Physics-Informed Neural Networks for Power Systems Applications

Federica Bragone

Kateryna Morozovska, Stefano Markidis, Tor Laneryd, Michele Luvisotto, Oliver Odeback, Khaoula Oueslati

KTH Royal Institute of Technology

PINN-PAD

Agenda

- **Introduction to PINNs**
  - *Solution of PDEs*
  - *ANNs as PDE Solvers*
  - *ANNs vs. Physics-Informed NNs*
  - *Physics-Informed Neural Networks (PINNs)*
- **Case Studies**
  - *Electric Power Systems*
  - *Insulation System of Power Transformers*
  - *PINNs for Power Systems Components*
  - *Case Study I: Transformer Thermal Model*
  - *Case Study II: Transformer Cellulose Degradation*
- **Conclusions**

Introduction to PINNs

Solution of Partial Differential Equations (PDEs)

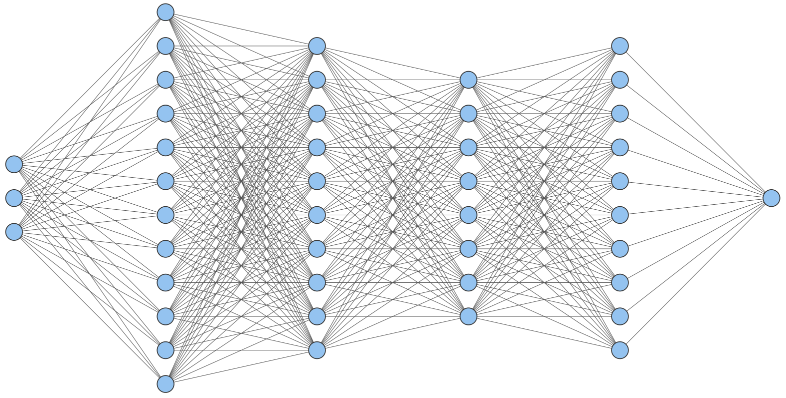
Analytical Solution

- Transparent and efficient solution.
- A few PDEs can be solved exactly.

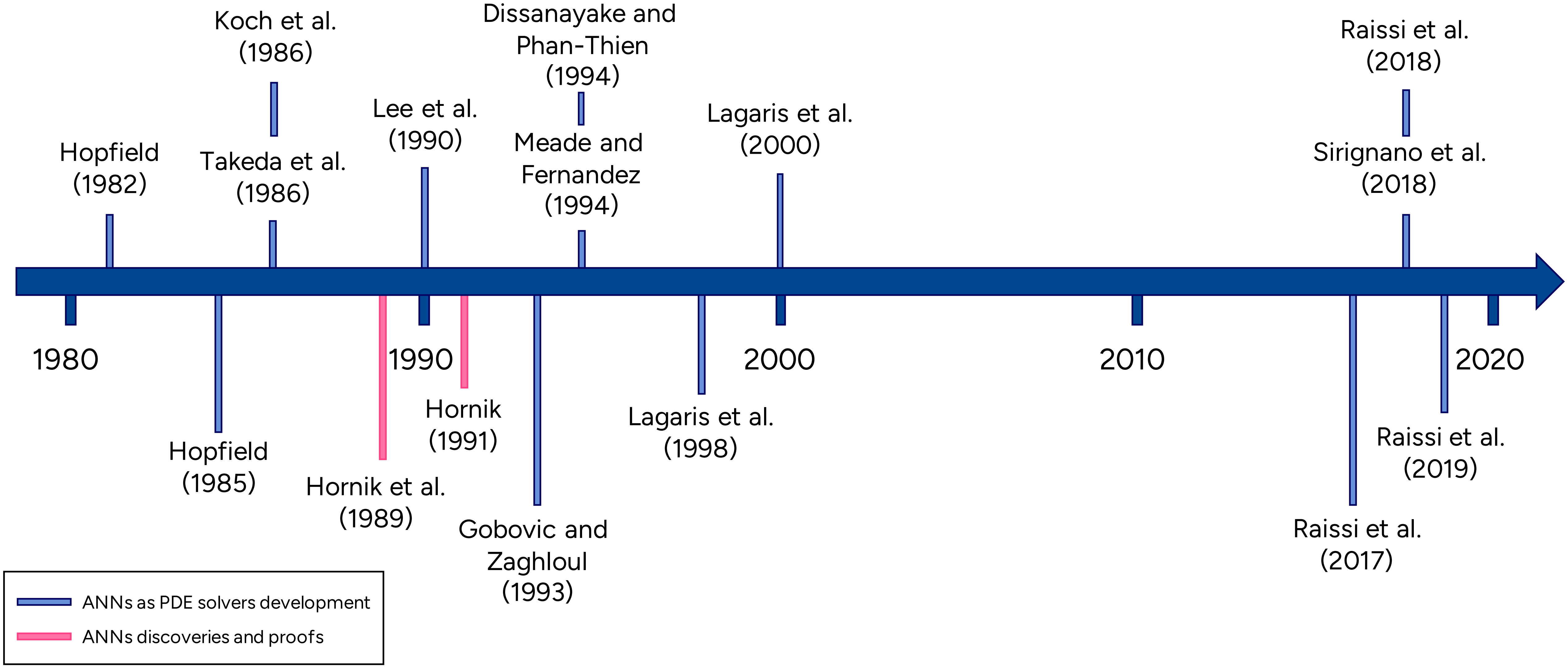
Numerical Solution

- Approximation of the solution when it cannot be solved analytically.
- Common numerical methods:
  - Finite Difference Method (FDM)
  - Finite Volume Method (FVM)
  - Finite Element Method (FEM)

Artificial Neural Networks to Solve PDEs



ANNs as PDE Solvers



ANNs vs. Physics-Informed NNs

Traditional ANNs

- Data-driven models.
- Use of large quantities of data to make accurate predictions.
- The training requires either analytical or numerical solutions (supervised learning).
- Complex and deep architectures.
- The solver does not use grids.

Physics-Informed NNs

- Use information stored in PDEs and ODEs, adding a part of the network to calculate the residual.
- No need for prior solutions of the equation (unsupervised learning).
- Good approximations both with large and small datasets.
- Not necessarily complex structures for the network.
- Use automatic differentiation to calculate the derivatives of the network.

Physics-Informed Neural Networks (PINNs)

Data-driven solutions to PDEs

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

- $u(x,t)$ is the latent hidden solution;
- $\mathcal{D}[\cdot]$ is a nonlinear differential operator;
- The domain Ω is a subset of $\mathbb{R}^d$.

Forward problem

Given some fixed parameters λ, what is the solution $u(x,t)$?

Data-driven discovery of PDEs

$$u_t + \mathcal{D}[u; \lambda] = 0,\ x \in \Omega,\ t \in [0, n]$$

- $u(x,t)$ is the latent hidden solution;
- $\mathcal{D}[\cdot; \lambda]$ is a nonlinear differential operator parametrized by λ;
- The domain Ω is a subset of $\mathbb{R}^d$.

Inverse problem

What are the best parameters λ of the PDE that describe the data?

[1] M. Raissi, P. Perdikaris, G.E. Karniadakis, "*Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations*".

Physics-Informed Neural Networks (PINNs)

Data-driven solutions to PDEs

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

Set the function:

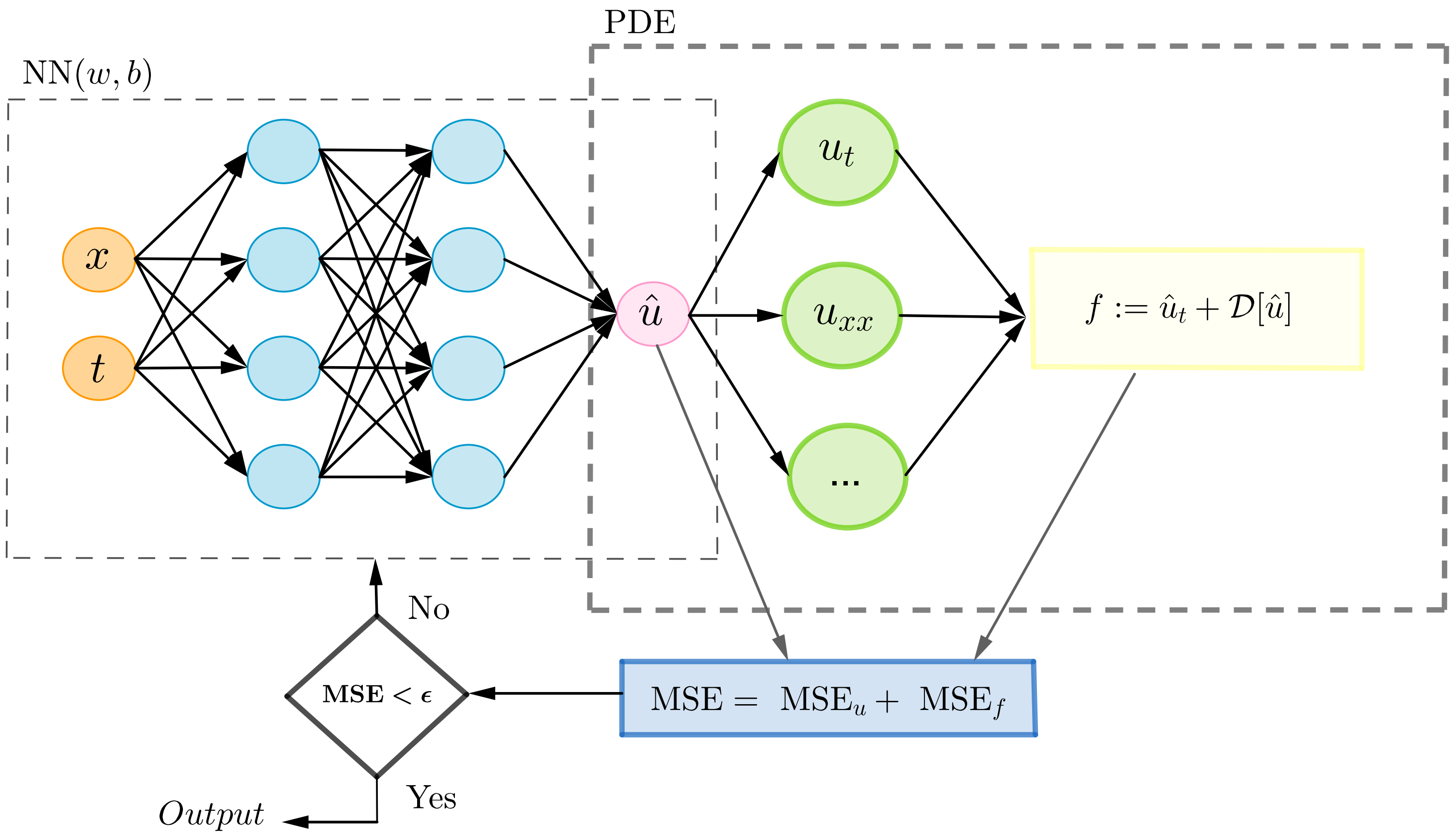
$$f \coloneqq u_t + \mathcal{D}[u]$$

Minimize the mean squared error loss:

$$MSE = MSE_u + MSE_f$$

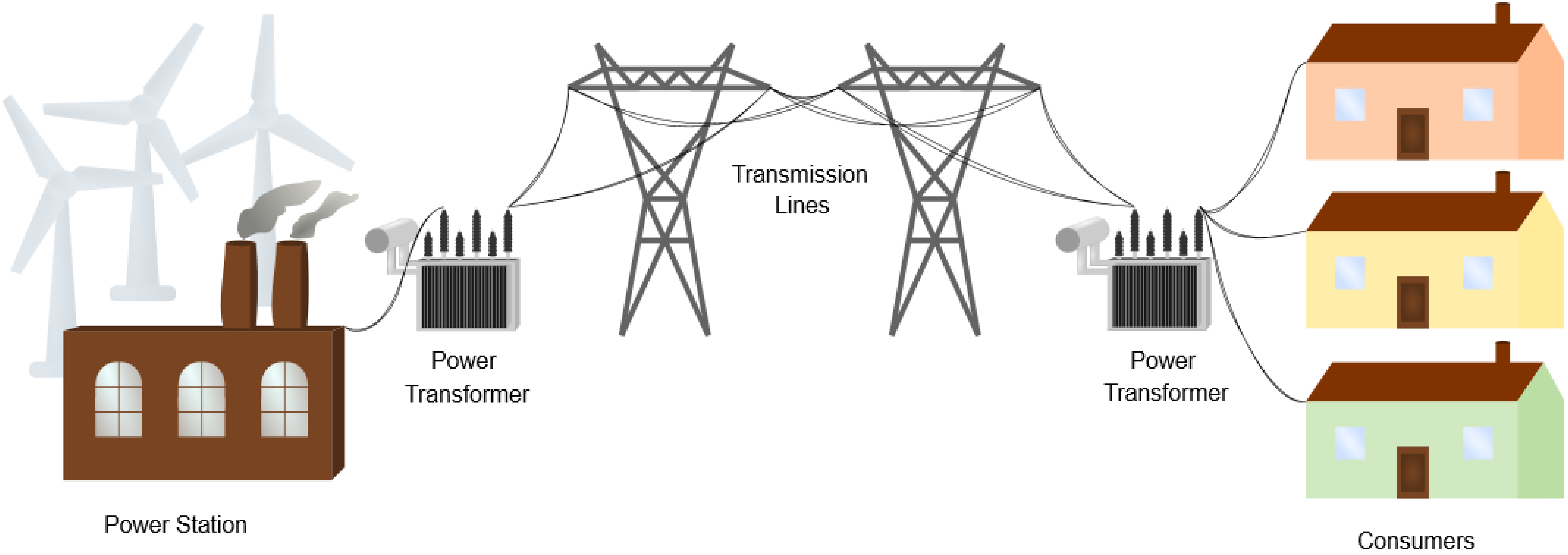
$$MSE_u = \frac{1}{N_u} \sum_{i=1}^{N_u} \left| \hat{u}\left(x_u^i, t_u^i\right) - u^i \right|^2$$

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



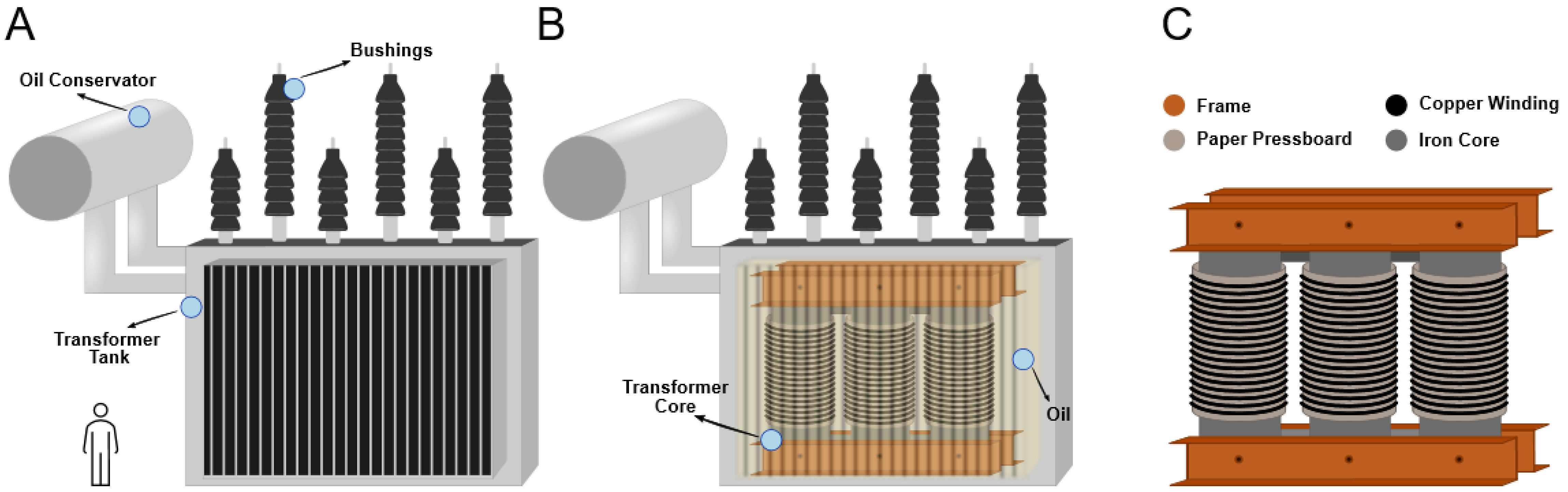
Case Studies

Electric Power Systems



- The power grid is an example of electric power system.
- It is a system that provides electricity from the producers to the consumers.
- It consists of power stations, power transformers and transmission.

Insulation System of Power Transformers



A: Three-phase, core-type power transformer.

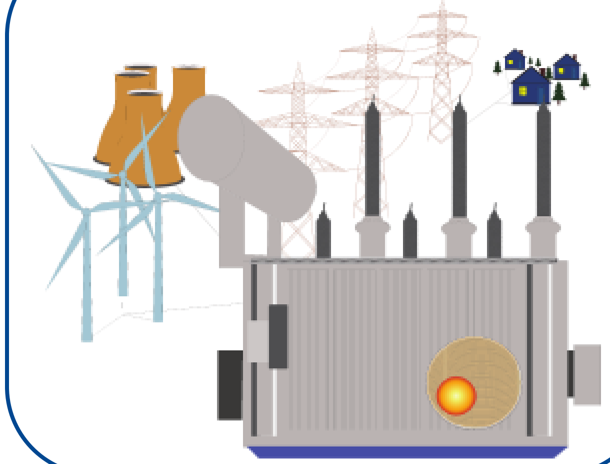
B: Transformer insulation oil and core.

C: Transformer core with corresponding insulation paper.

PINNs for Power Systems Components

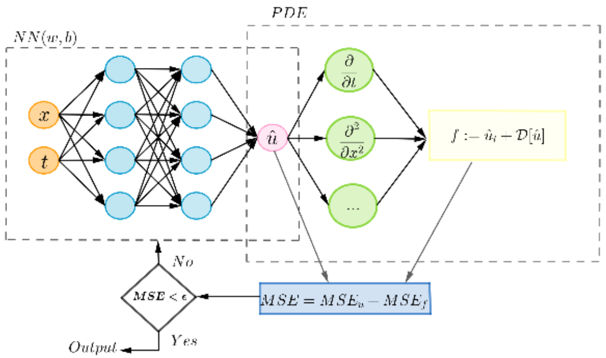
Data

Power components



Type of data: Top-oil temperature, ambient temperature, load factor, degree of polymerization

Physics-Informed Neural Networks



Models: Solution of PDE, Discovery of PDE

Maintenance

Transformer Thermal Model

[1] F. Bragone, K. Morozovska, T. Laneryd, M. Luvisotto, and P. Hilber, *"Physics-informed neural networks for modelling power transformer's dynamic thermal behaviour"*, Electric power systems research, vol. 211, p. 108447, 2022.

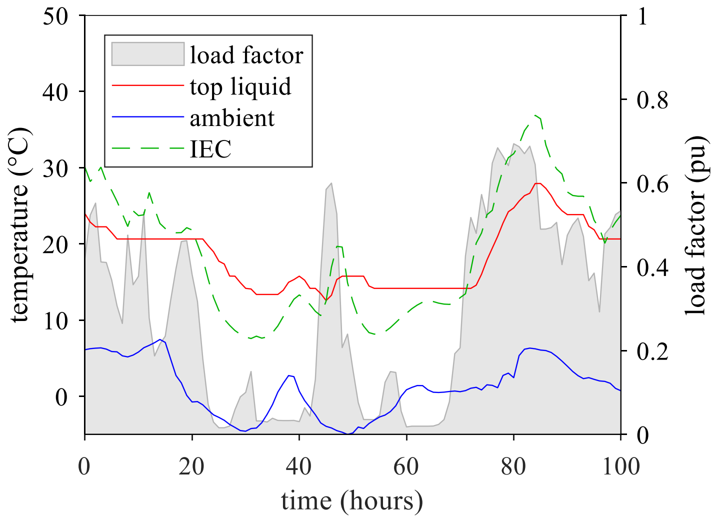
[2] T. Laneryd, F. Bragone, K. Morozovska, and M. Luvisotto, *"Physics informed neural networks for power transformer dynamic thermal modelling"*, in 10th Vienna International Conference on Mathematical Modelling, pp. 1–6, 2022.

[3] O. W. Odeback, F. Bragone, T. Laneryd, M. Luvisotto, and K. Morozovska, *"Physics-Informed Neural Networks for prediction of transformer's temperature distribution"*, in IEEE ICMLA 2022.

Problem

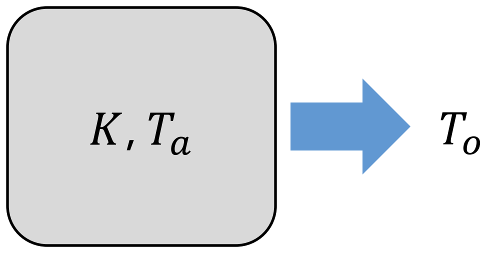
Background

- An indicator of transformer thermal performance is the top oil temperature T_o.
- Top oil temperature is a function of:
  - Ambient temperature T_a,
  - Load factor K.



Conventional dynamic thermal modelling

- Parameters for rated conditions are determined empirically during the factory test.
- Effects of K and T_a are included in the model.
- Model does not conserve energy.
- Model does not provide any temperature distribution.



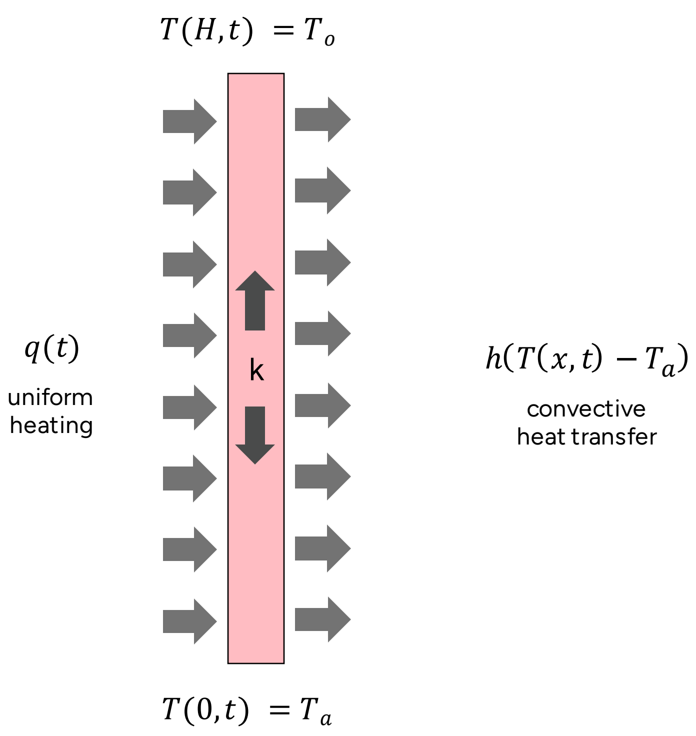
Transformer Thermal Modelling

$$\frac{\partial^2 T}{\partial x^2} + \frac{q}{k} = \frac{1}{\alpha}\frac{\partial T}{\partial t}$$

Where: $q = q(x,t) = P_0 + P_K(t) - h\big(T(x,t) - T_a(t)\big)$

Boundary conditions:

$$\mathrm{T}(0,t) = T_a, \quad T(H,t) = T_o$$



Known values:

$P_K \sim K^2 \cdot \mu$: load loss
μ : rated load loss
P_0 : no-load loss
h : heat transfer coefficient
H : height

Data:

T_a : ambient temperature
K : load factor
T_o : top-oil temperature

Model

Heat diffusion equation in 1D

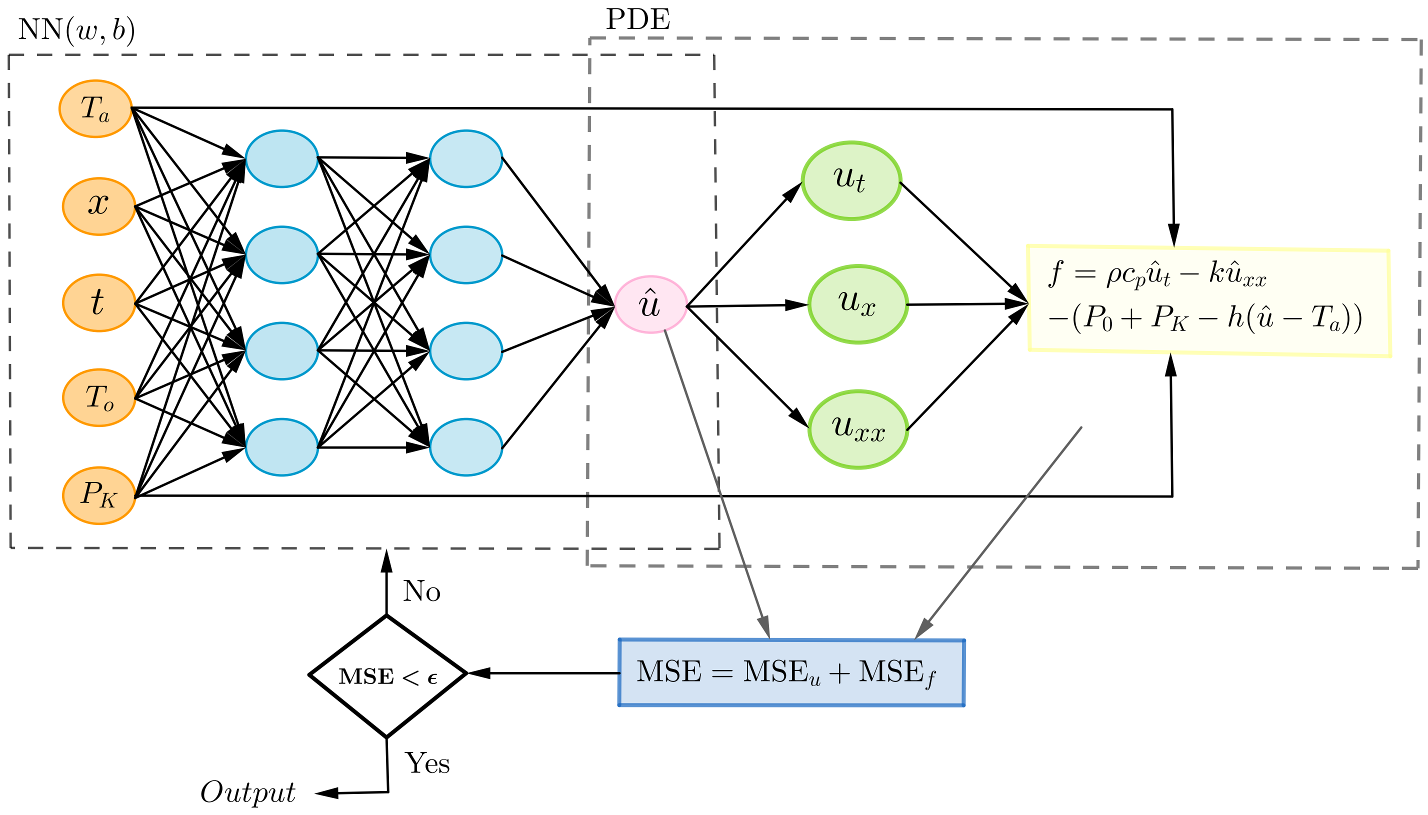
$$\rho c_p u_t - k u_{xx} - \big(P_0 + P_K - h(u - T_a)\big) = 0$$

$$u(0,t) = T_a \quad u(H,t) = T_o$$

Set the function:

$$f := \rho c_p u_t - k u_{xx} - \big(P_0 + P_K - h(u - T_a)\big)$$

- u is the temperature (K),
- k is the effective thermal conductivity $(W/m \cdot K)$,
- c_p is the specific heat capacity $(J/kg \cdot K)$,
- ρ is the density (kg/m^3),
- P_0 and P_K are losses,
- h is the heat transfer coefficient $(W/m^2 \cdot K)$,
- T_a is the ambient temperature (K),
- T_o is the top-oil temperature (K).



Model Structure

Hyperparameter Tuning

- Data: t, T_a, K , T_o
- Neural network hyperparameters:
  - Hidden layers: 2, 4, 6
  - Neurons: 10, 20, 50, 100
  - Number of boundary training data N_u: 50, 100, 150, 200
  - Number of collocation points N_f : 2000, 5000, 10000
  - Activation function: tanh
  - Optimizer: L-BFGS-B

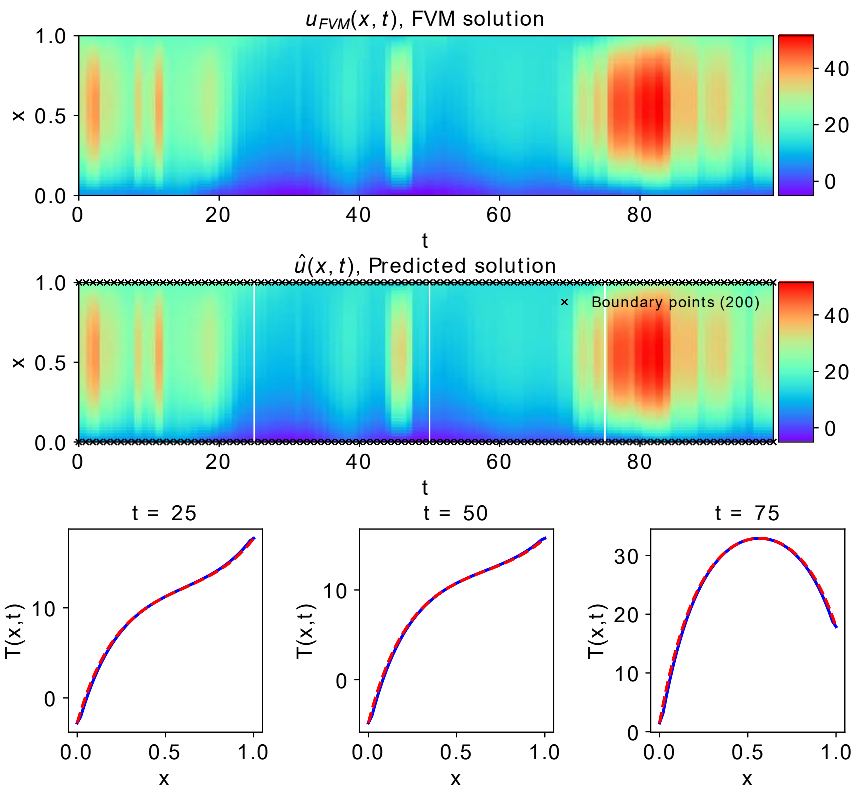
Structure Test Data

Test data to predict the model:

- Finite Volume Method (FVM)
- 50 data points for space x
- 100 data points for time t

Results

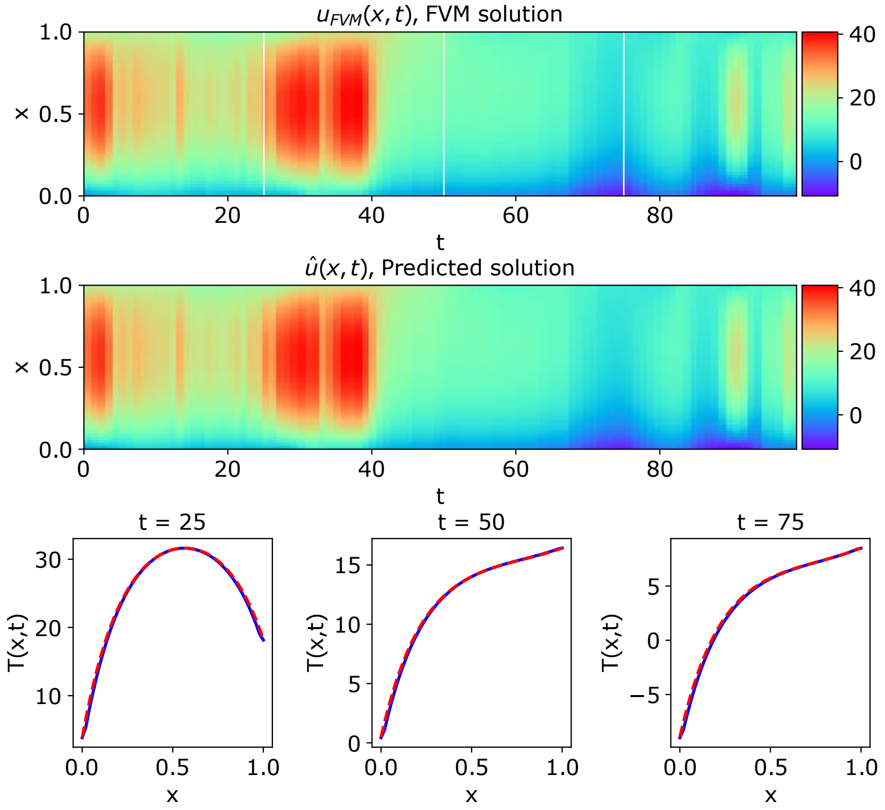
Solution of the first 100 hours



Blue line: FVM solution
Red-dotted line: PINNs prediction

L_2-error: 1.557e-2

Prediction of the following 100 hours



Blue line: FVM solution
Red-dotted line: PINNs prediction

L_2-error: 1.597e-2

Key Points

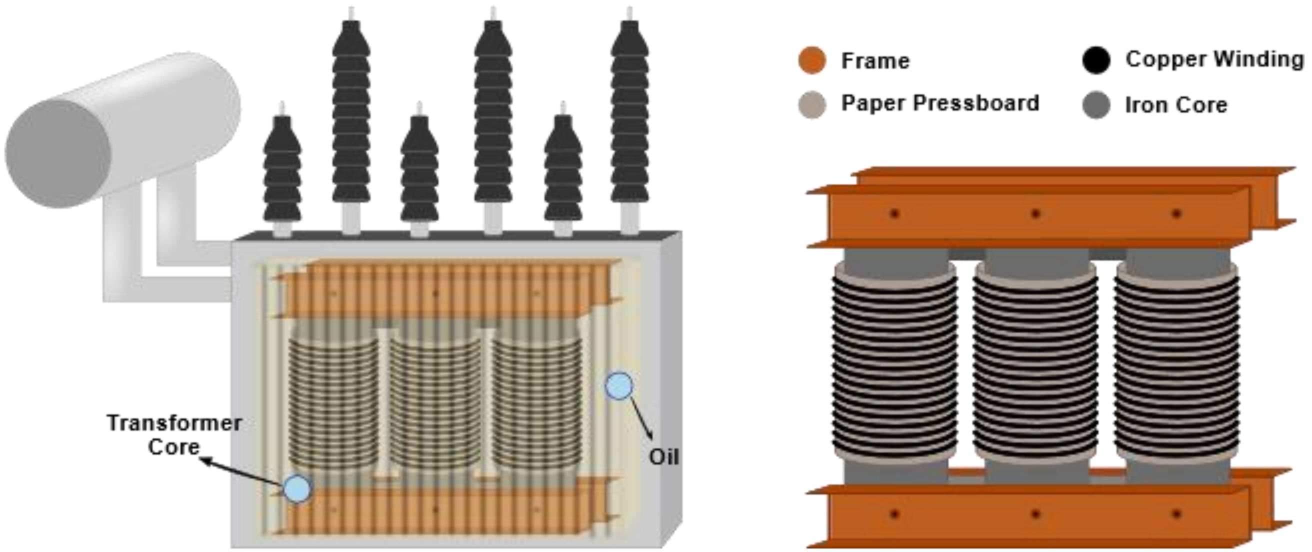
- Understanding when **PINN** should be used instead of a **numerical method**. PINNs can leverage existing measurements.
- **Measurements** are used from a **real transformer** rather than synthetic data.
- **Scaling** of the equation: PINNs face difficulties in handling large parameters.
- **Weights** assigned to the individual loss functions.
- PINNs as an ML **prediction tool**.

Transformer Cellulose Degradation

F. Bragone, K. Oueslati, T. Laneryd, M. Luvisotto, and K. Morozovska, *"Physics-Informed Neural Networks for Modeling Cellulose Degradation in Power Transformers"*, in IEEE ICMLA 2022.

Degree of Polymerization

Insulation System of Power Transformers



Correlation between DP and paper condition

Paper Condition	Degree of Polymerization (DP)
New	1000 - 1200
Good	650 - 1000
Average	350 - 650
Aged	< 350

Model

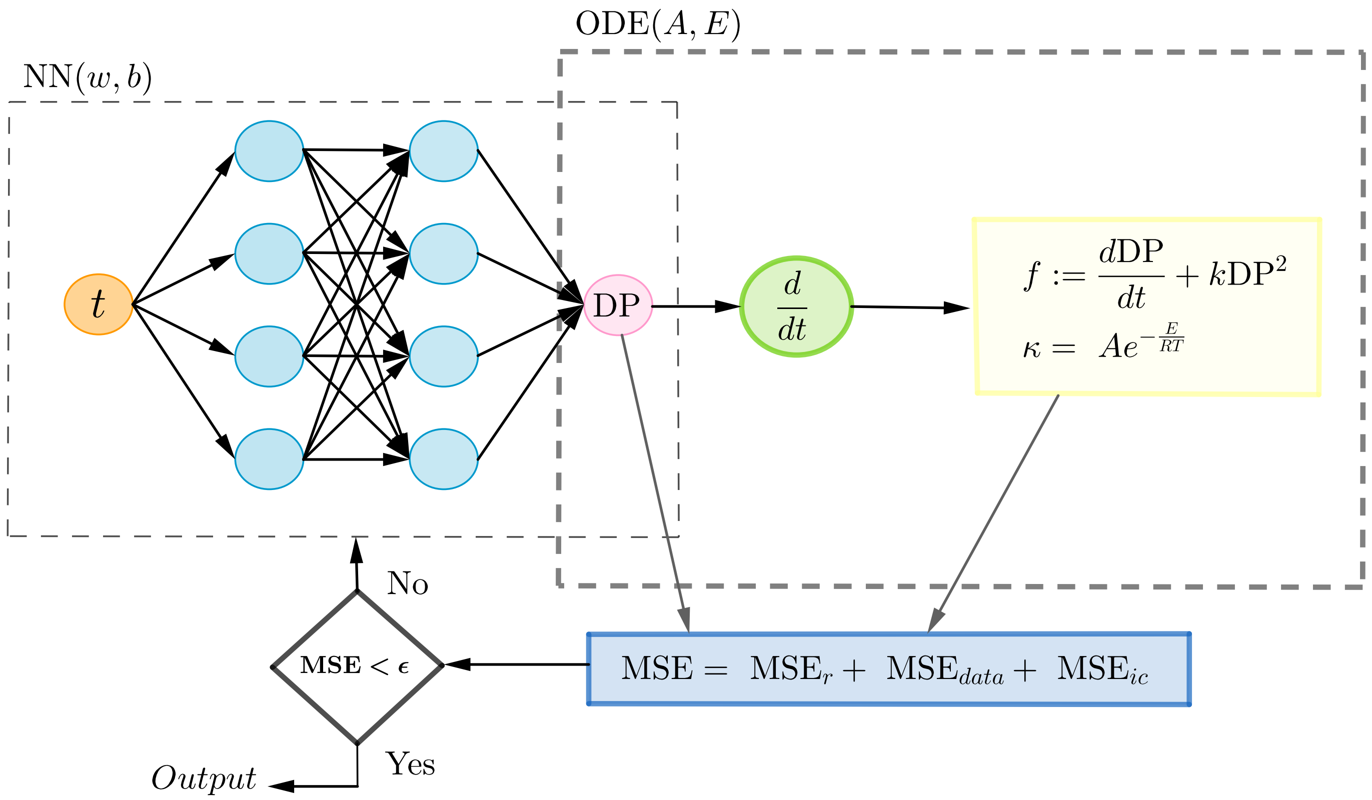
Emsley equation

$$\frac{dDP}{dt} = -k \cdot DP^2$$
$$k = A \cdot e^{-\frac{E}{RT}}$$
$$DP(0) = 1000$$

Set the function:

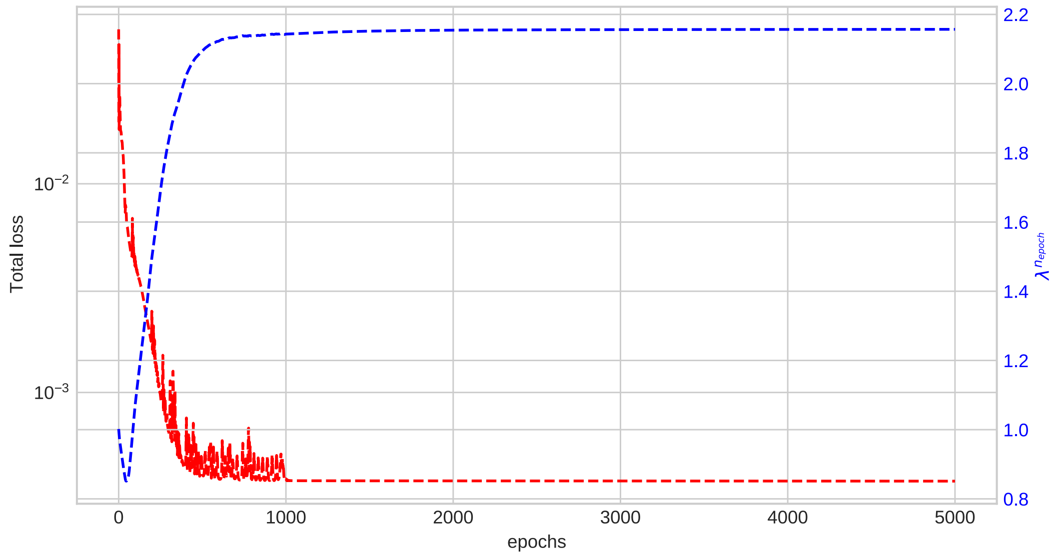
$$f \coloneqq \frac{dDP}{dt} + A \cdot e^{-\frac{E}{RT}} \cdot DP^2$$

- DP is the degree of polymerization,
- T is the temperature [K],
- R is the molar gas constant [8,314 J/K mol],
- A is the pre-exponential factor,
- E is the activation energy [kJ/mol].

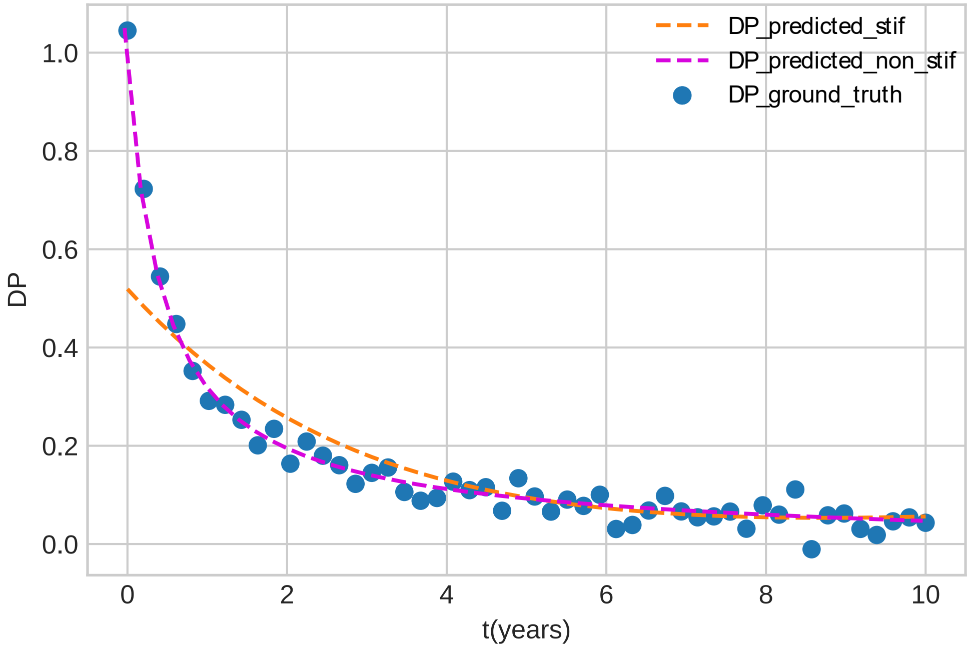


Results

Loss function evolution



DP prediction



- The mean of the inferred parameter A over 5 runs: 2.0803
- Real scaled value: 2.05

Stiff: using real values of the parameters A and E
Non-stiff: using scaled values of the parameters A and E

Key Points

- **Scaling** of large parameters: stiff and non-stiff problem.

- PINNs are a powerful tool for the **data-driven discovery** of PDEs and ODEs.

- Using **synthetic data** rather than real data: collecting data in this field is not easy.

Conclusions

Conclusions

- PINNs are neural networks that encode the **physics** expressed by ODEs and PDEs.
- PINNs can be used both for **solving** and **discovering** ODEs/PDEs.
    - Great potential to solve **inverse** problems.
- It can solve PDEs **without a mesh**.
- It calculates the derivatives of the network using **automatic differentiation**.
- Working with **real-world cases**:
    - Thermal distribution of power transformers: data-driven solution of the heat diffusion equation.
    - Cellulose degradation inside power transformers: data-driven discovery of the Emsley equation.
    - Scaling of the equation.
    - PINNs as a machine learning prediction tool.

References

[1] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities.," Proceedings of the national academy of sciences, vol. 79, no. 8, pp. 2554–2558, 1982.

[2] J. J. Hopfield and D. W. Tank, ""neural" computation of decisions in optimization problems," Biological cybernetics, vol. 52, no. 3, pp. 141–152, 1985.

[3] M. Takeda and J. W. Goodman, "Neural networks for computation: number representations and programming complexity," Applied optics, vol. 25, no. 18, pp. 3033–3046, 1986.

[4] C. Koch, J. Marroquin, and A. Yuille, "Analog" neuronal" networks in early vision.," Proceedings of the National Academy of Sciences, vol. 83, no. 12, pp. 4263–4267, 1986.

[5] H. Lee and I. S. Kang, "Neural algorithm for solving differential equations," Journal of Computational Physics, vol. 91, no. 1, pp. 110–131, 1990.

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

[7] K. Hornik, "Approximation capabilities of multilayer feedforward networks," Neural networks, vol. 4, no. 2, pp. 251–257, 1991.

[8] Dissanayake MW, Phan-Thien N. Neural-network-based approximations for solving partial differential equations. communications in Numerical Methods in Engineering. 1994 Mar;10(3):195-201.

[9] A. J. Meade Jr and A. A. Fernandez, "The numerical solution of linear ordinary differential equations by feedforward neural networks," Mathematical and Computer Modelling, vol. 19, no. 12, pp. 1–25, 1994.

[10] A. J. Meade Jr and A. A. Fernandez, "Solution of nonlinear ordinary differential equations by feedforward neural networks," Mathematical and Computer Modelling, vol. 20, no. 9, pp. 19–44, 1994.

[11] D. Gobovic and M. Zaghloul, "Design of locally connected cmos neural cells to solve the steady-state heat flow problem," in Proceedings of 36th Midwest Symposium on Circuits and Systems, pp. 755–757, IEEE, 1993.

References

[12] D. Gobovic and M. E. Zaghloul, “Analog cellular neural network with application to partial differential equations with variable mesh-size,” in Proceedings of IEEE International Symposium on Circuits and Systems-ISCAS’94, vol. 6, pp. 359–362, IEEE, 1994.

[13] R. Yentis and M. Zaghloul, “Cmos implementation of locally connected neural cells to solve the steady-state heat flow problem,” in Proceedings of 1994 37th Midwest Symposium on Circuits and Systems, vol. 1, pp. 503–506, IEEE, 1994.

[14] R. Yentis and M. Zaghloul, “Vlsi implementation of locally connected neural network for solving partial differential equations,” IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, vol. 43, no. 8, pp. 687–690, 1996.

[15] I. E. Lagaris, A. Likas, and D. I. Fotiadis, “Artificial neural networks for solving ordinary and partial differential equations,” IEEE transactions on neural networks, vol. 9, no. 5, pp. 987–1000, 1998.

[16] I. E. Lagaris, A. C. Likas, and D. G. Papageorgiou, “Neural-network methods for boundary value problems with irregular boundaries,” IEEE Transactions on Neural Networks, vol. 11, no. 5, pp. 1041–1049, 2000.

[17] J. Sirignano and K. Spiliopoulos, “Dgm: A deep learning algorithm for solving partial differential equations,” Journal of computational physics, vol. 375, pp. 1339–1364, 2018.

[18] M. Raissi, P. Perdikaris, and G. E. Karniadakis, “Inferring solutions of differential equations using noisy multi-fidelity data,” Journal of Computational Physics, vol. 335, pp. 736–746, 2017.

[19] M. Raissi, P. Perdikaris, and G. E. Karniadakis, “Machine learning of linear differential equations using gaussian processes,” Journal of Computational Physics, vol. 348, pp. 683–693, 2017.

[20] M. Raissi, P. Perdikaris, and G. E. Karniadakis, “Numerical gaussian processes for time-dependent and nonlinear partial differential equations,” SIAM Journal on Scientific Computing, vol. 40, no. 1, pp. A172–A198, 2018.

[21] M. Raissi and G. E. Karniadakis, “Hidden physics models: Machine learning of nonlinear partial differential equations,” Journal of Computational Physics, vol. 357, pp. 125–141, 2018.

Thank you all for listening!