Neural Network Pipeline for Systems Biology: Solving the Notch Signaling Pathway

Alex Huang, Kartik Ramachandrula, Agniv Sarkar

Prof. Lu Lu (Yale University)

MIT PRIMES October Conference 2024

October 12, 2024

Scientific Machine Learning

1. Core component of AI and a computational technology that can be trained, with scientific data, to augment or automate human skills.

Scientific Machine Learning

1. Core component of AI and a computational technology that can be trained, with scientific data, to augment or automate human skills.
2. Draws tools from both machine learning and scientific computing.

Scientific Machine Learning

1. Core component of AI and a computational technology that can be trained, with scientific data, to augment or automate human skills.
2. Draws tools from both machine learning and scientific computing.
3. Foundations: Domain-aware, avoids over-training, robust.

Scientific Machine Learning

1. Core component of AI and a computational technology that can be trained, with scientific data, to augment or automate human skills.
2. Draws tools from both machine learning and scientific computing.
3. Foundations: Domain-aware, avoids over-training, robust.
4. Examples: *Physics-Informed Neural Networks (PINNs)*, Neural Ordinary Differential Equations (ODEs), CNNs for imaging, GNNs for detectors.

Primer on Neural Networks

1. Artificial NNs are inspired by the structure of biological NNs in animal brains.

Primer on Neural Networks

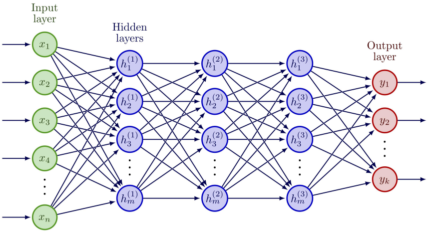
1. Artificial NNs are inspired by the structure of biological NNs in animal brains.
2. Connected nodes are called *artificial neurons*.

Primer on Neural Networks

1. Artificial NNs are inspired by the structure of biological NNs in animal brains.
2. Connected nodes are called *artificial neurons*.
3. Different layers perform different input transformations.

Primer on Neural Networks

1. Artificial NNs are inspired by the structure of biological NNs in animal brains.
2. Connected nodes are called *artificial neurons*.
3. Different layers perform different input transformations.
4. Signals travel from the input layer to the output layer through the hidden layers.

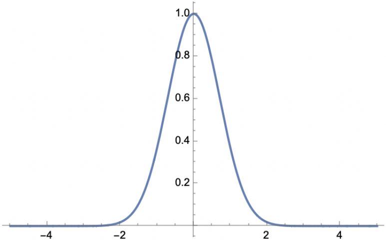


Applications of Neural Networks

1. Pattern recognition (radar systems, face identification, 3D reconstruction)
2. Function approximation (regression analysis, time series prediction)
3. Data and information visualization
4. Spam email filtering
5. Generative AI (including ChatGPT)

A Simple ODE

1. Solve for y as a function of x.
2. The equation is $\dot{y} = -2xy$.
3. The initial condition is $y(0) = 1$.
4. The analytic solution is $y(x) = e^{-x^2}$.

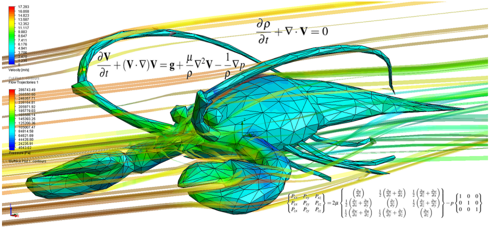


Navier-Stokes Equations

1. PDEs that describe the motion of viscous fluid substances.
2. Solution is a vector field in the form of flow viscosity.
3. Cauchy's momentum equation (convective form):

$$\rho \frac{\mathrm{Du}}{\mathrm{D}t} = -\nabla p + \nabla \cdot \tau + \rho \mathrm{a}$$

4. Navier-Stokes existence and smoothness is one of the Millennium Prize Problems.



Importance and Necessity of NNs

On October 8, Geoffrey Hinton and John Hopfield were awarded the Nobel Prize in Physics "for foundational discoveries and inventions that enable machine learning with artificial neural networks."

[1] Hornik, Stinchcombe, White, Neural Network, 1989 [1]

Importance and Necessity of NNs

On October 8, Geoffrey Hinton and John Hopfield were awarded the Nobel Prize in Physics "for foundational discoveries and inventions that enable machine learning with artificial neural networks."

1. Universal Approximation: Neural Networks can theoretically approximate functions to arbitrary accuracy [1]

[1] Hornik, Stinchcombe, White, Neural Network, 1989 [1]

Importance and Necessity of NNs

On October 8, Geoffrey Hinton and John Hopfield were awarded the Nobel Prize in Physics "for foundational discoveries and inventions that enable machine learning with artificial neural networks."

1. Universal Approximation: Neural Networks can theoretically approximate functions to arbitrary accuracy [1]
2. Curse of Dimensionality: Many variable problems are *very* computationally expensive.

[1] Hornik, Stinchcombe, White, Neural Network, 1989 [1]

Importance and Necessity of NNs

On October 8, Geoffrey Hinton and John Hopfield were awarded the Nobel Prize in Physics "for foundational discoveries and inventions that enable machine learning with artificial neural networks."

1. Universal Approximation: Neural Networks can theoretically approximate functions to arbitrary accuracy [1]
2. Curse of Dimensionality: Many variable problems are *very* computationally expensive.
3. Not Discretized: Does not struggle with inefficient meshes like FEM methods.

[1] Hornik, Stinchcombe, White, Neural Network, 1989 [1]

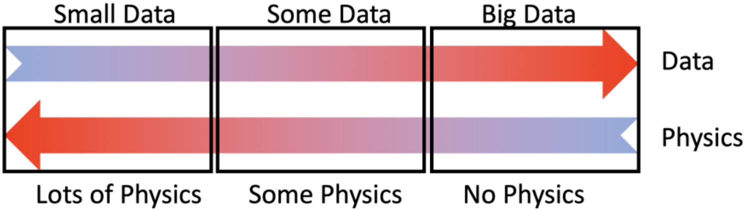
Importance and Necessity of NNs

On October 8, Geoffrey Hinton and John Hopfield were awarded the Nobel Prize in Physics "for foundational discoveries and inventions that enable machine learning with artificial neural networks."

1. Universal Approximation: Neural Networks can theoretically approximate functions to arbitrary accuracy [1]
2. Curse of Dimensionality: Many variable problems are *very* computationally expensive.
3. Not Discretized: Does not struggle with inefficient meshes like FEM methods.
4. Hybrid Models: Can combine both physics of the system as well as big data.

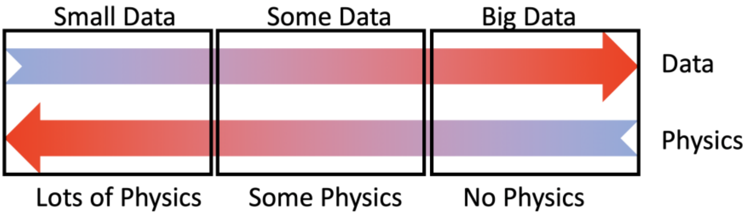
[1] Hornik, Stinchcombe, White, Neural Network, 1989 [1]

Data Driven Physics Discovery



[2] Karniadakis, Kevrekidis, Lu, et al., Nature Rev Phys, 2021 [2]

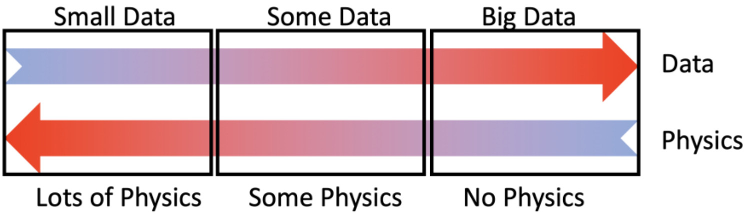
Data Driven Physics Discovery



1. Lots of Physics: Finite element methods.

[2] Karniadakis, Kevrekidis, Lu, et al., Nature Rev Phys, 2021 [2]

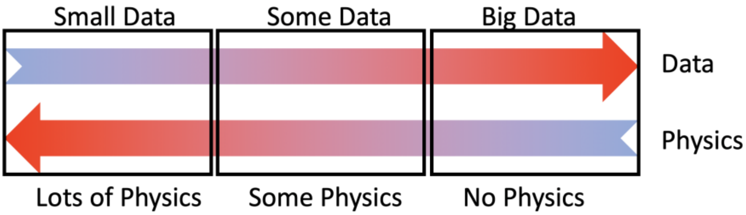
Data Driven Physics Discovery



1. Lots of Physics: Finite element methods.
2. Some Physics: **Physics-Informed Neural Networks**

[2] Karniadakis, Kevrekidis, Lu, et al., Nature Rev Phys, 2021 [2]

Data Driven Physics Discovery



1. Lots of Physics: Finite element methods.
2. Some Physics: **Physics-Informed Neural Networks**
3. No Physics: Operator and physics learning.[2]

[2] Karniadakis, Kevrekidis, Lu, et al., Nature Rev Phys, 2021 [2]

Inverse Physics-Informed Neural Networks Problem Setup

PINNs are powerful when there is **scarce data** and **not well-known** physics: Inverse Problems.

Inverse Physics-Informed Neural Networks Problem Setup

PINNs are powerful when there is **scarce data** and **not well-known** physics: Inverse Problems.

Goal: Discover the parameters of a system and make surrogate model.

Inverse Physics-Informed Neural Networks Problem Setup

Consider a PDE system with solution $\mathbf{u}$ over $\Omega \subset \mathbb{R}^d$:

$$\begin{aligned} \mathcal{F}[\mathbf{u}(\mathbf{x}); \gamma(\mathbf{x})] &= 0, & \mathbf{x} \in \Omega \\ \mathcal{B}[\mathbf{u}(\mathbf{x})] &= 0, & \mathbf{x} \in \partial\Omega \end{aligned}$$

Inverse Physics-Informed Neural Networks Problem Setup

Consider a PDE system with solution $\mathbf{u}$ over $\Omega \subset \mathbb{R}^d$:

$$\begin{aligned} \mathcal{F}[\mathbf{u}(\mathbf{x}); \gamma(\mathbf{x})] &= 0, && \mathbf{x} \in \Omega \\ \mathcal{B}[\mathbf{u}(\mathbf{x})] &= 0, && \mathbf{x} \in \partial\Omega \end{aligned}$$

Goal:

$$\boldsymbol{\theta}_u^*, \boldsymbol{\theta}_\gamma^* = \arg \min_{\boldsymbol{\theta}_u, \boldsymbol{\theta}_\gamma} \mathcal{L}(\boldsymbol{\theta}_u, \boldsymbol{\theta}_\gamma)$$

for loss function $\mathcal{L}$.

Inverse Physics-Informed Neural Network Model

1. Construct neural networks $\hat{u}(\mathsf{x}, \boldsymbol{\theta}_u), \hat{\gamma}(\mathsf{x}, \boldsymbol{\theta}_\gamma)$

Inverse Physics-Informed Neural Network Model

1. Construct neural networks $\hat{u}(\mathsf{x}, \boldsymbol{\theta}_u), \hat{\gamma}(\mathsf{x}, \boldsymbol{\theta}_\gamma)$
2. Define training sets of N_s, N_r, N_b for the PDE and IC/BC.

Inverse Physics-Informed Neural Network Model

1. Construct neural networks $\hat{u}(\mathbf{x}, \boldsymbol{\theta}_u), \hat{\gamma}(\mathbf{x}, \boldsymbol{\theta}_\gamma)$
2. Define training sets of N_s, N_r, N_b for the PDE and IC/BC.
3. Define loss functions:

$$\mathcal{L}_s(\theta) = \frac{1}{N_s} \sum_{i=1}^{N_s} (\hat{u}(\mathbf{x}_r^i) - u(\mathbf{x}_r^i))^2$$

$$\mathcal{L}_r(\theta, \lambda_r) = \frac{1}{N_r} \sum_{i=1}^{N_r} (\mathcal{F}[\hat{u}, \hat{\gamma}](\mathbf{x}_r^i))^2$$

$$\mathcal{L}_b(\theta, \lambda_b) = \frac{1}{N_b} \sum_{i=1}^{N_b} (\mathcal{B}[\hat{u}](\mathbf{x}_b^i))^2$$

Inverse Physics-Informed Neural Network Model

1. Construct neural networks $\hat{u}(\mathbf{x}, \boldsymbol{\theta}_u), \hat{\gamma}(\mathbf{x}, \boldsymbol{\theta}_\gamma)$
2. Define training sets of N_s, N_r, N_b for the PDE and IC/BC.
3. Define loss functions:

$$\mathcal{L}_s(\theta) = \frac{1}{N_s}\sum_{i=1}^{N_s}(\hat{u}(\mathbf{x}_r^i) - u(\mathbf{x}_r^i))^2$$

$$\mathcal{L}_r(\theta, \lambda_r) = \frac{1}{N_r}\sum_{i=1}^{N_r}(\mathcal{F}[\hat{u}, \hat{\gamma}](\mathbf{x}_r^i))^2$$

$$\mathcal{L}_b(\theta, \lambda_b) = \frac{1}{N_b}\sum_{i=1}^{N_b}(\mathcal{B}[\hat{u}](\mathbf{x}_b^i))^2$$

4. Total loss: $\mathcal{L}(\theta) = \mathcal{L}_s(\theta) + \lambda_r\mathcal{L}_r(\theta) + \lambda_b\mathcal{L}_b(\theta)$

Alphabet of PINNs

- Hard and Soft Constrained PINNs
- VPINN: Variational PINNs (and hp-VPINN: h, p refined variational PINNs)
- GPINN: Gradient-enhanced PINN
- CPINN: Conservative PINNs
- XPINN: Extended PINNs
- fPINN: PINNs for fractional PDEs
- sPINN: PINNs for stochastic DEs
- PI-GAN: Physics-informed Generative Adversarial Network

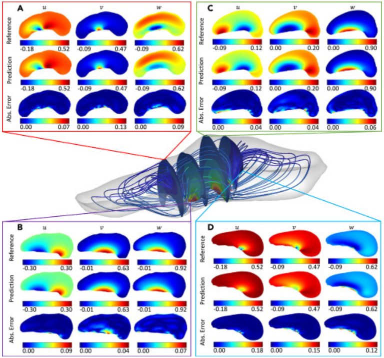
Applications of PINNs

1. Physics: Heat transfer, structural mechanics, electromagnetism, and fluid dynamics.[3]

[3] Daneker, Cai, Qian, et al., Nexus, 2024 [3]

Applications of PINNs

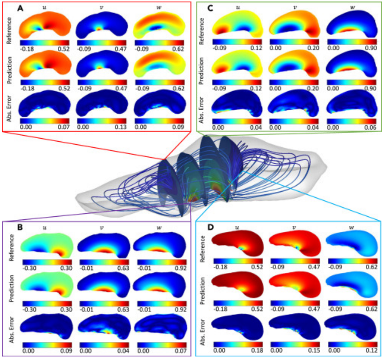
1. Physics: Heat transfer, structural mechanics, electromagnetism, and fluid dynamics.[3]



[3] Daneker, Cai, Qian, et al., Nexus, 2024 [3]

Applications of PINNs

1. Physics: Heat transfer, structural mechanics, electromagnetism, and fluid dynamics.[3]

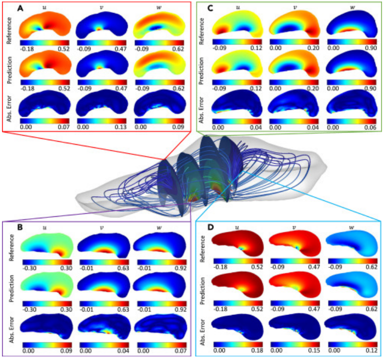


2. Financial Modeling, Epidemiology, Traffic Flow.

[3] Daneker, Cai, Qian, et al., Nexus, 2024 [3]

Applications of PINNs

1. Physics: Heat transfer, structural mechanics, electromagnetism, and fluid dynamics.[3]



2. Financial Modeling, Epidemiology, Traffic Flow.
3. **Systems-Biology**

[3] Daneker, Cai, Qian, et al., Nexus, 2024 [3]

Application of PINNs in Systems Biology

In Systems-Biology related problems, instead of dealing with *one* PDE, we instead have to deal with a system of coupled ODEs (though this can be extended to PDEs). Just like PINNs, we have the problems of scarce data and fuzzy measurements.

Goal: Create new ML pipeline using PINNs to discover parameters of a *biological* system to properly make conclusions about chemical pathways, disease, etc.

Application of PINNs in Systems Biology

In Systems-Biology related problems, instead of dealing with *one* PDE, we instead have to deal with a system of coupled ODEs (though this can be extended to PDEs). Just like PINNs, we have the problems of scarce data and fuzzy measurements.

Goal: Create new ML pipeline using PINNs to discover parameters of a *biological* system to properly make conclusions about chemical pathways, disease, etc.

Solution: Utilise... systems Biology-informed NNs! Or, sBINNs for short.

$$\mathcal{L}(\theta) = \mathcal{L}_S(\theta) + \sum_{\mathcal{F} \in \text{ODE Residuals}} \mathcal{L}_{\mathcal{F}}(\theta)$$

sBINNs specific problem

The system of ODEs that we focused on was specifically the Notch signalling pathway mentioned at the beginning. For the sake of clarity, the ODEs will be shown here. There are 22 state variables:

$$
\begin{aligned}
\frac{d[\mathrm{Dl4}_{\mathbf{c1}}]}{dt} &= -\left(k_f \cdot \mathrm{Dl4}_{c\mathbf{1}} \cdot \mathrm{Notch1}_{c\mathbf{2}} + k_r \cdot \mathrm{Dll1\_Notch1}_{c\mathbf{2}}\right) \\
&\quad -\left(k_f \cdot \mathrm{Dl4}_{c\mathbf{1}} \cdot \mathrm{Notch1}_{c\mathbf{1}} + k_r \cdot \mathrm{Dll4\_Notch1}_{c\mathbf{1}}\right) \\
\frac{d[\mathrm{Notch1}_{c\mathbf{1}}]}{dt} &= -\left(k_f \cdot \mathrm{Dl4}_{c\mathbf{2}} \cdot \mathrm{Notch1}_{c\mathbf{1}} + k_r \cdot \mathrm{Dll4\_Notch1}_{c\mathbf{1}}\right) \\
&\quad -\left(k_f \cdot \mathrm{Dl1}_{c\mathbf{2}} \cdot \mathrm{Notch1}_{c\mathbf{1}} + k_r \cdot \mathrm{Dll1\_Notch1}_{c\mathbf{1}}\right) \\
\frac{d[\mathrm{Dl4\_Notch1}_{c\mathbf{1}}]}{dt} &= \frac{G_s \cdot k_{\mathbf{cat}} \cdot \mathrm{Dl4}_{c\mathbf{1}} \cdot \mathrm{Notch1}_{c\mathbf{1}}}{K_m + \mathrm{Dl4\_Notch1}_{c\mathbf{1}}} - \frac{\mathrm{tetAhe} \cdot \mathrm{Hes1}_{c\mathbf{1}c\mathbf{2}}}{K_p + \mathrm{NICD}_{c\mathbf{1}c\mathbf{2}}} \\
\frac{d[\mathrm{NICD}_{c\mathbf{1}}]}{dt} &= -\frac{\mathrm{tetAhe} \cdot \mathrm{Hes1}_{c\mathbf{1}c\mathbf{2}}}{K_p + \mathrm{NICD}_{c\mathbf{1}c\mathbf{2}}} \\
\frac{d[\mathrm{Jagged1}_{c\mathbf{1}}]}{dt} &= -k_{\mathbf{deg\_Jag}} \cdot \mathrm{Jagged1}_{c\mathbf{1}} - \left(k_{\mathbf{on\_cis}} \cdot \mathrm{Jagged1}_{c\mathbf{1}} \cdot \mathrm{Notch1}_{c\mathbf{1}} + k_{\mathbf{on\_cis}} \cdot \mathrm{JagNotch}_{c\mathbf{1}}\right) \\
\frac{d[\mathrm{Jag\_Notch}_{c\mathbf{1}}]}{dt} &= k_{\mathbf{on\_cis}} \cdot \mathrm{Jagged1}_{c\mathbf{1}} \cdot \mathrm{Notch1}_{c\mathbf{1}} + k_{\mathbf{on\_cis}} \cdot \mathrm{JagNotch}_{c\mathbf{1}} \\
\frac{d[\mathrm{Notch\_jag2}_{c\mathbf{1}}]}{dt} &= k_f \cdot \mathrm{JagNotch\_jagc1}_{c\mathbf{1}} \cdot \mathrm{Notch1}_{c\mathbf{2}} + k_r \cdot \mathrm{JagNotch\_jagc1}_{c\mathbf{1}} - \left(k_f \cdot \mathrm{Dl1}_{c\mathbf{2}} \cdot \mathrm{Notch1}_{c\mathbf{1}} + \right. \\
d[\mathrm{pR2}_{c\mathbf{1}}]
\end{aligned}
$$

Utilization of Identifiability

So, because we have so many ODEs, we have to narrow down *what actually matters.*

Utilization of Identifiability

So, because we have so many ODEs, we have to narrow down *what actually matters.* In our case, we have:

Parameter	kf_{dllN}	kp_{R2}	kdp_{R2}	kr_{dllN}	Km	kcat	$kdeg_{\mathrm{NICD}}$	$kdeg_{\mathrm{Notch}}$	$kdeg_{\mathrm{Dll4}}$	kp_{Dll}	teta	$kdeg_{\mathrm{Hes1}}$	Kp_{Hes}
Identifiability	✓	✓	✓	✓	✓	×	✓	✓	✓	✓	✓	✓	✓
Parameter	tetaHe	kon_{cis}	$kdeg_{\mathrm{Jag}}$	kr_{jagNotch}	kr_{cis}	kf_{jagNotch}	Kp_{Jag}	tetaJag	$kdeg_{\mathrm{pR2}}$	$kdeg_{\mathrm{iR2}}$	Gs	$kform_{\mathrm{Notch}}$	kp
Identifiability	✓	✓	✓	✓	✓	✓	✓	✓	✓	×	×	✓	×

Table 1: Structural Identifiability of Notch model with STRIKE-GOLDD and StucturalIdentifiability.jl.

Utilization of Identifiability

So, because we have so many ODEs, we have to narrow down *what actually matters.* In our case, we have:

Parameter	kf_{dllN}	kp_{R2}	kdp_{R2}	kr_{dllN}	Km	kcat	$kdeg_{NICD}$	$kdeg_{Notch}$	$kdeg_{Dll4}$	kp_{Dll}	teta	$kdeg_{Hes1}$	Kp_{Hes}
Identifiability	✓	✓	✓	✓	✓	×	✓	✓	✓	✓	✓	✓	✓
Parameter	tetaHe	kon_{cis}	$kdeg_{Jag}$	$kr_{jagNotch}$	kr_{cis}	$kf_{jagNotch}$	Kp_{Jag}	tetaJag	$kdeg_{pR2}$	$kdeg_{iR2}$	Gs	$kform_{Notch}$	kp
Identifiability	✓	✓	✓	✓	✓	✓	✓	✓	✓	×	×	✓	×

Table 1: Structural Identifiability of Notch model with STRIKE-GOLDD and StucturalIdentifiability.jl.

Ultimately, we focused on $k_{f,\mathrm{DllN}}$, $k_{p,\mathrm{R2}}$, $k_{\mathrm{deg,NICD}}$, $k_{\mathrm{deg,Notch}}$, $k_{\mathrm{deg,Dll4}}$, θ, and $k_{\mathrm{deg,pR2}}$, with $k_{\mathrm{deg,NICD}}$ being the most sensitive.

Sensitivity Analysis

After selecting those 8 parameters, we run a standard OAT (one-at-a-time) sensitivity analysis to determine *how fast* each parameter should get trained in our inverse-SBINN setup.

Sensitivity Analysis

After selecting those 8 parameters, we run a standard OAT (one-at-a-time) sensitivity analysis to determine *how fast* each parameter should get trained in our inverse-SBINN setup.

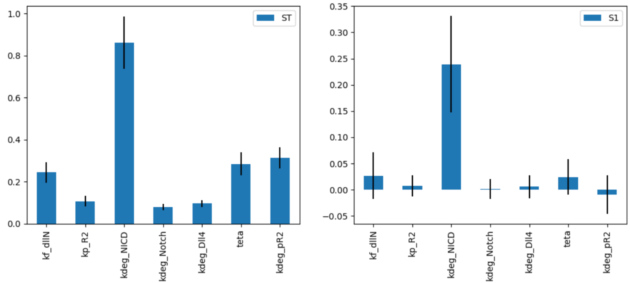


Figure 1: Sensitivity Analysis for the Notch ODE problem on parameters of interest.

Model Results

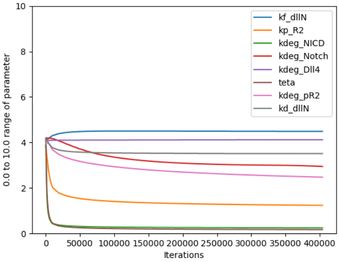


Figure 2: Parameters we tracked during the training of the final SBINN model.

Name	Value
kf_dllN	0.103
kp_R2	0.979
kdeg_NICD	1.84
kdeg_Notch	0.437
kdeg_Dll4	0.178
teta	2.03
kdeg_pR2	0.556
kd_dllN	0.307

Figure 3: Relative values attained after training.

Acknowledgements

- Mentor Prof. Lu Lu of Lu Group at Yale University.
  - For all of his hard work, mentoring, and letting us discuss our ideas, no matter how messy they started out.

Acknowledgements

- Mentor Prof. Lu Lu of Lu Group at Yale University.
  - For all of his hard work, mentoring, and letting us discuss our ideas, no matter how messy they started out.
- Mr. Mitchell Daneker and Ms. Rebecca Olivera.
  - For providing us with context on the Notch system, and understanding why this problem was relevant.

Acknowledgements

- Mentor Prof. Lu Lu of Lu Group at Yale University.
  - For all of his hard work, mentoring, and letting us discuss our ideas, no matter how messy they started out.
- Mr. Mitchell Daneker and Ms. Rebecca Olivera.
  - For providing us with context on the Notch system, and understanding why this problem was relevant.
- Dr. Tanya Khovanova, Dr. Slava Gerovitch, and the rest of the MIT PRIMES program.
  - Tanya, for her awesomeness and receptiveness.
  - Dr. Gerovitch, for his son. He's really cool!

Acknowledgements

- Mentor Prof. Lu Lu of Lu Group at Yale University.
  - For all of his hard work, mentoring, and letting us discuss our ideas, no matter how messy they started out.
- Mr. Mitchell Daneker and Ms. Rebecca Olivera.
  - For providing us with context on the Notch system, and understanding why this problem was relevant.
- Dr. Tanya Khovanova, Dr. Slava Gerovitch, and the rest of the MIT PRIMES program.
  - Tanya, for her awesomeness and receptiveness.
  - Dr. Gerovitch, for his son. He's really cool!
- Prof. Tyler Cowen of Emergent Ventures.
  - For a grant dedicated to our research, focused on compute.

Acknowledgements

- Mentor Prof. Lu Lu of Lu Group at Yale University.
  - For all of his hard work, mentoring, and letting us discuss our ideas, no matter how messy they started out.
- Mr. Mitchell Daneker and Ms. Rebecca Olivera.
  - For providing us with context on the Notch system, and understanding why this problem was relevant.
- Dr. Tanya Khovanova, Dr. Slava Gerovitch, and the rest of the MIT PRIMES program.
  - Tanya, for her awesomeness and receptiveness.
  - Dr. Gerovitch, for his son. He's really cool!
- Prof. Tyler Cowen of Emergent Ventures.
  - For a grant dedicated to our research, focused on compute.
- Parents, family, and friends for supporting us throughout our endeavors.
  - Without them, where would we be? (Not in Boston).

Thank You

Agniv Sarkar, Kartik Ramachandrula, Alex Huang

1. Introduction
2. PINNs
3. SBINNs
4. Conclusion
5. References

[1] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[2] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, "Physics-informed machine learning," *Nature Reviews Physics*, vol. 3, pp. 422–440, June 2021.

[3] M. Daneker, S. Cai, Y. Qian, E. Myzelev, A. Kumbhat, H. Li, and L. Lu, "Transfer learning on physics-informed neural networks for tracking the hemodynamics in the evolving false lumen of dissected aorta," *Nexus*, vol. 1, no. 2, p. 100016, 2024.

[4] Z.-J. Liu *et al.*, "Regulation of notch1 and dll4 by vascular endothelial growth factor in arterial endothelial cells: Implications for modulating arteriogenesis and angiogenesis," *Molecular and Cellular Biology*, vol. 23, no. 1, pp. 14–25, 2003. PMID: 12482957.