modely

A Framework for Designing Model-Structured Neural Networks for Modeling, Control, and Estimation of Physical Systems

29 August 2025

Gastone Pietro Rosati Papini

University of Trento, Italy

Overview

The Neu4mes Project

The Neu4mes Project

Pricipal Investigator Gastone P. Rosati P., University of Trento

Founded by the **Italian Ministry of University and Research** (MUR) under the *Science Italian Found 2023* program, for **1.2 Million Euros**.

Aim Develop a new approach to model and control physical system using **Model-Structured Neural Networks**

Outcome the modely framework, a **wiki website** for collecting MSNN models.

Case Study Several autonomous systems, including: **Drones**, **Quadrupeds**, **Vehicles**.

More details at https://tonegas.it/

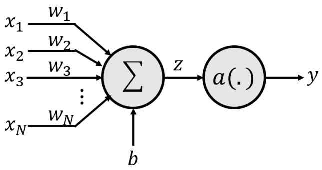
Motivation

Motivation

Why Model-Structured Neural Networks?

The Perceptron

- A **neural network** (NN) is a mathematical model inspired by the structure of the human brain
- The basic component of a NN is the ***Perceptron***[1]

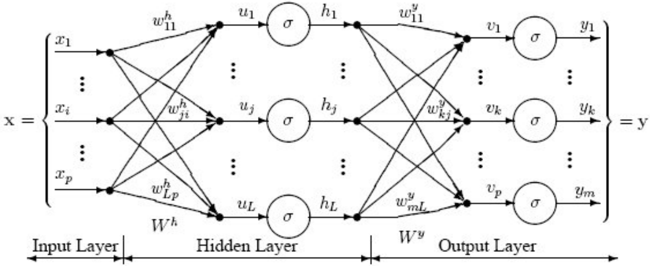


- From a mathematical point of view:

$$z = \bar{w}^T \bar{x} + b, \quad y = a(z)$$

[1]Frank Rosenblatt. "The perceptron: a probabilistic model for information storage and organization in the brain.". In: *Psychological review* (1958).

From Single- to Multi-layer Perceptron



- Multiple perceptrons can be concatenated to model general nonlinear functions (see Tensorflow Playground)
- **Universal approximation theorem**: a NN with a single hidden layer can approximate any continuous function[2]

[2]George Cybenko. "Approximation by superpositions of a sigmoidal function". In: *Math. Control Signal Systems* (1989).

The NN Internal Structure Matters

- *Computer vision*: **convolutional neural networks**, embedding the concept of spatial invariant[3]
- *Speech recognition*: **long short-term memory networks**, embedding the concept of short and long term memory[4]
- *Data generation*: **generative adversarial networks**, used for generative AI[5] (example: OpenAI Dall-E)
- *Natural language processing*: **transformers** embed the concept of self-attention[6] (example: chatGPT)

---

[3]Yann LeCun, Koray Kavukcuoglu, and Clement Farabet. "Convolutional networks and applications in vision". In: *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*. 2010.

[4]Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Computation* (1997).

[5]Ian Goodfellow et al. "Generative adversarial networks". In: *Communications of the ACM* (2020).

[6]Ashish Vaswani et al. "Attention is all you need". In: *Advances in neural information processing systems* 30 (2017).

Model-based approach

Data-driven approach

Model-Structured Neural Networks (MS-NNs)

Model-based approach

- Leverage physics knowledge
- Equation-based formulation
- Hard to model everything
- System identif. may require ad-hoc experiments

Data-driven approach

Model-Structured Neural Networks (MS-NNs)

Model-based approach

- Leverage physics knowledge
- Equation-based formulation
- Hard to model everything
- System identif. may require ad-hoc experiments

Data-driven approach

- No need for expert knowledge
- Model complex functions
- Black-box structure
- Data-hungry
- Poor generalization

Model-Structured Neural Networks (MS-NNs)

Model-based approach

- Leverage physics knowledge
- Equation-based formulation
- Hard to model everything
- System identif. may require ad-hoc experiments

Data-driven approach

- No need for expert knowledge
- Model complex functions
- Black-box structure
- Data-hungry
- Poor generalization

Model-Structured Neural Networks (MS-NN)

- **Flexibility** of data-driven (NN) techniques
- Embed **domain-specific knowledge** into the NN's internal **structure** $\longrightarrow$ increased **explainability** + **generalization**

Motivation

Approaches

Overview of Physics-Driven NN Approaches

How to embed physics- or model-based knowledge in NNs?

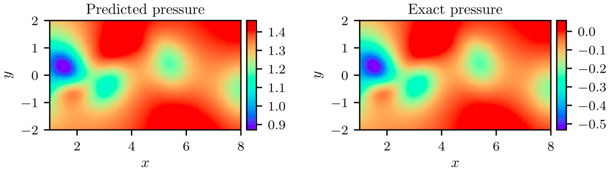
- **Decide on problem**
- **Curate the data**
- **Design the internal architecture**:
- **Design a loss function**
- **Design of optimization algorithms**

Motivation

Related Works

Physics-informed NNs (PINNs), Raissi et at. (2019)

- **Idea**: Data-driven solution or discovery of complex PDEs
- **How**: *Structured loss function* based on the PDE residuals, but using deep *unstructured* NNs
- **Pros**: Learn the solution of PDEs using small training datasets; accelerate simulations of complex systems
- **Examples**: Fluid dynamics (Navier Stokes), quantum mechanics, solid mechanics, reaction-diffusion[7]. **Note**: no modeling of real physical systems, only PDEs



[7] M. Raissi, P. Perdikaris, and G.E. Karniadakis. "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations". In: *Journal of Computational Physics* (2019).

Other approaches related to PINNs

- **Hamiltonian**[8] and **Lagrangian NNs**[9] that learn to conserve energy in mechanical systems and robotic tasks
- **Pontryagin AI**[10] to learn the policies of optimal control tracking problems (OCPs):
  - NN to approximate the controls of an OCP
  - NN trained with a tracking loss function
  - No need to solve the adjoint equations of the OCP
- NNs to approximate the value function of Hamilton-Jacobi-Bellman equations for **dynamic programming**[11]

---

[8]Sam Greydanus, Misko Dzamba, and Jason Yosinski. *Hamiltonian Neural Networks*. 2019.

[9]Michael Lutter, Kim Listmann, and Jan Peters. "Deep Lagrangian Networks for end-to-end learning of energy-based control for under-actuated systems". In: 2019.

[10]Lucas Böttcher, Nino Antulov-Fantulin, and Thomas Asikis. "AI Pontryagin or how artificial neural networks learn to control dynamical systems". In: *Nature Communications* (2022).

[11]Murad Abu-Khalaf and Frank L. Lewis. "Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network HJB approach". In: *Automatica* (2005).

Combining NNs and analytical models

- **Idea**: Use NNs to improve the prediction of analytical models
- **How**: Augment an analytical model with NNs to better learn some complex dynamics
- **Pros**: No need to learn the entire model from scratch; partly preserve the physical meaning of the model
- **Examples**:
  - NN to learn the parameters of multibody vehicle models[12]
  - Neural tire models in multibody vehicle models[13][14]
  - Neural ODE to predict the heart rate during workouts[15]

[12] John Chrosniak, Jingyun Ning, and Madhur Behl. "Deep Dynamics: Vehicle Dynamics Modeling With a Physics-Constrained Neural Network for Autonomous Racing". In: *IEEE Robotics and Automation Letters* 9.6 (2024).

[13] Jan Węgrzynowski et al. *Learning dynamics models for velocity estimation in autonomous racing*. 2024.

[14] Taekyung Kim, Hojin Lee, and Wonsuk Lee. "Physics Embedded Neural Network Vehicle Model and Applications in Risk-Aware Autonomous Driving Using Latent Features". In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2022.

[15] Achille Nazaret et al. "Modeling personalized heart rate response to exercise and environmental factors with wearables data". In: *npj Digital Medicine* 6 (Nov. 2023).

Our Model-Structured NNs

- **Idea**: Embed domain-specific prior knowledge into the NN's internal structure
- **How**: Neuralize the structure of a physics-based model
- **Pros**: Model interpretability; improved generalization
- **Examples**: Modeling *physical systems* (unknown dyna./param.)
  - Model and control the long.-lat. vehicle dynamics[16][17]
  - Vehicle state estimation[18]
  - Model mechanical systems[19];
  - Human motion prediction[20]

[16] Mauro Da Lio, Daniele Bortoluzzi, and Gastone Pietro Rosati Papini. "Modelling longitudinal vehicle dynamics with neural networks". In: *Vehicle System Dynamics* (2020).

[17] Mattia Piccinini et al. "A Physics-Driven Artificial Agent for Online Time-Optimal Vehicle Motion Planning and Control". In: *IEEE Access* (2023).

[18] Mauro Da Lio, Mattia Piccinini, and Francesco Biral. "Robust and Sample-Efficient Estimation of Vehicle Lateral Velocity Using Neural Networks With Explainable Structure Informed by Kinematic Principles". In: *IEEE Transactions on Intelligent Transportation Systems* (2023).

[19] Antonio Di Dino, Francesco Biral, and Paolo Bosetti. "Hybrid Modeling of Non-Linear Mechanical Systems: The Case of a Vehicle Shock Absorber". In: 2011.

[20] Alessandro Antonucci et al. "Efficient Prediction of Human Motion for Real-Time Robotics Applications With Physics-Inspired Neural Networks". In: *IEEE Access* (2022).

NNodely Framework

Motivation

- *Modeling*, *control* and *estimation* of **physical systems**, whose internal dynamics may be partially unknown

Motivation

- *Modeling*, *control* and *estimation* of **physical systems**, whose internal dynamics may be partially unknown
- **Speed-up** the development of MS-NN, which is complex to design using standard deep-learning framework

The code is reduced by 75%

Motivation

- *Modeling*, *control* and *estimation* of **physical systems**, whose internal dynamics may be partially unknown
- **Speed-up** the development of MS-NN, which is complex to design using standard deep-learning framework
- **Support professionals** with classical modeling backgrounds, such as **physicists** and **engineers**, in using data-driven approaches (but embedding knowledge inside) to address their challenges

It avoids the common mistakes of using standard deep-learning frameworks

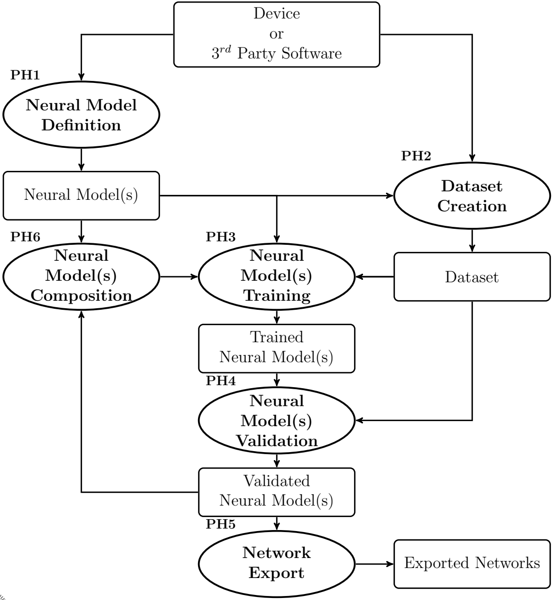
Motivation

- *Modeling*, *control* and *estimation* of **physical systems**, whose internal dynamics may be partially unknown
- **Speed-up** the development of MS-NN, which is complex to design using standard deep-learning framework
- **Support professionals** with classical modeling backgrounds, such as **physicists** and **engineers**, in using data-driven approaches (but embedding knowledge inside) to address their challenges
- A **repository** of incremental **know-how** that effectively collects approaches with the same purpose, i.e. building an increasingly advanced library of MS-NNs for various applications

NNodely Framework

Workflow

Framework workflow



modely
neuralize your model

NNodely Framework

Physical Principles

The Principle

The **MS-NN** embeds and generalizes **physical principles** and prior-model knowledge, where the parameters are learned from **experimental data**

Embedding Physical Principles (Main Ideas)

Main nnodely concepts:

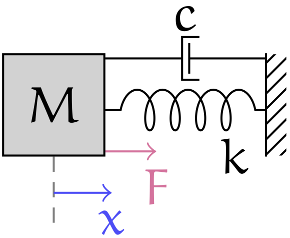
- Linear **superposition of forces**.
- The system dynamics are treated using **FIR** or **IIR** filter/layer.
- The non-linearities can be addressed using **local models** or **custom functions**
- Capturing the residuals of analytical models (such as friction, delays, and backlash) through **ad-hoc neuralization**
- Embedding easily the concept of **time** inside the NN
- Flexible managing of **recursive networks**

Use Cases

Use Cases

Example of Learning Model and Control of a System

Learn the dynamics of a mass-spring-damper system

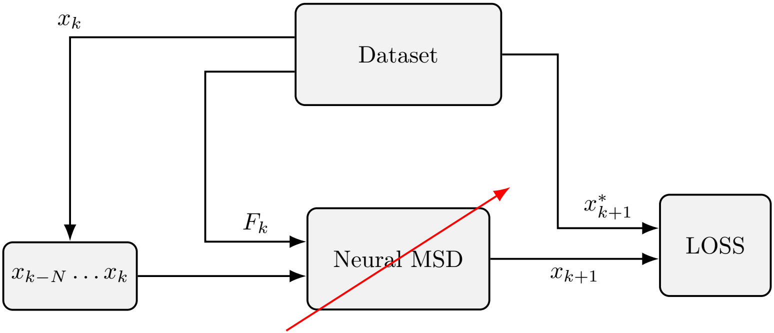


$$M\ddot{x} + c\dot{x} + kx = F$$

$$x_{k+1} = \overbrace{\mathrm{FIR}(\bar{x})}^{\overbrace{\sum_{k=0}^{N-1} x[-k]\cdot h_x[(N-1)-k]}^{\frac{c\dot{x}+kx}{M}}} + \overbrace{\mathrm{FIR}(F)}^{\overbrace{F[0]\cdot h_F}^{\frac{F}{M}}}$$

Model and Control a Mass-spring-damper

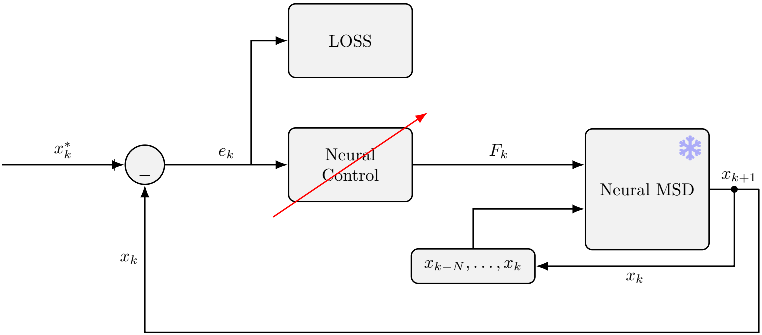
Learn the neural model for a mass-spring-damper system



$$\text{LOSS} = \sum_{\text{examples}} e_k^2 = \sum (x_k - x_k^*)^2$$

Model and Control a Mass-spring-damper

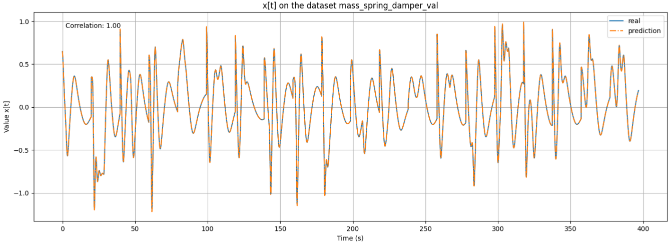
Learn the Neural control for a mass-spring-damper system

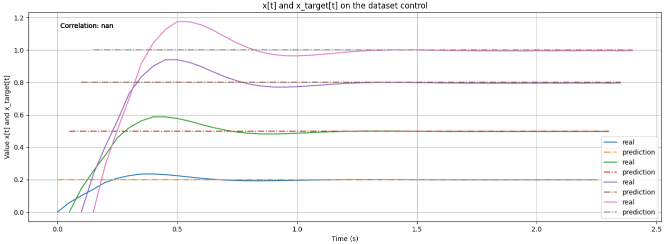


$$\text{LOSS} = \sum_{\text{examples}} e_k^2 = \sum (x_k - x_k^*)^2$$

Model and Control a Mass-spring-damper

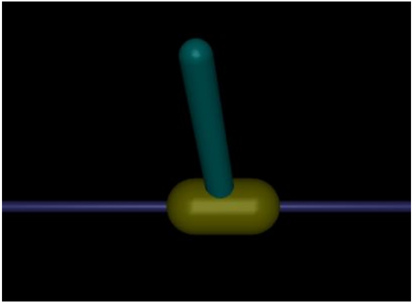
Results for modelling and low levelcontrol





Model and Control anInverted Pendulum

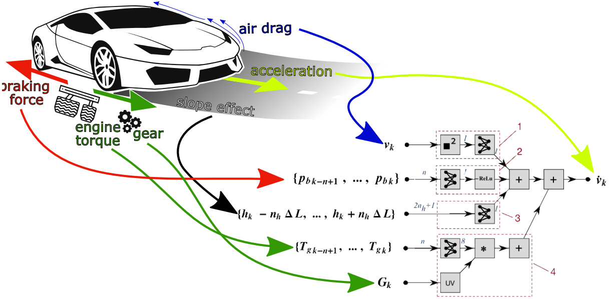
Model and Control of an inverted pendulum



Show video of the controlled the inverted pendulum

Use Cases

Modeling and Control Complex Systems

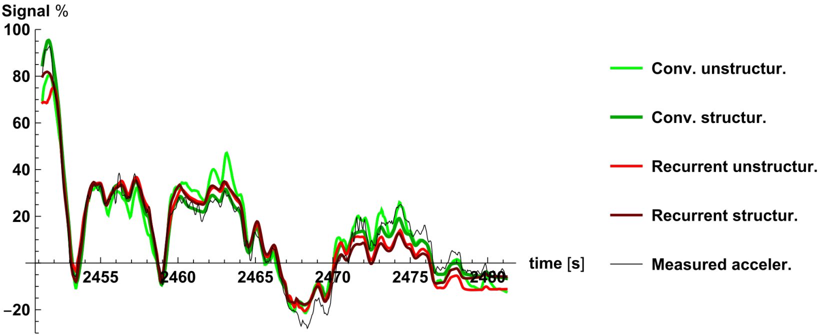


$$m\,\dot{v}_x = F_{x_f} + F_{x_r} - k_d v_x^2 - m\left(g c_r \cos(\theta) + g \sin(\theta)\right)$$

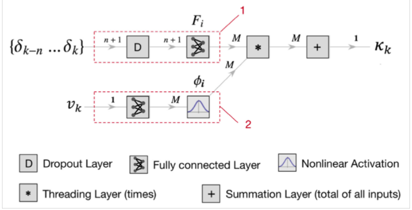
$$\dot{v}_x = \sum_{i}^{\text{gear}} \overbrace{g_i \mathrm{FIR}_i(\bar{T}_g) - \mathrm{Relu}(\mathrm{FIR}(\bar{p}_b))}^{(F_{x_f}+F_{x_r})/m} + \overbrace{\mathrm{FIR}(v_x^2)}^{k_d v_x^2/m} - \overbrace{\mathrm{Linear}(\bar{h}_k)}^{g c_r \cos(\theta)+g \sin(\theta)}$$

[21] Mauro Da Lio, Daniele Bortoluzzi, and Gastone Pietro Rosati Papini. "Modelling longitudinal vehicle dynamics with neural networks". In: *Vehicle System Dynamics* (2020).

Results for modelling of longitudinal vehicle dynamics



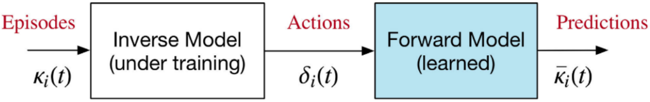
Forward model of a vehicle's lateral dynamics[22]

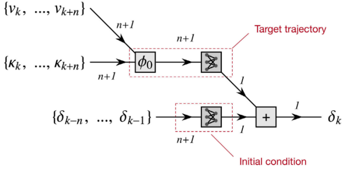


[22] Mauro Da Lio et al. "A mental simulation approach for learning neural-network predictive control (in self-driving cars)". In: *IEEE Access* (2020).

Vehicles

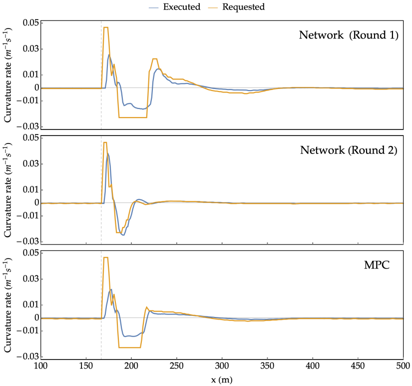
1. Train the **forward model** of the vehicle's lateral dynamics
2. Use the trained forward model to train the **inverse model**, which is a control system → unsupervised learning

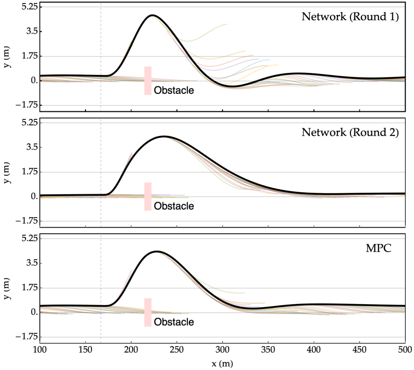




Vehicles

Results of control of a vehicle’s lateral dynamics of a **passanger vehicle**

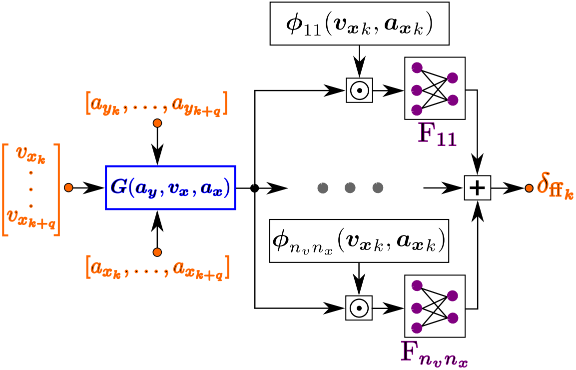




Show video of Renegade car driven by our lateral controller

Control Lateral Vehicle Dynamics for Racing Vehicles

Results of control of a vehicle's lateral dynamics of a **racing vehicle**



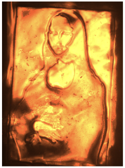
Show video of the F1Tenth car driven by our lateral controller

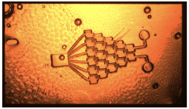
Use Cases

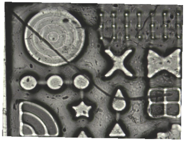
Other examples of MS-NNs

Controlling the 3D printing process

The network for defining the 3D printing process parameters.







$$\phi(t) = 1 - \exp^{-t^3 k}$$

Models nucleation and growth in crosslinking during polymerization.

$$d_c = D_p \log\left(\frac{E}{E_C}\right)$$

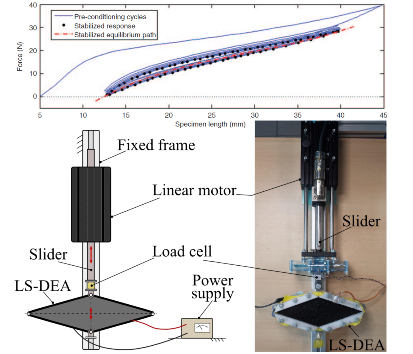
Describes how deep the resin will cure, depending on $\frac{E}{E_C}$.

- Degree of polymerization $\phi(t)$ (unitless)
- Reaction rate constant k (s^{-n})
- Cure depth d_c (μm)
- Penetration depth D_p (μm)
- Delivered energy dose E (mJ/cm^2)
- Critical energy E_C (mJ/cm^2)

Model a response of electroactive polymers

The neural network model is base on a physical model of a viscoelastic material

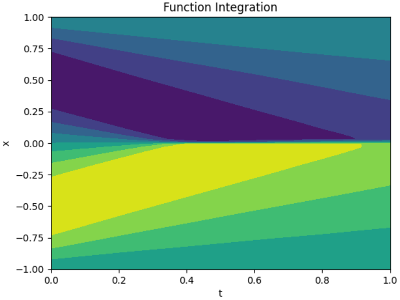
$$f_{\text{tot}} = f_{\text{e}}(\lambda) + f_{\text{d}}\left([\lambda_t, \dots, \lambda], [\dot{\lambda}_t, \dots, \dot{\lambda}]\right) + f_{\text{V}}(V, \lambda)$$



- total force realized: f_{tot}
- stretch: λ
- voltage: V
- elastic force: f_{e}
- viscoelastic and hysteretic force: f_{d}
- electric force: f_{V}

Integrate a nonlinear differential equation

Integrate the Burger's equation[23] in the range $x \in [-1, 1]$, $t \in [0, 1]$.



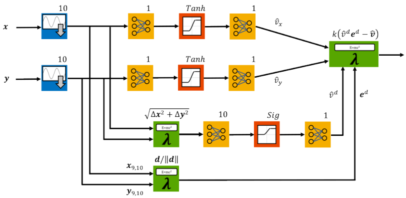
$$u = NN(t, x),$$
$$\frac{u}{dt} + u\frac{u}{dx} - \left(\frac{0.01}{\pi}\right)\frac{u}{dx^2} = 0,$$
$$u(0, x) = -\sin(\pi x),$$
$$u(t, -1) = u(t, 1) = 0.$$

[23] M. Raissi, P. Perdikaris, and G.E. Karniadakis. "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations". In: *Journal of Computational Physics* (2019).

Learning a pedestrian model

The network is based on social force model[24]

$$\mathbf{f}(t) = m\frac{v^d(t)\mathbf{e}^d(t) - \mathbf{v}(t)}{\tau} + \sum_w \mathbf{f}_w^W(t)$$



- forces on pedestrian: $\mathbf{f}$
- pedestrian mass: m
- pedestrian speed: $\mathbf{v}$
- desired velocity: v^d
- waypoint direction: $\mathbf{e}^d$
- velocity change rate: τ
- obstacle forces: $\mathbf{f}_w^W$

[24] Alessandro Antonucci et al. "Efficient Prediction of Human Motion for Real-Time Robotics Applications With Physics-Inspired Neural Networks". In: *IEEE Access* (2022).

NNodely-applications on github

The applications are available at

https://github.com/tonegas/nnodely-applications

modely
neuralize your model

Apply the framework on **your** task

and

Star it if you like!

Gastone Pietro Rosati Papini
gastone.rosatipapini@unitn.it
tonegas.it