update parameters through backpropagation, helping the model to minimize error

(4) Optimization: optimizers like AdamW helps in adjusting the model's parameters efficiently using adaptive learning rates.

Real world example: $^{88}Y \rightarrow ^{88} Sr$

PAUL SCHERRER INSTITUT PSI

STEP 5: Evaluating

After training, the neural network approximates:

$$\hat{N}_Y(t) \approx N_Y(t) = N_Y(0)e^{-\lambda_Y t}$$

$$\hat{N}_{Sr}(t) \approx N_{Sr}(t) = N_Y(0)(1 - e^{-\lambda_Y t})$$

Real world example: $^{88}Y \rightarrow ^{88} Sr$

PAUL SCHERRER INSTITUT PSI

References

1. Boiger, R., Pacifico, G., Alba, A., & Adelmann, A. (2024, February). Solving the Bateman Equation using Physics Informed Neural Networks. Presented at the PINN-PAD: Physics Informed Neural Networks in Padova, Italy. Retrieved from https://pinn-pad.dicea.unipd.it/contributed/02_boiger.pdf
2. "Neural Network (Machine Learning)." Wikipedia, Wikimedia Foundation, https://en.wikipedia.org/wiki/Neural_network_(machine_learning).