

# Embarque de Rede Neural Recorrente Fenomenologicamente Informada para Controle de Nível em Tanques Esféricos

**Silas Henrique Alves Araújo, Wildson Oliveira dos Santos, Leonardo Silva de Souza,  
Raony Maia Fontes, Márcio André Fernandes Martins**

PRH 35.1, Escola Politécnica da Universidade Federal da Bahia (UFBA)



## Objetivos

- Explorar o uso de Redes Neurais Recorrentes Fenomenologicamente Informadas (PIRNNs) para implementação de um analisador virtual.
- Utilizar esse analisador virtual para manter o controle PI embarcado de um sistema dinâmico.
- Embarcar a PIRNN proposta em um microcontrolador de baixo custo, testando o comportamento em um esquema *hardware-in-the-loop* (HIL).

## Relevância

- Velocidade é essencial em aplicações que exigem alta velocidade e respostas em tempo real.
- Hardware de baixo custo torna a tecnologia viável para mais aplicações, inclusive em contextos industriais e acadêmicos.

# Visão Geral: ANNs, RNNs, PINNs e PIRNNs

- **Redes Neurais Artificiais (ANNs)** são modelos computacionais inspirados no cérebro humano, capazes de aprender padrões a partir de dados.
- **Redes Neurais Fenomenologicamente Informadas (PINNs)** incorporam conhecimento físico ao treinamento da rede neural.
- **Redes Neurais Recorrentes (RNNs)** estendem as ANNs com “memória”, sendo ideais para sequências temporais.
- **Redes Neurais Recorrentes Fenomenologicamente Informadas (PIRNNs)** combinam as vantagens das RNNs e das PINNs, sendo mais adequadas para sistemas dinâmicos.

Introduzidas por Raissi et al. em 2017, as **PINNs** são uma metodologia que integra o modelo fenomenológico do problema ao treinamento das **ANNs**. Elas têm o potencial de:

- Acelerar simulações em relação aos métodos numéricos tradicionais.
- Melhorar a qualidade das previsões em comparação com ANNs convencionais.

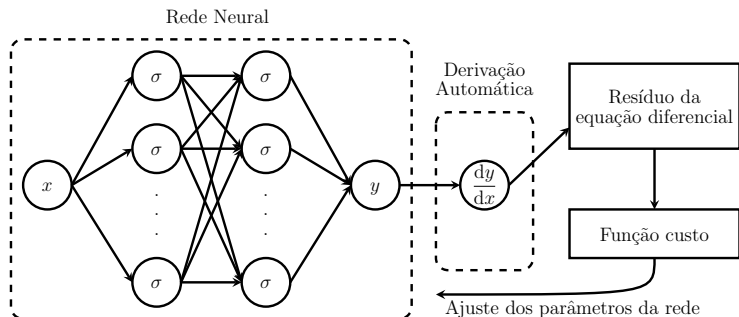
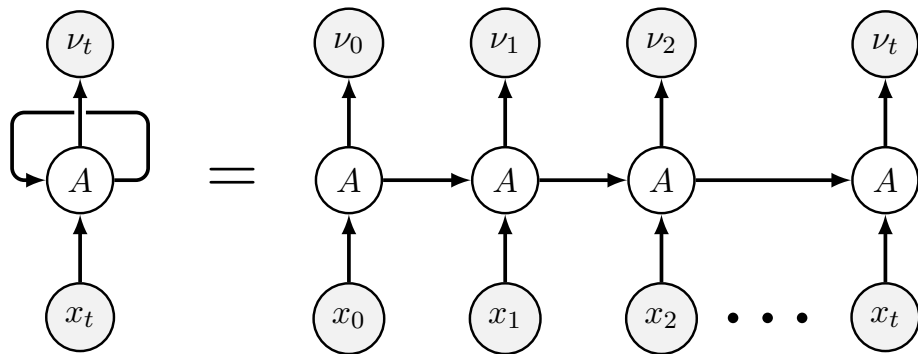


Figura 1: Representação esquemática de uma PINN.

As **RNNs** são redes que processam dados sequenciais (como séries temporais ou texto), mantendo o chamado *hidden state* ( $\nu_t$ ) que carrega informações do passado.



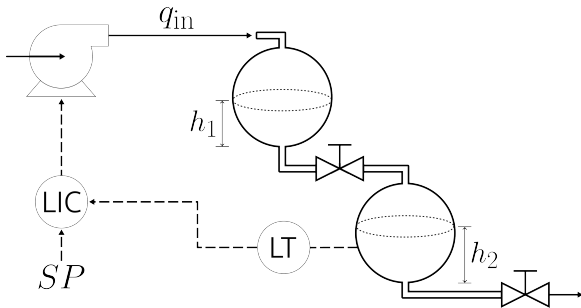
**Figura 2:** Representação de uma rede neural recorrente (RNN) nas formas *folded* e *unfolded*. Adaptado de Stack Exchange (2019)

Esse *hidden state* é usado para alimentar o mesmo neurônio com as informações dos passos anteriores. Repetindo o processo até obter o estado oculto final (Ansel et al. 2024).

$$\nu_t = \sigma(x_t \cdot W_{i\nu}^T + b_{i\nu} + \nu_{t-1} \cdot W_{\nu\nu}^T + b_{\nu\nu})$$

Já as **PIRNNs** são uma variação das PINNs que incorpora memória recorrente e permite uma modelagem mais precisa de sistemas dinâmicos (Zheng et al. 2023).

# Problema Exemplo



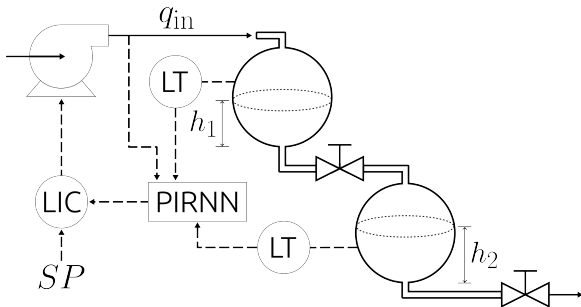
Sistema de equações diferenciais dos tanques

$$\frac{dh_1(t)}{dt} = \frac{q_{in} - \alpha_1 s_1 \sqrt{2gh_1}}{\pi(2Rh_1 - h_1^2)} \quad (1a)$$

$$\frac{dh_2(t)}{dt} = \frac{\alpha_1 s_1 \sqrt{2gh_1} - \alpha_2 s_2 \sqrt{2gh_2}}{\pi(2Rh_2 - h_2^2)} \quad (1b)$$

Figura 3: Representação esquemática do sistema.

# Problema Exemplo



Sistema de equações diferenciais dos tanques

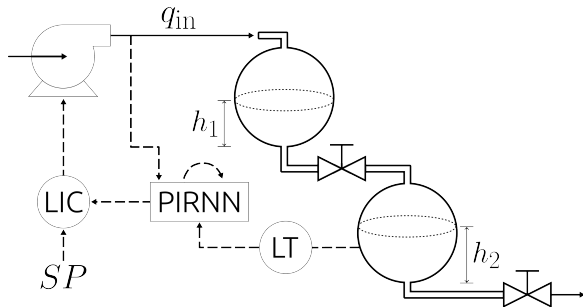
$$\frac{dh_1(t)}{dt} = \frac{q_{in} - \alpha_1 s_1 \sqrt{2gh_1}}{\pi(2Rh_1 - h_1^2)} \quad (1a)$$

$$\frac{dh_2(t)}{dt} = \frac{\alpha_1 s_1 \sqrt{2gh_1} - \alpha_2 s_2 \sqrt{2gh_2}}{\pi(2Rh_2 - h_2^2)} \quad (1b)$$

Figura 3: Representação esquemática do sistema.



# Problema Exemplo



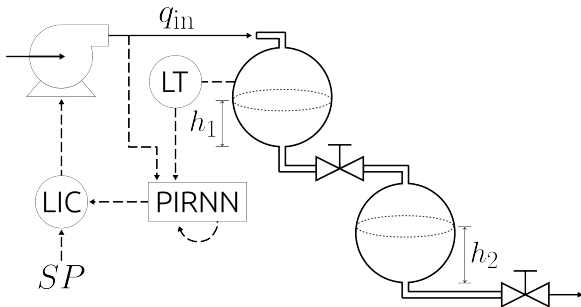
Sistema de equações diferenciais dos tanques

$$\frac{dh_1(t)}{dt} = \frac{q_{in} - \alpha_1 s_1 \sqrt{2gh_1}}{\pi(2Rh_1 - h_1^2)} \quad (1a)$$

$$\frac{dh_2(t)}{dt} = \frac{\alpha_1 s_1 \sqrt{2gh_1} - \alpha_2 s_2 \sqrt{2gh_2}}{\pi(2Rh_2 - h_2^2)} \quad (1b)$$

Figura 3: Representação esquemática do sistema.

# Problema Exemplo



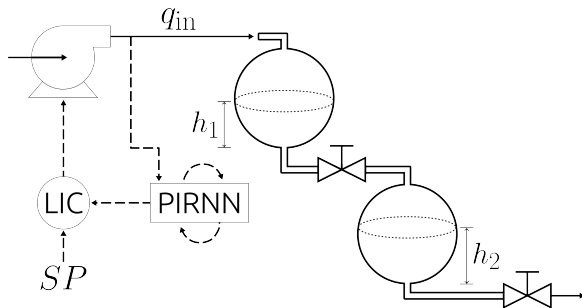
Sistema de equações diferenciais dos tanques

$$\frac{dh_1(t)}{dt} = \frac{q_{in} - \alpha_1 s_1 \sqrt{2gh_1}}{\pi(2Rh_1 - h_1^2)} \quad (1a)$$

$$\frac{dh_2(t)}{dt} = \frac{\alpha_1 s_1 \sqrt{2gh_1} - \alpha_2 s_2 \sqrt{2gh_2}}{\pi(2Rh_2 - h_2^2)} \quad (1b)$$

Figura 3: Representação esquemática do sistema.

# Problema Exemplo



Sistema de equações diferenciais dos tanques

$$\frac{dh_1(t)}{dt} = \frac{q_{in} - \alpha_1 s_1 \sqrt{2gh_1}}{\pi(2Rh_1 - h_1^2)} \quad (1a)$$

$$\frac{dh_2(t)}{dt} = \frac{\alpha_1 s_1 \sqrt{2gh_1} - \alpha_2 s_2 \sqrt{2gh_2}}{\pi(2Rh_2 - h_2^2)} \quad (1b)$$

Figura 3: Representação esquemática do sistema.

**Tabela 1:** Constantes do sistema de equações diferenciais formado por (1a) e (1b).

Símbolo	Descrição	Valor
$\alpha_1$	Coeficiente de fluxo da válvula 1	0,56
$\alpha_2$	Coeficiente de fluxo da válvula 2	0,30
$s_1$	Área da seção de saída do tanque 1	0,50 cm <sup>2</sup>
$s_2$	Área da seção de saída do tanque 2	0,50 cm <sup>2</sup>
$g$	Aceleração da gravidade	980,665 cm/s <sup>2</sup>
$R$	Raio dos tanques	14,85 cm

## Estrutura da PIRNN utilizada

A PIRNN utilizada consiste em 4 camadas, todas com a função de ativação tangente hiperbólica ( $\sigma$ ), com exceção da saída, que não possui função de ativação.

No total, a rede possui  $32 + 32 + 32 + 2 = 98$  neurônios.

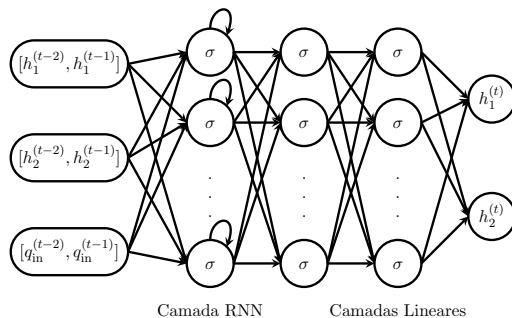


Figura 4: Diagrama da PIRNN utilizada.

## A função custo utilizada

A função custo combina o erro das equações diferenciais físicas com o erro dos dados simulados.

Entretanto, agora não podemos usar a derivação automática, pois o tempo ( $t$ ) não é uma entrada. Aqui, foi utilizada a derivada regressiva de 3 pontos (os 2 pontos fornecidos para a rede e o ponto previsto).

$$\mathcal{L} = w_1 \cdot \mathcal{L}_{\text{EDO}} + w_2 \cdot \mathcal{L}_{\text{data}} \quad (2)$$

onde:

- $w_1, w_2$ : pesos para o cálculo da função custo;
- $\mathcal{L}_{\text{EDO}}$ : erro das equações diferenciais;
- $\mathcal{L}_{\text{data}}$ : erro dos dados;
- $\mathcal{L}$ : erro total.

# Resultados

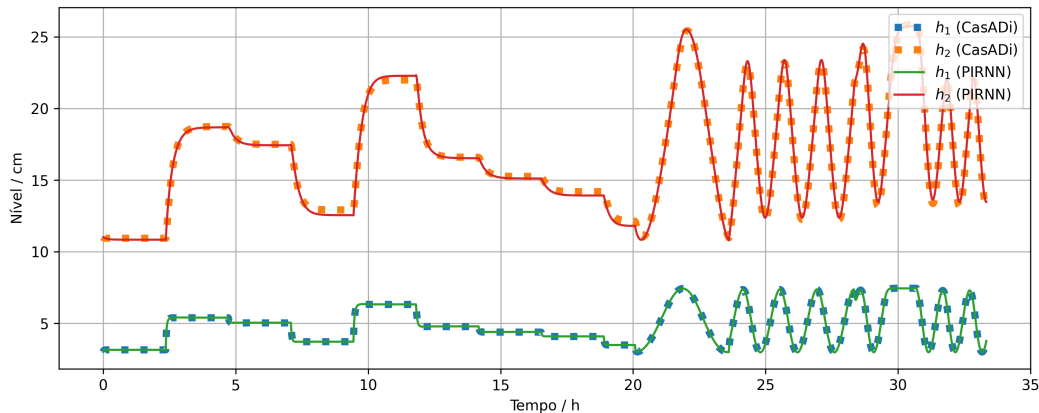
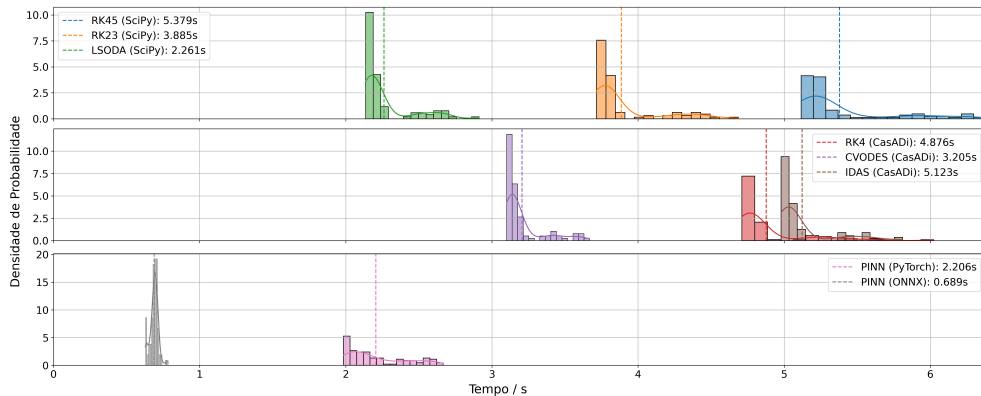


Figura 5: Comparação entre as previsões da PIRNN e o método numérico (RK)

# Comparação de Tempos de Execução



**Figura 6:** Densidade de probabilidade dos tempos de execução dos métodos avaliados. Cada método foi executado 100 vezes; as linhas tracejadas indicam os tempos médios.



# Por que o ONNX Runtime é tão rápido?

O ONNX Runtime é focado em desempenho de inferência e possui uma série de otimizações:

- “constant folding”;
- “redundant node eliminations”;
- “node fusions”.

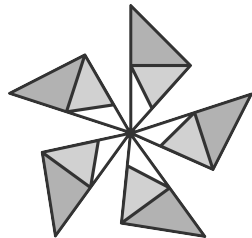


Figura 7: Logo do ONNX Runtime. Fonte: [onnxruntime.ai](https://onnxruntime.ai) (2017)

# O que é o Arduino?

O Arduino é uma plataforma prototipagem amplamente utilizada por programadores devido ao seu baixo custo e facilidade para desenvolver (Hughes 2016). Entre as placas mais populares, destaca-se o Arduino UNO, a escolhida para esse trabalho.

Memória Flash	Memória RAM
32 kilobytes	2 kilobytes

Tabela 2: Memória disponível no Arduino UNO.



Figura 8: Placa Arduino UNO. Fonte: makerhero.com

# Ciclo de Comunicação e Execução dos Componentes do Sistema

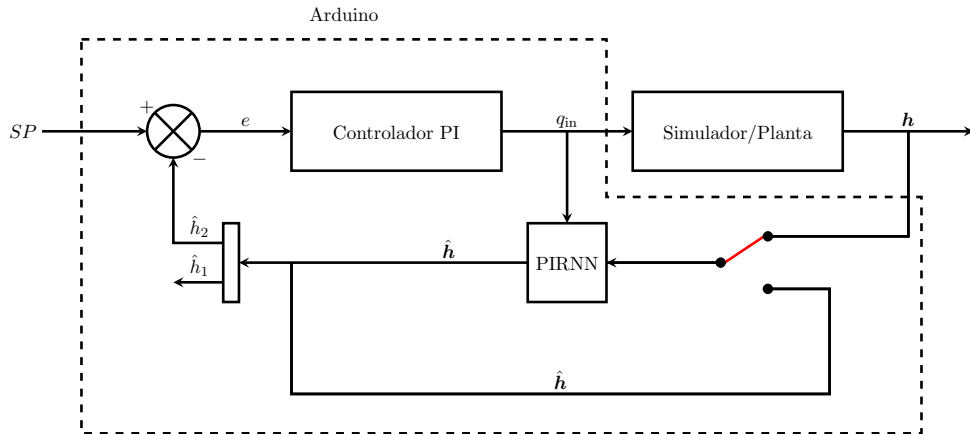


Figura 9: Fluxograma da comunicação entre os componentes do sistema.

# Ciclo de Comunicação e Execução dos Componentes do Sistema

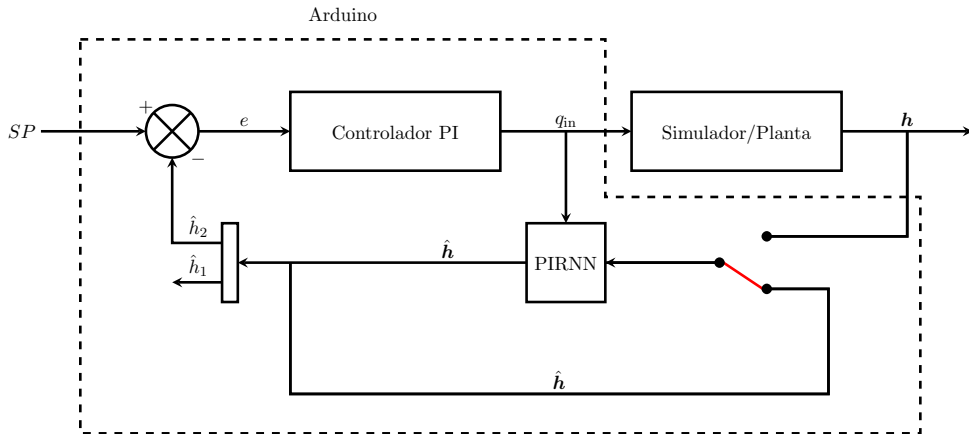


Figura 9: Fluxograma da comunicação entre os componentes do sistema.

# O *TankSim*



O software *TankSim* foi desenvolvido para:

- simular e manipular o funcionamento de uma planta;
- realizar a comunicação com o Arduino;
- visualizar os valores previstos pela rede neural;

Figura 10: Captura de tela do *TankSim*.

# Resultados

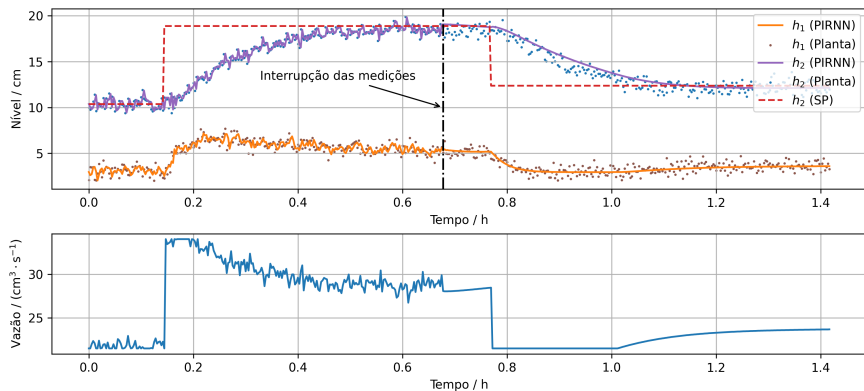


Figura 11: Vazão de entrada  $q_{in}$  definida pelo controlador PI e nível dos tanques  $h_1$  e  $h_2$  previstos pelo simulador e pela PIRNN.

# Considerações finais

- Usando o *onnxruntime*, a PIRNN foi mais de três vezes mais rápida que os métodos numéricos tradicionais no mesmo computador.
- O Arduino demonstrou capacidade para executar PIRNNs, com ampla margem de recursos (**69,1%** da memória flash e apenas **15,9%** da memória RAM foram utilizados).
- Mesmo com ruído ou ausência de medições, a PIRNN manteve previsões de boa qualidade, demonstrando sua robustez.
- O controlador PI manteve o controle da planta recebendo apenas os valores previstos pela PIRNN, garantindo operação contínua mesmo sem medições diretas.





# Agradecimentos

Agradeço à Agência Nacional do Petróleo, Gás Natural e Biocombustíveis (ANP), no âmbito do PRH 35.1, pelo suporte financeiro e apoio ao desenvolvimento deste trabalho.





# Bibliografia I

-  Ansel, Jason et al. (2024). “PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation”. *Em: Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2. ASPLOS '24. La Jolla, CA, USA: Association for Computing Machinery*, pp. 929–947. ISBN: 9798400703850.
-  Hughes, J M (mai. de 2016). *Arduino: A Technical Reference*. “O’Reilly Media, Inc.” ISBN: 9781491934500.
-  Raissi, Maziar et al. (2017). “Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations”. *Em: arXiv preprint arXiv:1711.10561*.
-  Zheng, Yingzhe et al. (2023). “Physics-informed recurrent neural network modeling for predictive control of nonlinear processes”. *Em: Journal of Process Control* 128, p. 103005. ISSN: 0959-1524.