university of groningen

faculty of science and engineering

University of Groningen

Control of nonlinear fully actuated physical system via Neural Interconnection and Damping Assignment - Passivity Based Control

Integration Project submitted in fulfillment of the requirements
for the degree of Bachelor of Industrial Engineering and Management

Under the supervision of dr. ir. B. Jayawardhana
and dr. M. Muñoz Arias

Veronika Tajgler (s3351068)

July 15, 2022

Abstract

Recently developed Neural Interconnection and Damping Assignment - Passivity Based Control (IDA-PBC) can be applied to a wide range of mechanical systems described in the port-Hamiltonian (pH) framework. It demonstrates a great potential to overcome an obstacle that comprises solving the matching equations of closed-loop system. This control scheme relies upon exploiting the universal function approximation property of neural networks and reformulating non-parameterized IDA-PBC into a supervised learning optimization problem. This study aims to highlight the applicability of Neural IDA-PBC by conducting real-life experiments on a fully-actuated physical system, namely working with a swing-up pendulum problem. Finally, the performance of the Neural nonlinear controller is compared to the analytical IDA-PBC, and the system's transient response to different controller parameters is analysed.

Contents

Acronyms

AD Automatic differentiation.

DNN Deep Neural Networks.

DSP Digital Signal Processor.

DTPA Discrete Technology & Production Automation.

FFT Fast Fourier Transform.

IDA-PBC Interconnection and Damping Assignment Passivity-Based Control.

LQ Linear Quadratic.

ME Matching equations.

NN Neural Networks.

Op-amp Operational power amplifier.

PBC Passivity-Based Control.

PDE Partial differential equation.

pH port-Hamiltonian.

PINN Physics-informed neural network.

RTI Real-Time Interface.

SGD Stochastic gradient descent.

UG University of Groningen.

List of Figures

List of Tables

1 Introduction

An appeal towards nonlinear control is caused by modern emerging technologies that demand sophisticated control laws to meet design and functional requirements. A class of nonlinear energy-based control techniques has demonstrated prominent results when dealing with a wide range of mechanical systems that can be described in a port-Hamiltonian (pH) framework. One of the examples is Interconnection and Damping Assignment - Passivity Based Control (IDA-PBC) (Ortega et al., 2002). The key idea has emerged from modeling the dynamics of the physical system via energy and assigning the desirable interconnection and damping matrices given new energy storing function has its minimum at the desired equilibrium point. Recognizing the energy as 'lingua Franca' provides an exceptional advantage to this method, precisely its potential to deal with multi-domain physical systems (Schaft and Jeltsema, 2014).

Even though this methodology provides a clear physical interpretation of the control scheme in terms of energy and offers control while respecting the original dynamics of the system, the main barrier that hinders its implementation is solving a set of partial differential equations (PDEs). So-called matching equations (MEs) preserve closed-loop dynamics of the system and are required to fulfill the equilibrium assignment and achieve Lyapunov stability of the closed-loop. Hence, most of the research in this field relies on restricting application to certain types of systems or forming assumptions depending on the design choice (Nageshrao et al., 2014).

Embedding Neural Networks (NN) into an IDA-PBC control scheme proposed by Sanchez-Escalonilla et al. (2021) enables estimating the solution for partial differential equations. By exploiting the universal approximation property of neural networks and encoding physics law into supervised learning tasks, this research has shown that traditional non-parameterized IDA-PBC can be reconstructed into optimization problem based on IDA-PBC. Hereby, this methodology is referred to as Neural IDA-PBC. So far, it has shown promising results via simulations on fully- and under-actuated physical systems (Sanchez-Escalonilla et al., 2022).

This research aims to validate the potential of Neural IDA-PBC via conducting experiments on a fully actuated physical system, namely working on a swing-up pendulum problem. To make this feasible simple pendulum is being seen as a port-Hamiltonian system. Furthermore, all required parameters for control implementation is obtained via experiments. The analysis of system's transient response is done for two different NN architectures, and the results are compared to the analytical solution of the Algebraic IDA-PBC control scheme.

Finally, major critical parameters that affect nonlinear controller performance are analysed.

This study comprises five following chapters. Chapter 2 presents preliminaries on Neural IDA-PBC. It covers fundamentals of port-Hamiltonian system modeling and Interconnection and Damping Assignment; an introduction to universal function approximation property of neural networks, and the method of incorporating physics laws into supervised learning algorithm known as Physics Informed Neural Networks (PINNs). Chapter 3 encompasses the analysis of the studied system and a methodology used during the research that serves for further control implementation. Chapter 4 determines simulation and experimental results for different scenarios to illustrate the working principle of the controller design. Chapter 5 covers the discussion on the results and the limitations of this research, and finally, it embraces the suggestions for future work. Finally, the discussion and conclusion of derived results, followed by recommendations for future work, are presented in Chapter 6.

2 Preliminary on Neural IDA-PBC

2.1 Port-Hamiltonian systems

Port-Hamiltonian (pH) framework emerges as a methodology for modeling complex physical systems. It is achieved by recognizing energy as the 'lingua Franca' between physical domains, employing interconnection of identified ideal system components that capture the main characteristics of the system (Schaft and Jeltsema, 2014). In general, in port-based modeling, physical systems are regarded as the interconnection of three types of ideal components: (1) energy-storing elements, (2) energy-dissipating (resistive) elements, and (3) energy-routing elements (Schaft and Jeltsema, 2014).

Furthermore, pH systems are the open dynamical systems, and can interact with their environment through ports (van der Schaft, 2006). Hence, pH is a preliminary framework for the control of complex physical systems. Particularly, when working with non-linear systems, port-Hamiltonian provides a range of concepts and tools for modeling and interpreting the physical properties of the system that should be exploited and/or respected in the design of control laws (Schaft and Jeltsema, 2014).

2.1.1 Mechanical systems in port-Hamiltonian framework

The dynamics of a mechanical system in the pH framework with generalized coordinates q on the configuration space $\mathcal{Q} \subset \mathbb{R}^n$ and velocity $\dot{q} \in T_q\mathcal{Q}$ can be represented as follows:

$$\begin{aligned} \dot{x} &= [J(x) - R(x)]\frac{\partial H}{\partial x}(x) + g(x)u, \\ y &= g^\top(x)\frac{\partial H}{\partial x}(x), \end{aligned} \tag{1}$$

where $x = (q, p) \in \mathcal{X}$ is a state vector, $p := M(q)\dot{q}$ is the generalized momentum, and $u \in \mathbb{R}^m$, $m \leq n$ is the control input; $J(x), R(x) : \mathbb{R}^n \mapsto \mathbb{R}^{n\times n}$, with $J(x) = -J^\top(x)$ and $R(x) = R(x)^\top \geq 0_{n\times 0}$ are interconnection and dissipation matrices respectively. $H(x) : \mathbb{R}^n \mapsto \mathbb{R}$ is the Hamiltonian, which is a total stored energy and is equal to $H(x) = \frac{1}{2}p\top M^{-1}(q)p + U(q)$, where scalar function $U(q)$ is the potential energy and $M(q) = M^\top(q) > 0$ is the inertia matrix. $g(x) : \mathbb{R}^n \mapsto \mathbb{R}^{n\times m}$ is the input matrix. The input u represents generalized forces, while the output y gives the generalized velocity, so its product has the units of power, while $u, y \in \mathbb{R}^m$. If $n = m$, the mechanical system in 1 is fully-actuated. Otherwise, if the rank of input matrix is $m < n$, it means that the number of control inputs is less than the number of independent degrees of freedom (Venkatraman, 2010).

The power balance equation that provides a relation between internal and external power is,

$$\begin{aligned}\dot{H}(x) &= \frac{\partial H^\top}{\partial q}(x)\dot{q} + \frac{\partial H^\top}{\partial p}(x)\dot{p} \\ &= -\frac{\partial H^\top}{\partial p}(x)R(x)\frac{\partial H}{\partial p}(x) + y^\top u \leq y^\top u,\end{aligned} \tag{2}$$

where the first term on the right-hand side (non-positive) represents the dissipation due to the resistive (friction) elements in the system (Ortega et al., 2002). $y^\top u$ is the external supplied power.

2.2 Interconnection and Damping Assignment Passivity-based Control

Passivity-based control (PBC) is a well-established controller design methodology introduced by Ortega and Spong (2000). The control objective of this method is to achieve *stabilization by passivation*, through passivizing the system with a storage function that is at its minimum at the desired equilibrium point. Further work carried out by Ortega et al. (2002) introduced Interconnection and Damping Assignment Passivity-Based Control (IDA-PBC) that extends controller implementation to a broader class of physical systems that are described via the pH framework. Contrary to PBC, in this case, the system is controlled by assigning it a desirable interconnection and damping matrices giving the new energy storage function that has a minimum at the desired equilibrium point; and as it was stated by Ortega et al. (2002) the stabilization is achieved via *energy-balancing*. The objective is to design a static state feedback control law $u(x) = \beta(x) + v$, such that the closed-loop dynamics are given by the desired passive pH system,

$$\begin{aligned}\dot{x} &= [J_d(x) - R_d(x)]\frac{\partial H_d}{\partial x}(x) + g(x)u(x) \\ y' &= g(x)\frac{\partial H_d}{\partial x}(x),\end{aligned} \tag{3}$$

where the new energy function $H_d(x)$ has a strict local minimum at the desired equilibrium point $x^* = (q^*, 0)$. $J_d(x) = -J_d^\top(x), R_d(x) = R_d^\top(x)$ are the desired interconnection and damping matrices respectively; y' (which may be equal to y) is the new passive output.

Given $J(x), R(x), H(x), g(x)$ of the pH system in (1) and the desired equilibrium to be stabilized is $x^* \in \mathcal{X}$. Assuming that there are functions $\beta(x), J_a(x), R_a(x)$, and a vector function

$K(x)$ satisfying the matching equation

$$[J(x) + J_a(x) - (R(x) + R_a(x))]K(x) = -[J_a(x) - R_a(x)]\frac{\partial H}{\partial x}(x) + g(x) \tag{4}$$

such that the following four conditions hold:

(P1) Structure preservation:

$$J_d(x) := [J(x) + J_a(x)] = -[J(x) + J_a(x)]^\top \tag{5}$$
$$R_d(x) := [R(x) + R_a(x)] = [R(x) + R_a(x)]^\top \geq 0 \tag{6}$$

(P2) Integrability: $K(x) = \frac{\partial H_a}{\partial x}$ is the gradient of the scalar function

$$\frac{\partial K}{\partial x}(x) = \left[\frac{\partial K}{\partial x}(x)\right]^\top \tag{7}$$

(P3) Equilibrium assignment: $K(x)$, at x^* verifies

$$K(x^*) = -\frac{\partial H}{\partial x^*} \tag{8}$$

(P4) Lyapunov stability: The Jacobian of $K(x)$, at x^* satisfies the bounds:

$$\frac{\partial^2 K}{\partial x^2}(x^*) = -\frac{\partial H}{\partial x^*} \tag{9}$$

Under this condition, the closed-loop system PBC has the dissipation of the form (3), where

$$H_d(x) := H(x) + H_a(x) \quad J_d(x) := J(x) + J_a(x) \quad R_d(x) := R(x) + R_a(x) \tag{10}$$

and

$$\frac{\partial H_a}{\partial x}(x) = K(x). \tag{11}$$

Furthermore, matching equation can be rewritten as:

$$g^\perp(x)[J_d(x) - R_d(x)]\frac{\partial H_d}{\partial x}(x) = g^\perp(x)[J(x) - R(x)]\frac{\partial H}{\partial x}(x), \tag{12}$$

where $g^\perp(x)$ is the full rank left annihilator of $g(x)$, i.e. $g^\perp(x)g(x) = 0$ for all $x \in \mathbb{R}^n$ (Nunna et al., 2013). Thus, the control law, where locally stabilized system (1) and the closed-loop

system (3) is passive with the new input v and output y, can be written as:

$$u = \underbrace{[g^\top(x)g(x)]^{-1}g^\top(x)\left((J_d(x) - R_d(x))\frac{\partial H_d}{\partial x}(x) - (J(x) - R(x))\frac{\partial H}{\partial x}(x)\right)}_{\beta(x)} + v \tag{13}$$

Moreover, if x^* is the largest invariant subset in $x \in \mathcal{X} | \frac{\partial H_d^\top}{\partial x}(x) R_d(x) \frac{\partial H_d^\top}{\partial x}(x) = 0$, then x^* is asymptotically stable (Sanchez-Escalonilla et al., 2021). The results hold globally if x^* is global minimum of $H_d(x)$ and $H_d(x)$ radially unbounded.

As it can be seen, the control law in (13), which achieves the desired objective, is obtained via solving partial differential equations (PDEs) in (12). This is due to arbitrate choice of the desired subsystems interconnections and damping (Ortega et al., 2002). If the system is underactuated, it is crucial to implement J_a, otherwise, it can be left out of the control design.

2.2.1 IDA-PBC challenge

One of the main factors that hinder the use of the IDA-PBC method is solving matching equations (Duindam et al., 2014). Hence, this topic has been a subject of interest for many researchers. One of the approaches proposed by Fujimoto and Sugie (2001) was Algebraic IDA-PBC. Using this technique, desired Hamiltonian function is fixed, such as $H_d(x) = (x - x^*)^2$. However, J_a and R_a are unknown and have to be determined to fulfill the design requirements (Nunna et al., 2013).

2.3 IDA-PBC Example: Simple pendulum

One of the prevailing examples of nonlinear systems is a simple pendulum. The pH system of the pendulum is given by:

$$\begin{aligned} \underbrace{\begin{bmatrix} \dot{q} \\ \dot{p} \end{bmatrix}}_{\dot{x}} &= \left(\underbrace{\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}}_{J(x)} - \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & \mathcal{D} \end{bmatrix}}_{R(x)} \right) \underbrace{\begin{bmatrix} \nabla_q H(q,p) \\ \nabla_p H(q,p) \end{bmatrix}}_{\frac{\partial H}{\partial x}} + \underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{g(x)} u(x) \\ y &= \underbrace{\begin{bmatrix} 0 & 1 \end{bmatrix}}_{g^\top(x)} \underbrace{\begin{bmatrix} \nabla_q H(q,p) \\ \nabla_p H(q,p) \end{bmatrix}}_{\frac{\partial H}{\partial x}}, \end{aligned} \tag{14}$$

Table 1: Simple Pendulum case: Parameters

Model Parameters	Symbol	Value	Units
Pendulum mass	m_p	1	kg
Gravity	g_p	9.80655	m/s^2
Pendulum length	l_p	1	m
Pendulum Inertia	J_p	1	kgm^2
Friction	$\mathcal{D}$	0.5	Nms

where $\mathcal{D}$ is a natural damping (in this case it is equal to 0) and $g(x) = \begin{bmatrix} 0 & 1 \end{bmatrix}^\top$ since the system is actuated at its join. The total energy of this system is given by the Hamiltonian function

$$H(q,p) = \frac{1}{J_p^2}p^2 + m_p g_p l_p(1 - cos(q)), \tag{15}$$

where J_p is the rotational moment of inertia, m_p and l_p are the mass and length of the pendulum, respectively, and g_p is a gravitational constant. The states q, p is the position (angle with respect to the normal) and the momentum accordingly. The model parameters used for the simulation are given in Table 1.

The objective is to stabilize the system at equilibrium point $(q^*, p^*) = (k\pi, 0)\ k \in \mathbb{Z}$, that can also be seen in Figure 1. Using the analytical solution for algebraic IDA-PBC method as described by Nageshrao et al. (2014), $H_d(x)$ is define as $H(x) + H_a(x)$, where $H_d = c(q^* - q)^2$. c is auxiliary constant (gain) depends on design requirements. Hence, $\frac{\partial H_a}{\partial q} = \frac{\partial H}{\partial q} + \frac{\partial H_a}{\partial q} = 2c(q^* - q)$. The closed loop feedback is calculated using (13) and is $\beta(p) = -\frac{\partial H_d}{\partial q} + \frac{\partial H}{\partial q}$. As a result, by setting the desired angle of the pendulum, it is possible to stabilize it as it can be seen in Figure 2.

2.4 Neural networks as universal function approximators

The method for solving ordinary and partial differential equations, that relies on function approximation capabilities of feedforward Neural Networks (NN) was introduced by Hornik et al. (1989) and later by Lagaris et al. (1998). In recent years approximation properties of NN have attracted many researchers, including Raissi et al. (2019), Karumuri et al. (2020), and Bai (2021) that utilized this technique for different applications.

This method is based upon a regression optimization problem. With this regard, the feedfor-

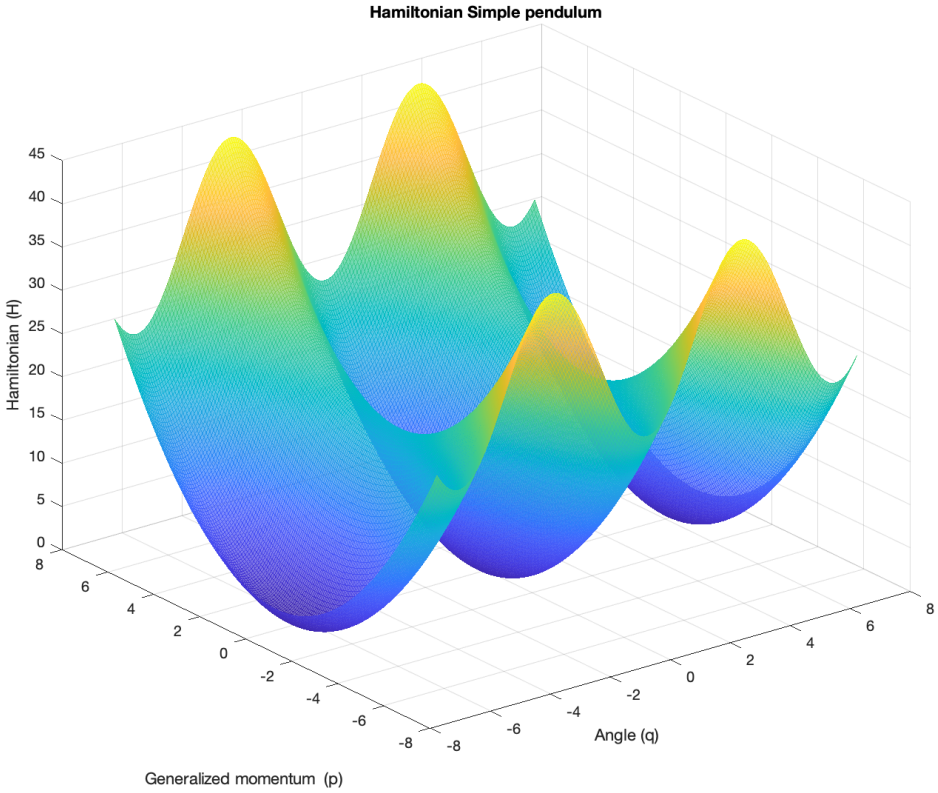


Figure 1: Hamiltonian energy profile.

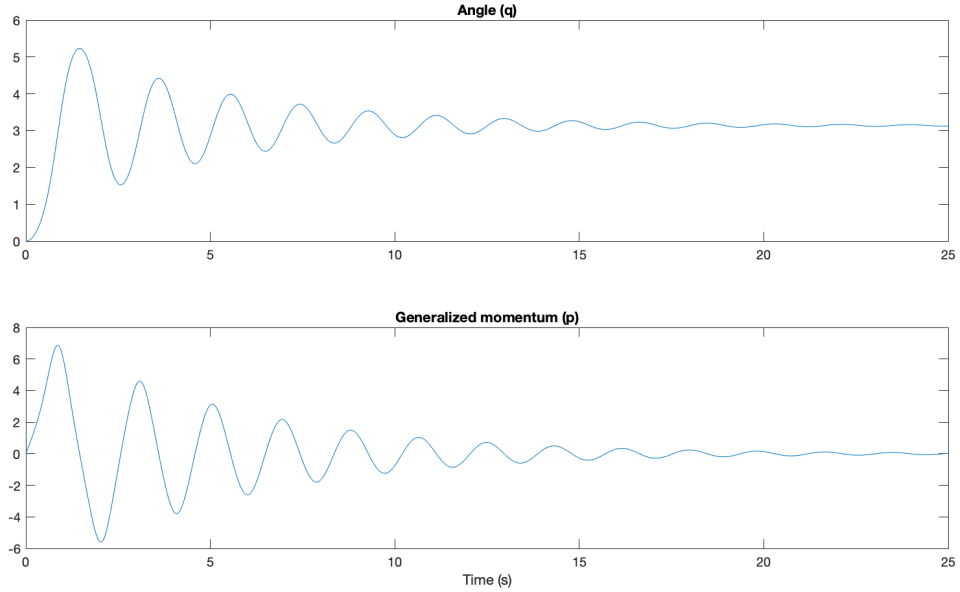


Figure 2: Closed loop Algebraic IDA-PBC: Simple Pendulum.

ward neural networks function is exploited as an approximation element, whose parameters (weight and biases) are adapting to minimize the loss (error) function defined beforehand according to the initial and boundary conditions of the particular ODEs or PDEs. Assuming, there is data that consist of inputs $\mathcal{X} : x_1, x_2, x_3...x_n$ and outputs $\mathcal{Y} : y_1, y_2, y_3...y_n$. The objective is to find the NN $N(\theta; x)$ that goes from input to output and minimizes the loss function :

$$\mathcal{L}(\theta) = \frac{1}{n}\sum_{i=1}^{n}[y_i - N(x_i;\theta)]^2 \tag{16}$$

This optimization method also involves the calculation of the stochastic gradient descent (SGD) of the error with respect to the network parameters:

$$\theta_{t+1} = \theta_t - a_t\frac{1}{m}\sum_{j=1}^{m}\nabla_\theta[y_{i_j} - N(x_{i_j};\theta_t)]^2 \tag{17}$$

where ∇_θ is a gradient, a_t is a learning rate. Starting from arbitrarily selected set of parameters, NN is trained until the loss function reaches optimal error.

However, depending on a function dealt with, NN may require an exponentially large number of neurons. According to Shannon's theorem, there always exists a Boolean function with n variables, which are greater than two ($n > 2$), that require at least $2^n/n$ Boolean gates regardless of the depths of neural networks. Moreover, when the problem gets complicated, such as PDE constrained optimization and (Bayesian) inverse problems, generating training data gets expensive (Bai, 2021).

2.5 Physics informed neural networks

Research done by Raissi et al. (2019) and Chen et al. (2020) has demonstrated the general methodology, namely physics-informed neural networks (PINNs). The main purpose of this technique is to implement physics laws into supervised learning tasks. This allows obviating the need for generating an extensive amount of training data since the PDE residual is collocated on training points of the approximating Deep Neural Networks (DNN) (Bai, 2021). To start with, consider parameterized nonlinear partial differential equations of the general form:

$$u_t + \mathcal{N}[u;\lambda] = 0, \tag{18}$$

where $\mathcal{N}[\cdot;\lambda]$ is a nonlinear operator parameterized by λ and u_t is the time derivative of the function $u(t,x)$. This setup comprises a wide range of mathematical physics, such as conser-

vation laws and kinetic equations (Raissi et al., 2019). To proceed with a solution, a residual term $f(t,x)$ is defined as a left hand side of 17; i.e. $f(t,x) := u_t + \mathcal{N}[u;\lambda]$. Hence $u(t,x)$ is approximated by neural network $f(t,x)$, while the network is derived by implementing automatic differentiation (AD) and activation functions (Raissi et al., 2019). The neural network that minimizes the residual is considered as an approximated solution of (18) and is trained using the loss function $\mathcal{L}(\theta;x)$ based on soft constraints approach is given in equation 19 as a main objective.

$$\mathcal{L}(\theta;x) = u_t(\theta;t,x) + \mathcal{N}[u(\theta;t,x);\lambda], \tag{19}$$

where $\theta^* = argmin_\theta \mathcal{L}(\theta;x)$, $u(t,x) = u_\theta(\theta^*;t,x) + \epsilon$. To maintain computational efficiency during the training, stochastic gradient descent is exploited together with mini-batch setting (Goodfellow et al., 2017).

This methodology has provided empirical evidence for approximating the solutions for PDEs as long as the equation is well-posed and has a unique solution(Sanchez-Escalonilla et al., 2021).

2.6 Solving IDA-PBC using PINNS

Assuming the existence of the solutions for the matching equation (ME) problem defined in 12, and taking into account the proposal (P1)-(P4), finding the missing auxiliary functions $J_a(x)$, $R_a(x)$, and $H_a(x)$ is an inverse problem, in particular ill-posses (Karniadakis et al., 2021), (Mishra and Molinaro, 2021).

To reduce the indeterminism of the problem, the non-parameterized IDA-PBC approach is used for this methodology. As it was stated by Nageshrao et al. (2014), matching equations then can be solved after fixing $J_d(x)$ and $R_d(x)$. Furthermore, Sanchez-Escalonilla et al. (2021) argue that fixing $J_d(x)$ and iteratively adapting $R_d(x)$ assists evading from triviality while assigning the values. Hence desired properties of the closed-loop system are achieved accurately. Simultaneously, $H_d(x) = H(x) + H_a(x)$ is approximated via NNs.

By implementing the PINNs methodology from section 2.5 into the IDA-PBC principles (P1)-(P4) in 2.2, the new objective function, that will be minimized by NNs, according to Sanchez-Escalonilla et al. (2021), is defined as follows.

(P1) Structure preservation residual
To assure the property, an auxiliary interconnection matrix $J_a(x)$ has to be a skew-symmetric

matrix. Thus, as proposed by Ortega and Spong (2000), $J_a(x) = \begin{bmatrix} 0 & J_1(x) \\ -J_1(x) & J_2(x) \end{bmatrix}$, where $J_1(x)$ is an arbitrary $n \times n$ matrix and $J_2(x) \in \mathbb{R}^{n \times n}$ is a skew-symmetric matrix. Both are fixed prior the optimization process.

The desired damping matrix R_d must be positive semi-definite. Hence, $R_a(\theta; x) = \begin{bmatrix} 0 & 0 \\ 0 & R_a(\theta;x) \end{bmatrix}$, where $R_a(\theta; x) \in \mathbb{R}^{n \times n}$ is symmetric and positive-definite matrix parameterized by θ. The parameterization enables to define the justifiable damping that fulfills the design criteria during the optimization process (Sanchez-Escalonilla et al., 2021). In particular, the residual is defined as:

$$f_{struct}(\theta; x) := \underbrace{\max\{0, c + \Re(\sigma(F_d(x)))\}}_{f_{damping}} + \underbrace{\|\Im(\sigma(F_d(x)))\|}_{f_{harmonic}}, \tag{20}$$

where $c > 0$ is the convergence rate of the new energy function H_d, $\Re$ and $\Im$ are the real and imaginary parts, respectively. The first term is related to the residual for prescribing the damping of the target system, while the second term corresponds to the residual for minimizing harmonic oscillations.

(P2) Integrability residual
As the NNs are used to obtain the scalar function $Ha(\theta; x)$, the integrability condition (P2) is fulfilled by construction.

(P3) Equilibrium assignment residual
This condition implies that the closed-loop energy function, $Hd(\theta; x) := H(x) + Ha(\theta; x)$, has a global minimum at the desired equilibrium $x^* = (q^*, 0)$. Correspondingly, the following residual function to assign the equilibrium point can be considered as:

$$f_{eq}(\theta; x, x^*) := \underbrace{\left\| \frac{\partial H_d}{\partial x}(\theta, x^*) \right\|^2}_{f_{eq,1}} + \underbrace{(H_d(\theta; x^*))^2 + \max\{0, -H_d(\theta, x^*)\}}_{f_{eq,2}}, \tag{21}$$

where $f_{eq,1}$ defines that x^* is a stationary point and $f_{eq,2}$ is used to induce a lower bound on H_d.

(P4) Lyapunov stability residual
This condition enforces $\frac{\partial H_d}{\partial x}(\theta; x)$ to be positive. Therefore, the residual function to guarantee (P4) is given by:

$$f_{lyap} := \max\{0, \sigma(F_d(\theta; x))\}, \tag{22}$$

where σ denotes the spectrum of a matrix and $c > 0$ is again a constant value that can be added to make this condition strict.

As a result, the IDA-PBC ME 12 can be rewritten in the residual form moving all the terms to the left-hand side as it is presented in section 2.5.

$$f_{matching} := g^{\perp}(x)[J_d(x) - R_d(x)]\frac{\partial H_d}{\partial x}(x) - g^{\perp}(x)[J(x) - R(x)]\frac{\partial H}{\partial x}(x) \tag{23}$$

Combining all residuals, the final loss function is determined as:

$$\mathcal{L} = f_{struct} + f_{eq} + f_{lyap} + f_{matching} \tag{24}$$

which will be minimised using NN optimization algorithm, hence auxiliary functions $R_a^*(x) \approx R_a(\theta^*; x) + \epsilon$ and $H_a^*(x) \approx H_a(\theta^*; x) + \epsilon$ are approximated.

3 Methodology

The object of this study is a pendulum system located in the Discrete Technology & Production Automation (DTPA) lab at the University of Groningen (UG). Simple pendulum system is a prevailing example of a nonlinear fully-actuated system with force applied by a DC motor as input and the pendulum angle as a single output. The scope of this research contains the real-life experiment on the stabilization of pendulum using a nonlinear controller based on neural IDA-PBC methodology to validate the applicability of the proposed control methodology. The system is controlled to achieve the desired (upright) position. In this section, the methods and tools used throughout the research are described.

3.1 Control objective

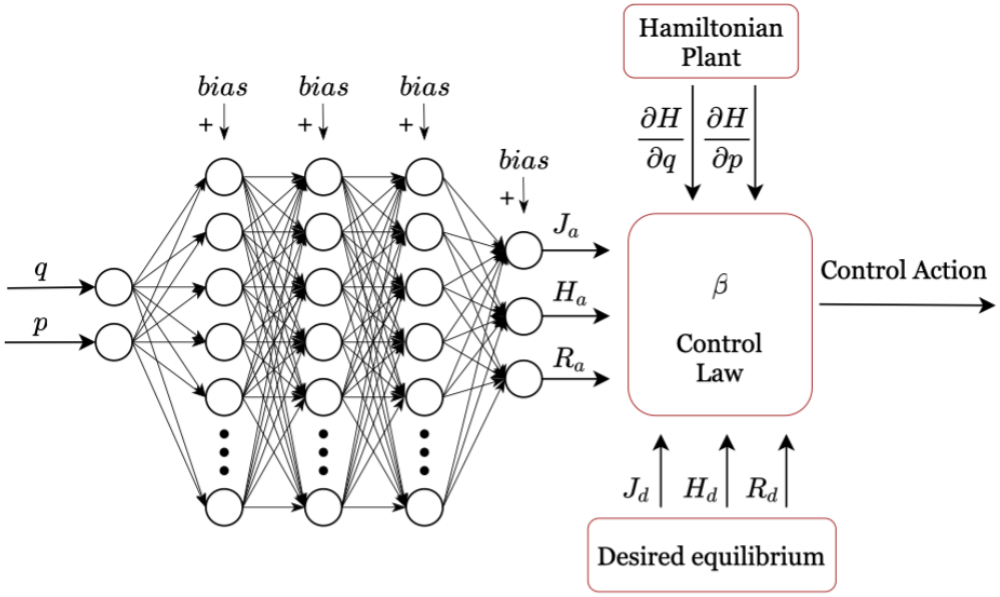


Figure 3: Neural IDA-PBC Controller

The design of the nonlinear controller is based on neural IDA-PBC methodology for pH systems proposed by Sanchez-Escalonilla et al. (2021). The significance of this method is that it facilitates control of any physical system described in the port-Hamiltonian framework. To solve the limitation of the theory that is encompassed by solving matching equations Physics Informed Neural Networks (PINNs) were exploited. Within the scope of this research, the objective is to stabilize the pendulum in the upright position. Furthermore, stabilizing the system at any desired angle is considered as well. Achieving zero steady-state error in controlled system is required. To validate the robustness of the proposed nonlinear controller,

transient response of the system with two different NN architectures were analysed. The visualisation of control idea is represented in Figure 3.

3.2 Physics informed neural network architecture

The neural networks approach implemented in this research was developed by Sanchez-Escalonilla et al. (2021). Initialization of parameters for training NN is done with the use of Glorot Normal Distribution. The error between the predicted and expected values is calculated after each mini-batch. Furthermore, the learning algorithm worked through the entire training data set 1000 times, namely epochs. As a result of trained neural networks, final values of weights and biases used for calculating auxiliary Hamiltonian (H_a) function are obtained. The function for calculating H_a used for simulations in MATLAB and SIMULINK can be found in Appendix A. Other important characteristics are mentioned in Table 2. Within the scope of this research, auxiliary interconnection and damping matrices were defined before the training.

Moreover, when dealing with DNN, defining the optimal amount of hidden layer plays crucial role in neural networks performance. If the amount of hidden layers is very large with regard to the complexity of the task, the problem known as overfitting occurs(Uzair and Jamil, 2020). This affects the training time, efficiency, and, most importantly, error propagation during the training process. On the contrary, when there are not enough hidden layers, underfitting appears. As a result, NN does not manage to deal with a given task. Therefore, within the scope of this research, two NN architectures were considered:

- one input layer, three hidden layers, and an output layer, with 20 nodes inside each hidden layer (2-20-20-20-1)
- one input layer, one hidden layer with 60 nodes, and one output layer (2-60-1).

3.3 Experimental setup

The interconnected parts are represented in Figure 4. Maxon DC motor generates power to move the axle with a gearing ratio of 4.4:1 to which the pendulum is connected. Hence, the motor translates the Voltage input to mechanical rotational force and is driven by the Kepco Operational Amplifier that, in turn, supplies the required voltage based on the control law from SIMULINK. To transfer the digital signal to a real-time experimental setup, dSpace Hardware is used. SIMULINK computes the required voltage by comparing the current state (in this case angle) of the inverted pendulum to the desired one. The current angle of the

Table 2: Characteristics of Physics Informed Neural Networks

Characteristic	Specification
Optimizer	ADAM, LBFGS
Parameter initialization	Glorot Normal Distribution
NN Class	Tensorflow Keras
Activation function	tanh
Number of Epoch	1000
Batch size	64
Learning rate	0.001

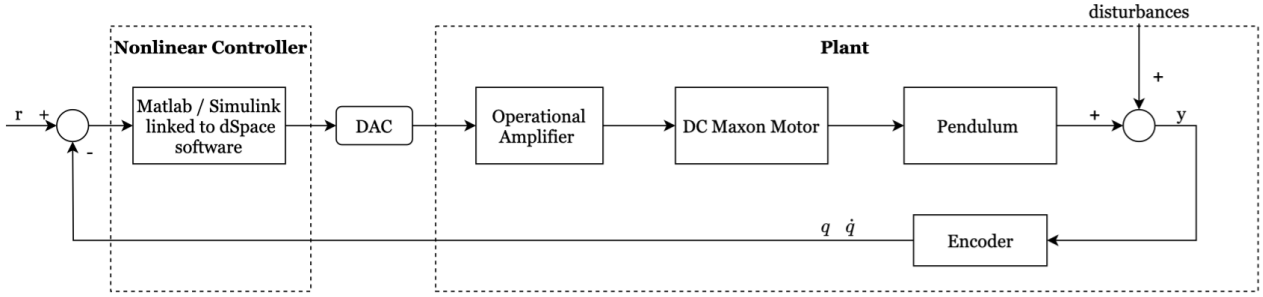


Figure 4: Closed loop representation of controlled plant

inverted pendulum is obtained by HEDL-5540 A02 encoder. Each of the components of the interconnected system is further described in this section and can be found in Appendix B.

3.3.1 Controller board/ Converter

For conducting the experiments with the real system and embedding the Neural IDA-PBC controller into the physical system, Digital Signal Processor (DSP) is required. In this case, dSpace CLP1104 Connector Panel for DS1104 with 8 ADC Inputs (4 Multiplex $\times 16 - bit$; $4 \times 12 - bit$) and 8 DAC Outputs ($16 - bit$) allows converting digital signal to analog and vise versa. The main function of DSP is to mediate between the controller in SIMULINK and the power amplifier by means of signal processing and enabling controller implementation.

3.3.2 Power Amplifier

The KEPCO BOP 36-12M Bipolar Operational Power Amplifier (Op-amp) can act as a Voltage or Current source for the appliance connected to it. Within the scope of this research, Voltage is used as a source. The saturation of Op-amp is ± 36 V and ± 12 A. Hence, 10 V supplied to the amplifier generates an output of 36 V, meaning that the conversion factor is

$0.2\dot{7}$. Similarly, both Current and Voltage read-back from the KEPCO BOP 36-12M Op-amp is 12 A/V and 36 V/V respectively KEPCOINC. (2018).

3.3.3 Sensor: Encoder

To determine the angle of the pendulum, Quick Assembly Two Channel Optical Encoder HEDL-5540 A02 with Line driver was used during the experiments. The resolution is approximately $N = 500$ cycles per revolution. The encoder is located on the back side of the DC motor which is connected to the pendulum via Planetary Gearhead GP 26 B with a 4.4:1 ratio. Hence, the angle at a certain moment can be calculated as follows:

$$q = \frac{2\pi}{4.4N}[rad] \tag{25}$$

3.3.4 Actuator: DC motor

RE 25 ø25 mm, Precious Metal Brushes CLL, 10 Watt Maxon DC motor 118748 was used as an actuator is the experimental setup. The voltage from the amplifier, according to the control law, is supplied to the DC motor. Hence, the electrical energy is translated to the torque that drives the pendulum. Characteristics of the motor in Table 3 are used for further calculations in Section 4.5 (MaxonAG, 2021).

Table 3: DC Maxon motor characteristics

Characteristic	Value	Units
Terminal resistance	32.6	Ω
Terminal inductance	3.48	mH
Torque constant	89.9	mNm/A
Speed Constant	106	rpm/A
Mechanical time constant	4.23	ms
Rotor inertia	10.5	gcm^2

3.4 Mathematical expression

When considering the interconnected system of the Pendulum and DC motor, such as in an experimental setup, there are three storing energy elements with physical states ϕ, p, q flux, generalized momentum, and angle correspondingly. Coupling of the electrical and mechanical systems occurring via coupling coefficient α: $\tau = \alpha i$, $V_{emf} = 1/\alpha\omega$, where τ is torque

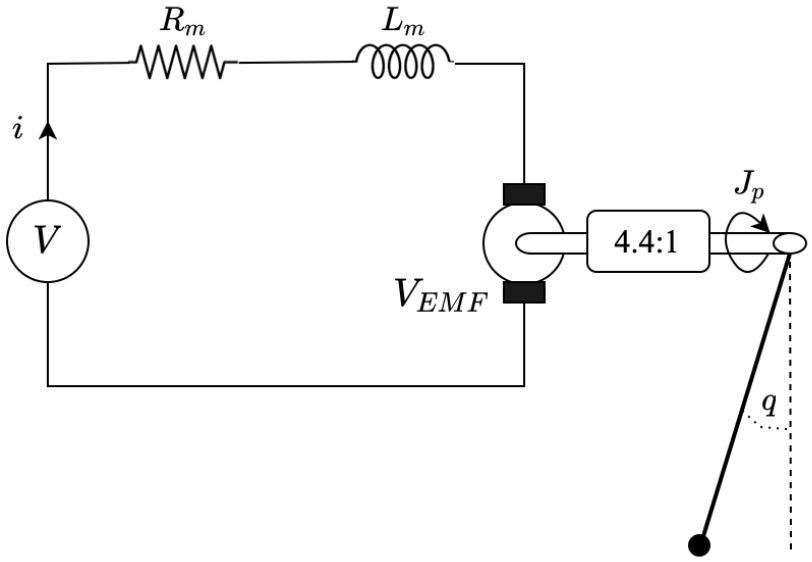


Figure 5: DC Motor Pendulum System

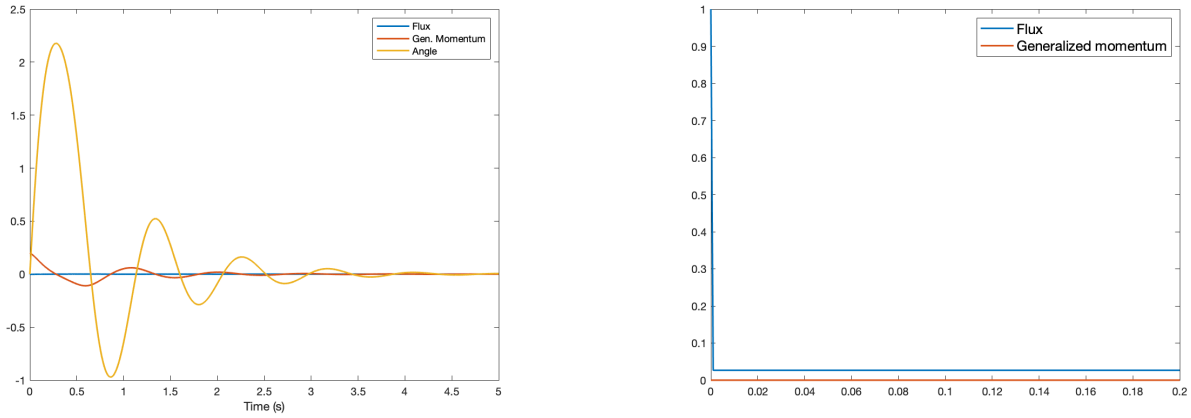
applied by DC motor, i is a current running through the circuit, V_{emf} is an electromagnetic force voltage, and ω is an angular velocity of the pendulum. For calculation simplification, it is assumed that $\Phi = \alpha\phi$. The Hamiltonian function then is:

$$H(\Phi, p, q) = \frac{1}{2L_m}\Phi^2 + \frac{1}{2J_p}p^2 - mglcos(q) \tag{26}$$

where L_p is an inductance, $J_p = I + ml^2$ is rotational inertia (sum of electrical and mechanical inertia).

$$\begin{aligned} \underbrace{\begin{bmatrix} \dot{\Phi} \\ \dot{p} \\ \dot{q} \end{bmatrix}}_{\dot{x}} &= \underbrace{\begin{bmatrix} -R & -1 & 0 \\ 1 & -\mathcal{D} & -1 \\ 0 & 1 & 0 \end{bmatrix}}_{J(x)-R(x)} \underbrace{\begin{bmatrix} \nabla_\Phi H(\Phi, q, p) \\ \nabla_p H(\Phi, q, p) \\ \nabla_q H(\Phi, q, p) \end{bmatrix}}_{\frac{\partial H}{\partial x}} + \underbrace{\begin{bmatrix} \alpha \\ 0 \\ 0 \end{bmatrix}}_{g(x)} u \\ y &= \underbrace{\begin{bmatrix} \alpha & 0 & 0 \end{bmatrix}}_{g^\top(x)} \underbrace{\begin{bmatrix} \nabla_\Phi H(\Phi, p, q) \\ \nabla_p H(\Phi, p, q) \\ \nabla_q H(\Phi, p, q) \end{bmatrix}}_{\frac{\partial H}{\partial x}}, \end{aligned} \tag{27}$$

However, considering the characteristics of the DC motor presented in an experimental setup, the system was analysed via the simulation. The results of the simulation can be



(a) DC Motor and Pendulum

(b) DC motor only

Figure 6: Response of the system to constant input

seen in Figure 6. On the left side, Figure 6a shows angle response (with present damping). However, it can be seen that flux has a very low amplitude; it oscillates around 0. Hence, the DC motor was modeled separately. Figure 6b represents the response of the flux to 300 V input. The conclusion has been made that the used DC motor is a stiff system with almost immediate response. Hence, it can be excluded from the modeling if the correct coupling between voltage input and response of the mechanical system is found.

Hence, the final mathematical expression of the system becomes:

$$\begin{aligned} \begin{bmatrix} \dot{q} \\ \dot{p} \end{bmatrix} &= \begin{bmatrix} 0 & (1+J_a) \\ -(1+J_a) & -(R+R_a) \end{bmatrix} \begin{bmatrix} \nabla_q H(q,p) \\ \nabla_p H(q,p) \end{bmatrix} + \begin{bmatrix} 0 \\ \alpha \end{bmatrix} \beta(x) \\ y &= \begin{bmatrix} 0 & \alpha \end{bmatrix} \begin{bmatrix} \nabla_q H(q,p) \\ \nabla_p H(q,p) \end{bmatrix}, \end{aligned} \tag{28}$$

where $\alpha = \frac{k_n}{R_m}$, where k_n is torque constant, R is resistance of the motor according to the Table 3. Thus, the Hamiltonian energy profile is expressed in terms of two states, namely angle and generalized momentum as can be seen in Figure 7.

3.5 Parameter acquisition

Next step towards control of the real physical system is defining the parameters involved. It also assists in comprehending the systems behaviour, and ensures the accuracy of the experimental data. As it was shown in section 3.4 some of the parameters have to be found via

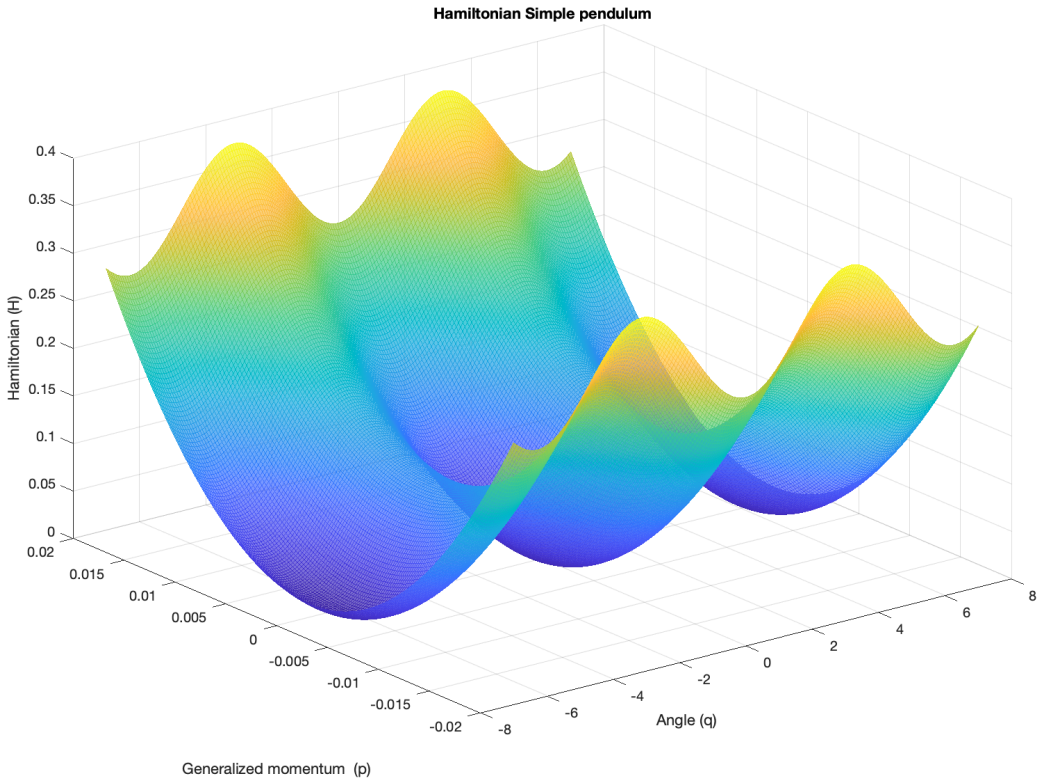


Figure 7: Hamiltonian Energy of the research system

conducting experiments and further analysing of data. This subsection covers acquisition of the required parameters.

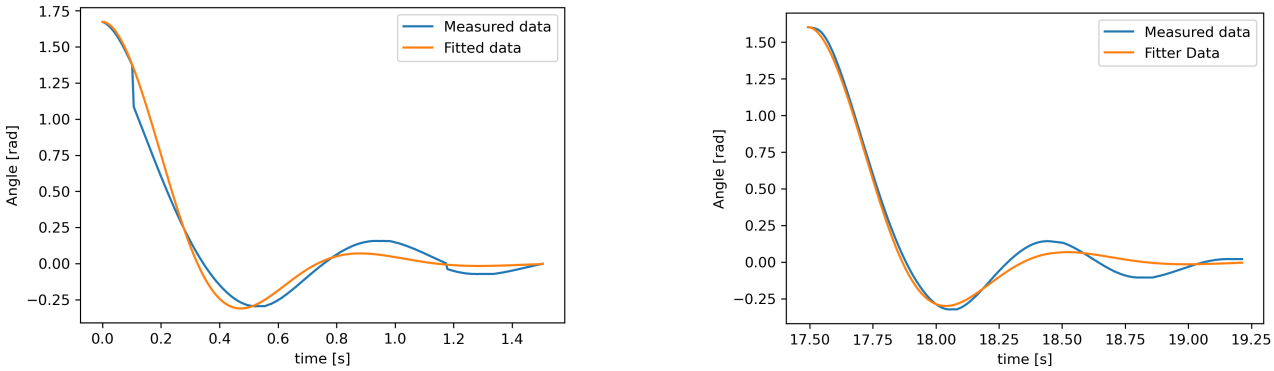
3.5.1 Pendulum

Since the mass of the pendulum bob is very low, the whole mass, considering both rod and bob jointly, was measured, and is equal to 42 g (0.042 kg). Furthermore, the length of the pendulum was measured to be 20.8 cm (0.0208 m). Taking complete mass of the pendulum complicates the task of defining the center of mass that is crucial for modeling. The approach taken for the final definition of mass and length is described in a further section.

3.5.2 Damping identification

To define the damping coefficient of the System, two experiments of releasing the pendulum from the 45 degrees position were conducted. Considering Newton's second law, a simple pendulum system equation of motion can be written as:

$$m_p l_p^2 \ddot{q} + d_p \dot{q} + m_p g_p l_p sin(q) = \tau \tag{29}$$



(a) Experiment 1

(b) Experiment 2

Figure 8: Fitting data in Python for damping identification

where m_p, l_p, g_p, d_p are the mass, length of the pendulum, gravitational constant, and damping coefficient respectively; q, $\dot{q}$, $\ddot{q}$ are the angle, angular velocity and acceleration of the pendulum; τ is applied force, considered to be zero in this case. Obtained data of angle measurement was used to approximate the parameters m_p, l_p and d_p of the differential equation (29) when input is equal to zero. Parameter fitting was done via Non-Linear Least-Squares Minimization and Curve-Fitting in Python. The results of the data fitting are depicted in Figure 8, where the blue line represents measured data and yellow is the fitted one.

Both analysis have shown the following results: mass m_p = 0.042 [kg], l_p length = 0.185 [m], d_p damping = 0.006 [Ns/m]. These values were used as parameters for further simulations and experiments.

Moreover, in Figure 8, it can be seen that the fitted data is not completely aligned with the measured one. It can be due to inconsistent friction throughout the pendulum trajectory. Moreover, in Figure 8a blue line is not smooth. Therefore, the conclusion can be made that nonlinearities are present in the friction. However, it is known that PBC is robust to passive unmodeled dynamics, such as friction (Romero et al., 2013)

3.5.3 Coupling coefficient

As it was mentioned in Section 3.4, the coupling coefficient between the voltage and torque is k_n/R, where k_n is a torque constant and R is the resistance of the motor. Hence, considering the characteristics of Maxon DC Motor in *Table 3* the coupling coefficient is given as $\alpha = 2.7576688$ [N/V].

To verify the coupling coefficient between the input voltage and mechanical system ten ex-

Table 4: Experimental data for coupling coefficient calculation

N	Voltage fed	Measured Voltage			Measured Angle			α
	[V]	[V]			[rad]			
		Trial 1	Trial 2	Average	Trial 1	Trial 2	Average	
1	1	0.9659	0.96679	0.966345	0.4332	0.43911	0.436155	0.0322
2	1.5	1.4591	1.458984	1.459042	0.6815	0.674	0.67775	0.0319
3	1.75	1.7068	1.7067	1.70675	0.8305	0.7947	0.8126	0.0316
4	2	1.9648	1.9648	1.9648	0.9997	0.9399	0.9698	0.0314
5	2.25	2.2141	2.2145	2.2143	1.1586	1.1564	1.1575	0.031

periments with Voltage pulse response were conducted. A pulse signal with different amplitudes, a period of 5 sec and 50% width (2.5 sec) was fed to the system. Considering equation 29 with torque $\tau = \alpha V$ and the range of recorded data when the velocity and acceleration of the pendulum are constant, the coupling coefficient can be found as $\frac{mglsin(q)}{V_{fed}}$. Recorded experimental data can be found in Table 4. The value of the coupling coefficient is given as $\alpha = 0.03162\ [mN/V]$.

Moreover, Figure 9 shows recorded experimental data of the system response to the fed pulse signal with an amplitude of 1.5 V and width of 5 sec. Again, it can be seen that the friction is not linear as spikes are present in velocity readouts. On the other hand, those can also be due to the noise in the digital encoder or eliminating the DC motor from the mathematical expression.

Finally, curve-fitting for pulse response of the system was done in PYTHON, while considering system as pH. Equation 28 with J_a, $R_a = 0$ was used for fitting when voltage is applied (Appendix C). The results can be seen in Figure 10.

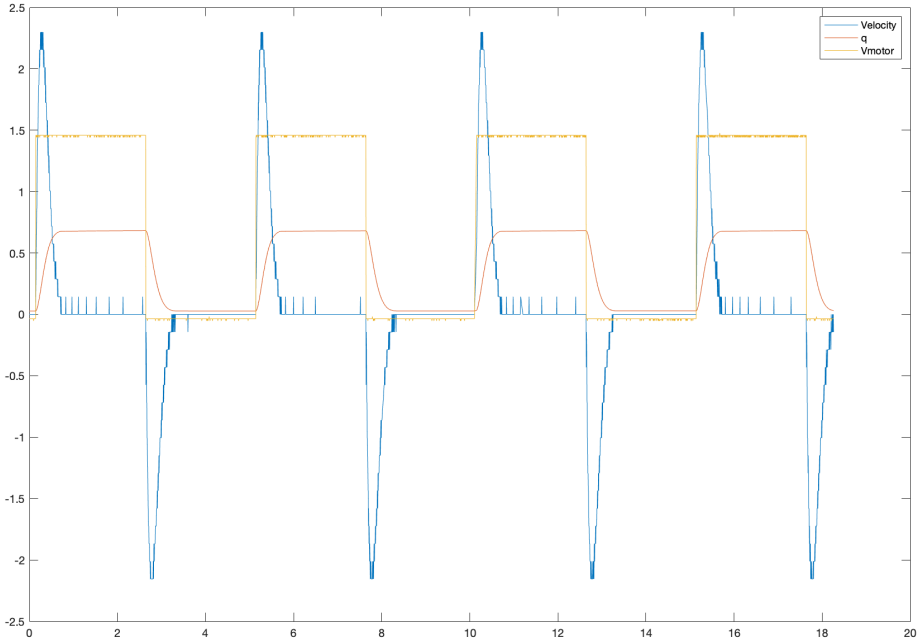


Figure 9: Experimental data: Voltage pulse input.

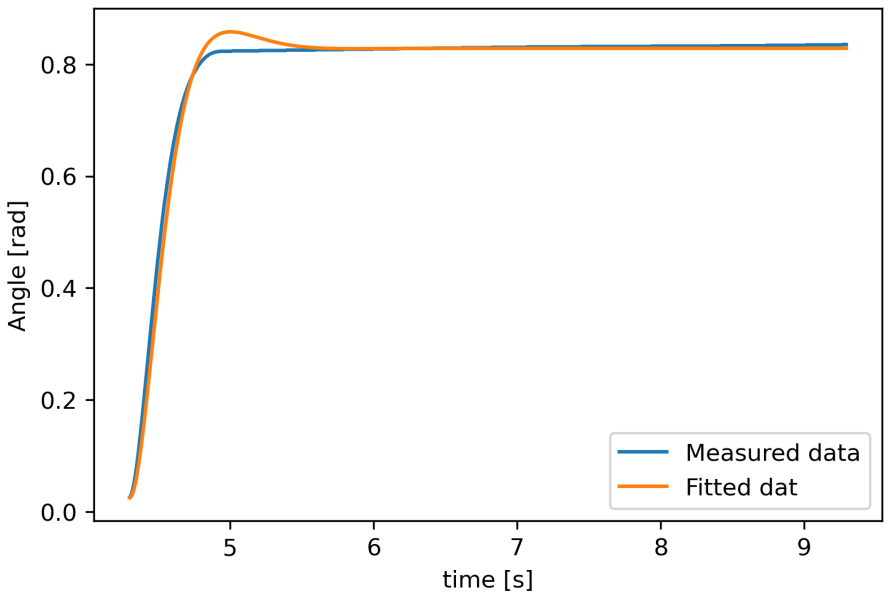


Figure 10: Fitting data for coupling coefficient definition. System response for 1.5 V input

4 Results

In this chapter we aim to illustrate the applicability of Neural IDA-PBC methodology via explaining the results of simulations and real life experiments of the nonlinear control for studied system. To meat the objective of the research the following analysis was done. Firstly, the performance of NN was compared to the nonlinear controller using analytical solution for the control law. Secondly, to see the effect of amount of hidden layers with respect to the complexity of the problem, two different architectures of DNN were considered, namely (2-20-20-20-1) and (2-60-1). Finally, the influence of the desired damping on the controller performance was analysed.

4.1 Towards design of nonlinear controller for a fully actuated system

The final control law was defined according to equation 13, and is given for research System as:

$$\beta(x) = \frac{1}{\alpha}\Big[-(1+J_a)\frac{\partial H}{\partial q} + \frac{\partial H}{\partial q} - (R+Ra)\frac{\partial H_d}{\partial p}\Big] \tag{30}$$

Modelling of the neural nonlinear controller was designed according to Neural IDA-PBC Methodology described in section 3.3. Prior to NN training $J_a(x)$, $R_a(x)$, initial conditions, desired equilibrium point, and defined parameters of physical system were fixed. Afterwards, neural network was trained to define the $Ha(\theta; x)$ function with weights and biases as parameters, and *tanh* as activation function.

Looking at the performance of two adopted NN architectures, it can be seen that both manage to estimate auxiliary Hamiltonian function (Table 5). However, one of the crucial factors is training time. DNN with only one hidden layer (2-60-1) can solve the task faster, even though the entire dataset needs to pass through more epochs before getting to sufficient final loss. In this setting, there is no crucial difference between both architectures. However, as the complexity of the problem increases (for instance, when dealing with under-actuated systems), the amount of hidden layer plays a significant role in terms of accuracy of the results and training time efficiency. Afterward, $H_a(\theta; x)$ is induced into the control law in SIMULINK.

Table 5: Training performance of two different NN architectures, $J_a = 0$, $R_a = 0$

Characteristic	[2-20-20-20-1]	[2-60-1]
Training time (s)	4.22	2.31
Training stopped at epoch	500	1000
Training loss at last epoch	0.0037	0.0030
Final validation loss at last epoch	0.0006	0.0016
Final loss ($\mathcal{L}$)	5.09177771e-05	1.4559766e-05

4.2 Simulation results

Before moving to the real life experiments the simulation of the NN IDA-PBC control was carried out both in PYTHON and MATLAB. It provides with a basis for predicting systems behaviour. As an intermediate step before moving to RTI, simulation with discrete time step of 0.001 sec and ode4 Runge-Kutta solver in SIMULINK took place. The simulation results can be found in Figure 11, while the model itself in Appendix D.

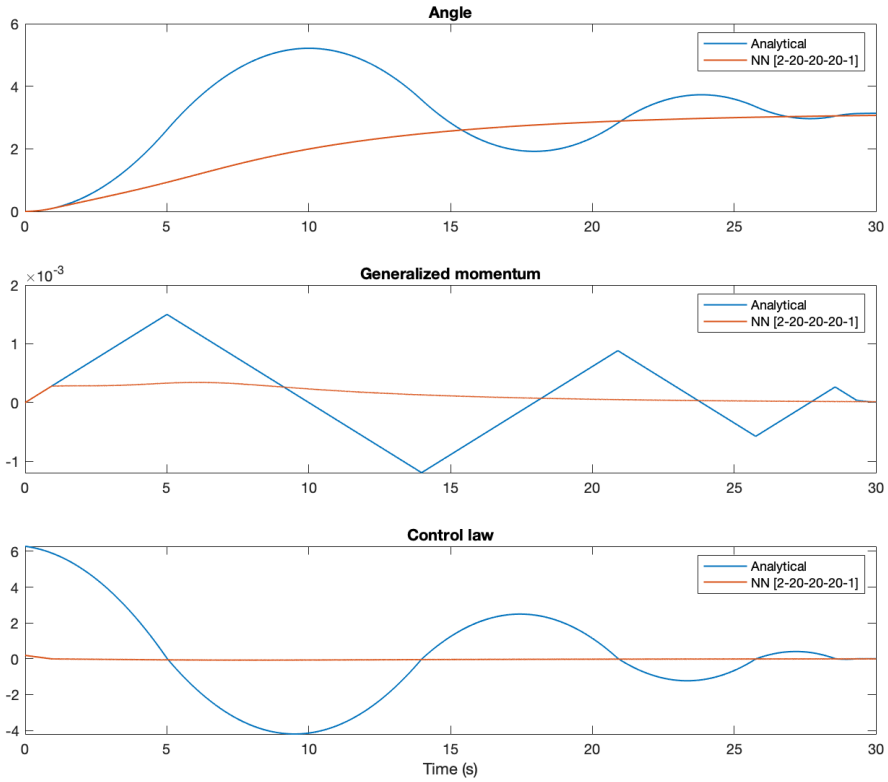
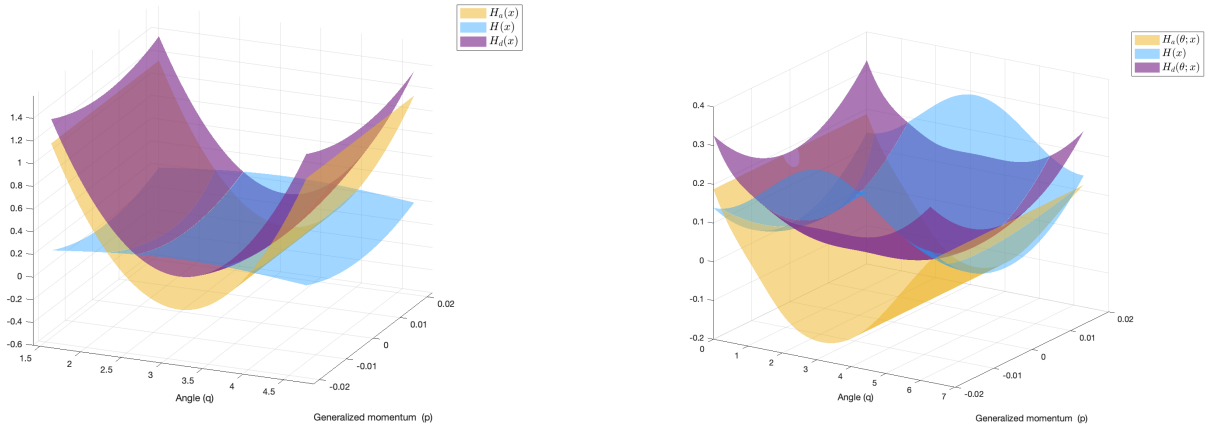


Figure 11: Simulation of System response and Control action in SIMULINK; $R_a = 1$, $J_a = 0$.



(a) Energy profile: Algebraic IDA-PBC

(b) Energy profile: Neural IDA-PBC

Figure 12: Numerical simulation of the pendulum Hamiltonian Energy profile for $J_a = 0$, $R_a = 1$, $H_d(x) = H_a(x) + H(x)$: a) Analytical solution that comprises pure potential energy compensation; b) $H_a(\theta; x)$ approximated by Neural Network with [2-20-20-20-1] architecture that fulfils matching equations of the system.

Parameters of the pendulum, excluding coupling coefficient, defined in previous section were used. Since the DC motor is left out from simulation results differ from real life implementation. Furthermore, because the generalized momentum of a real physical system is very low, it was necessary to implement Saturation block, which caused such a rough behaviour for analytical solution in the middle graph of Figure 11. Even though in both cases system reaches to the desired equilibrium, NN control outperforms analytical case in terms of overshoot and settling time.

Moreover, the analysis of the Hamiltonian energy profile (Figure 12) again demonstrates that Analytical IDA-PBC encompasses potential energy compensation only. Even though it does not impede control of fully actuated systems, full energy compensation is required for underactuated systems.

4.3 Experimental results

The aforementioned simulations show the capability of Neural IDA-PBC. This allows the researcher to proceed with a swing-up pendulum problem in real life. For this, the setup in the DTPA lab was used. Similar to the previous section, the performance of Neural control was compared to Algebraic IDA-PBC on the studied System. Finally, the effect of the auxiliary damping matrix was studied by conducting experiments with different values.

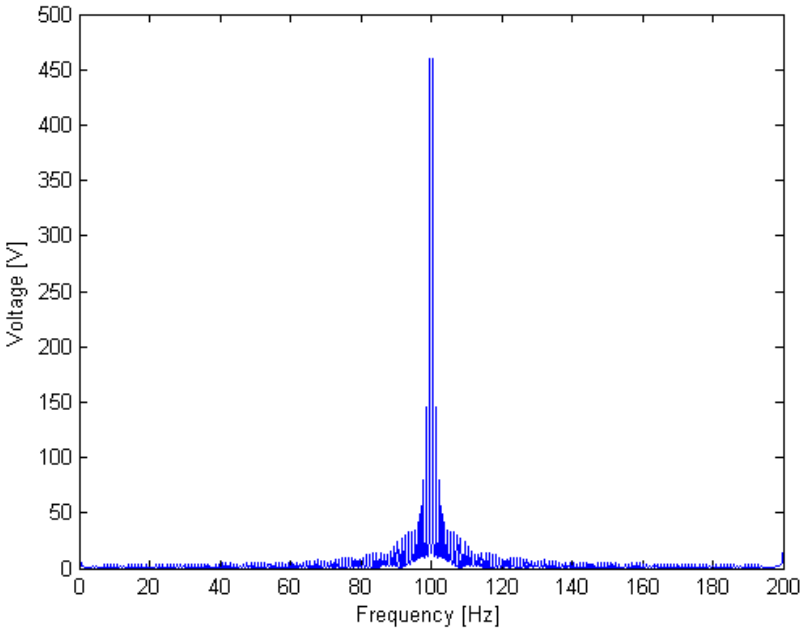


Figure 13: Fast Fourier Transform of recorded Voltage signal

Nevertheless, noise was detected during the first trials for implementing control for experimental setup. It led to disturbances of the control of such a sensitive system. Hence, there is a need for low-pass filter in a SIMULINK model. To identify the stopband frequency, Fast Fourier Transform (FFT) was applied to the recorded voltage signal. As a result, the noise was detected at 100 [Hz] (Figure 13).

4.3.1 Algebraic vs Neural IDA-PBC

In this subsection the performance of the Neural nonlinear controllers is compared to Algebraic IDA-PBC with analytical solution for control law. Since the main requirement for Algebraic IDA-PBC is to have desired Hamiltonian quadratic in increments (Section 2.3), the arbitrary gain for its partial derivative can be chosen. Here, standard gain ($c = 2$) was used, and it shows that this variant cannot manage to bring the pendulum to the desired equilibrium. Control action in this case depends purely on potential energy of the System, as it can be seen in a third graph of Figure 14. Hence, depending on initial gain, system gets enough momentum to get to the desired position. The decision has been made to double the gain, which is represented as a red line in Figure 14. Moreover, given non-parameterized IDA-PBC methodology, auxiliary interconnection and damping were fixed prior the control implementation. Within the frame of this analysis $J_a = 0$ and $R_a = 0.25$.

Furthermore, the results of the Neural control show that both NN[2-20-20-20-1] and NN[2-

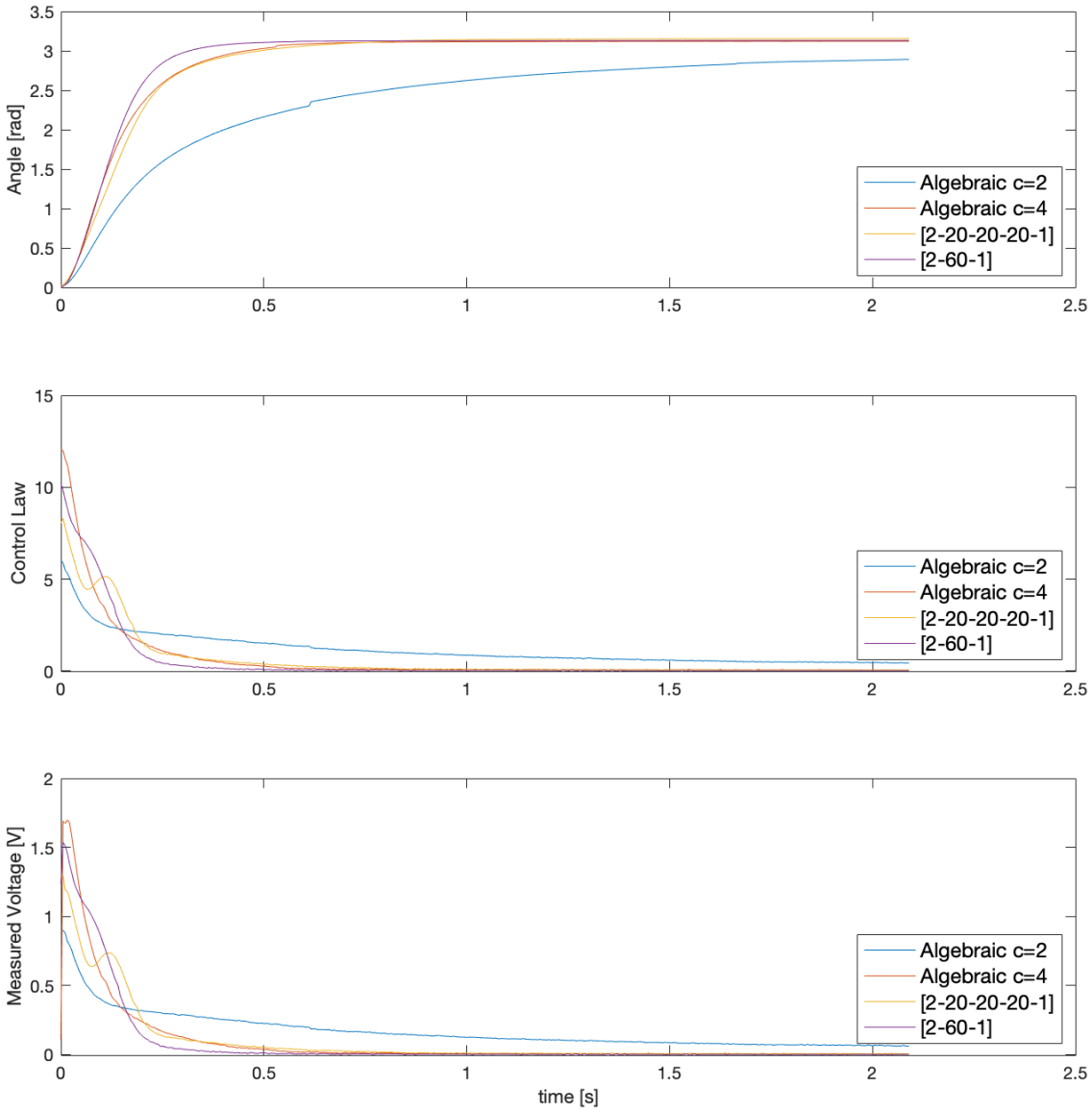


Figure 14: System's transient response and Control laws; $J_a = 0$, $R_a = 0.25$

60-1] manage to deal with a task, however, DNN with one hidden layer has an advantage in raising and settling time. Besides, the control law (Figure 14) is more smooth. In case of DNN with three hidden layers, the trajectory of the pendulum looks similar to Analytical control with $c = 4$, however, it can be clearly seen that the control action compensated both kinetic and potential energy. On the other hand, when moving towards the implementation of a control law in real life, for instance in case of a automatized factory, the amount of power used to drive the system plays crucial role as it leads to expenses. In the measured Voltage graph of Figure 15 it can be seen that NN [2-20-20-20-1] requires the least amount of voltage to bring the pendulum to the desired equilibrium.

4.3.2 Damping Analysis

One of the features of non-parameterized IDA-PBC is the variety of options for tuning the auxiliary interconnection and damping depending on the design requirements. This section aims to analyse the effect of assigning different desired damping for the controller since closed loop system behaviour highly depends on this parameter. The main criteria for analysing the system's response to the controller and, as a result, checking controllers performance are: raising time, overshoot, and settling time. Hence, the objective is to determine the optimal desired damping that keeps the system from high overshooting and does not make the system underdamped. For this, the transient response of the system was analysed while applying four different values for auxiliary damping R_a while $J_a = 0$ as it can be seen in Figure 15.

The measured data is compiled in Table 6. Low auxiliary damping leads to high overshoot (14.33%). While in the case of $R_a = 0.25$, the system is critically damped and settles within 0.6 sec. If the damping value is increased, high kinetic energy counteraction occurs, which stops the pendulum from reaching desired equilibrium.

Table 6: Transient response characteristics depending on desired damping.

Experiment	Raising time T_r $[s]$ (90%)	Overshoot M %	Settling time T_s $[s]$ (2%)
$R_a(x) = 0.01$	0.475	14.33	0.55
$R_a(x) = 0.25$	0.31	0.01	0.6
$R_a(x) = 0.5$	0.8	0	1.8302
$R_a(x) = 1$	1.135	0	-

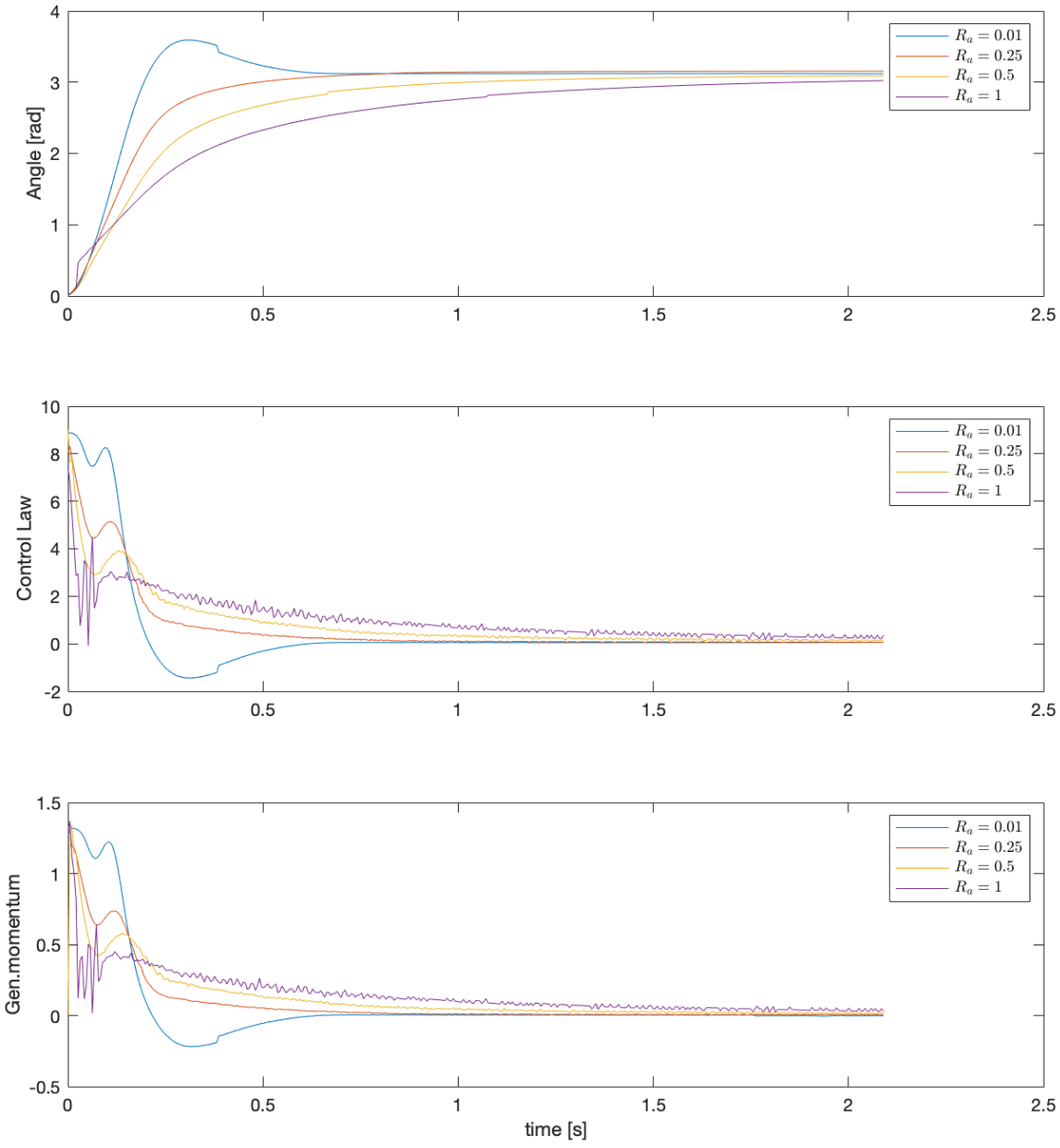


Figure 15: Experimental results: different desired damping used

5 Discussion

5.1 Results

This study demonstrates the experimental results of nonlinear control for pendulum using the Neural IDA-PBC. Hence, the results of this study unravel the method of implementation of a neural nonlinear control to fully actuated systems. It elaborates on the behaviour of the controlled system when two variants of DNN are used for approximating the solution of the matching equations required to fulfill closed-loop stability. For this problem, both NN[2-20-20-20-1] and NN[2-60-1] control-informed neural networks managed the task. However, it can be concluded from the results that one hidden layer is enough to ensure the fastest rising and settling time for the swing-up pendulum. The conclusion can be made that the architecture of NN has to be chosen with regard to the complexity of the system.

In this setting, the desired damping, as one of the tuning control parameters, was analysed via conducting experiments with different damping. Since the desired damping is included in the training of NN, this approach is not efficient when working with more complex problem where training requires more time or computational power. Moreover, even though considering the studied system analytical IDA-PBC is cooping with the task, the NN IDA-PBC controller still outperforms it.

In general, IDA-PBC methodology shows that nonlinear system is controlled by shaping its closed-loop energy while respecting the dynamics of the original system. Inherently, this method can assure high-performance and, most importantly, cost-efficient controllers.

5.2 Limitations

Reflecting on the result of this work, a few limitations can be observed. Firstly, the findings cannot be extrapolated to any system, as IDA-PBC has a high dependency on the model dealt with. Even though this experimental work shows the potential of the Neural IDA-PBC for fully actuated systems, the implementation of the method is highly constrained by the software and hardware used in this research. Furthermore, the studied mechanical system was modeled by relying on physics principles, such as Newton's law or other realistic assumptions. However, within the scope of this research, the modeling of the DC motor was omitted considering that the system is stiff. Hence, if high precision is required and the parameters of the mechanical system are bigger at scale, it might be important to include the DC motor in the modeling, which significantly changes the problem. Moreover, parameter acquisition was done via experiments, and an assumption has been made, such as in

the case of damping identification. However, most uncertainties usually come from inertial parameters (generalized momentum directly depends on damping in the case of a simple pendulum) and can deteriorate the performance of the controller.

5.3 Future work

From the results of this work, it can be seen that there are some aspects that can be addressed in future work. Firstly, training time and accuracy of the neural networks can be significantly reduced by introducing hard constraints into PINNs by means of penalty method Lu et al. (2021). Secondly, throughout this research controller parameter, namely auxiliary damping, was analysed by applying different values. However, a method to define optimal controller parameters can be obtained bypassing the whole training procedure. For instance, Linear Quadratic (LQ) optimal control can be used Vu and Lefèvre (2018). Finally, this research has shown the method of implementation Neural IDA-PBC controller for fully actuated system. The further study on under actuated systems need to be conducted (Sanchez-Escalonilla et al., 2022).

6 Conclusion

From the theoretical perspective, the IDA-PBC methodology allows for stabilizing a wide spectrum of systems modeled in the port-Hamiltonian framework. Likewise, it provides a clear interpretation of the control scheme in terms of energy. However, certain obstacles hinder its application. The first one regards solving partial differential equations, which are required for preserving closed-loop stability. Another obstacle is a high dependency on the physical system dealt with. Within the scope of this research, the Neural IDA-PBC method that proposes eliminating the difficulty of solving the matching equations was analysed.

Towards the end of this research, the mathematical expression of the nonlinear controller for a given system based on Neural IDA-PBC was defined. To examine the performance of the proposed controller another case was considered as well: the analytical solution of Algebraic IDA-PBC. Furthermore, the methods for the acquisition of unknown pendulum parameters, such as mass length and damping coefficient, were described. Besides the parameters of the mechanical system, the interconnection and coupling of separate parts required for experimenting were done.

The verification of Neural IDA-PBC applicability took place via experimenting on a fully actuated system, namely a pendulum. The effect of critical controller parameters, such as the desired damping, was analysed. Moreover, two different architectures with different amounts of hidden layers were considered. It assisted in correlating the system's complexity to the depths of neural networks. As it was enough to use deep neural networks with one hidden layer to get exceptional results, it shows that this methodology has the potential to deal with more complex systems. Moreover, obtained results were compared to an analytical solution for IDA-PBC control that offers pure potential energy compensation. Even though for fully-actuated systems it is enough to implement an analytical solution of Algebraic IDA-PBC, Neural IDA-PBC outperformed it.

The studied approach highly depends on the system to be controlled, so the results of this research cannot be generalised and extrapolated to apply to other physical systems. Nevertheless, it shows great opportunity for dealing with nonlinear systems in terms of respecting model dynamics and its robustness to unmodelled nonlinearities, such as friction.

7 Bibliography

Bai, G. (2021), 'Physics informed neural networks (pinns) for approximating nonlinear dispersive pdes', *Journal of Computational Mathematics* **39**(6), 816–847.

Chen, Y., Lu, L., Karniadakis, G. E. and Dal Negro, L. (2020), 'Physics-informed neural networks for inverse problems in nano-optics and metamaterials', *Optics Express* **28**(8), 11618.

Duindam, V., Macchelli, A., Stramigioli, S. and Bruyninckx, H. (2014), *Modeling and control of complex physical systems the port-hamiltonian approach*, Springer Berlin.

Fujimoto, K. and Sugie, T. (2001), 'Canonical transformation and stabilization of generalized hamiltonian systems', *Systems & Control Letters* **42**(3), 217–227.

Goodfellow, I., Bengio, Y. and Courville, A. (2017), *Deep learning*, The MIT Press.

Hornik, K., Stinchcombe, M. and White, H. (1989), 'Multilayer feedforward networks are universal approximators', *Neural Networks* **2**(5), 359–366.

Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikaris, P., Wang, S. and Yang, L. (2021), 'Physics-informed machine learning', *Nature Reviews Physics* **3**(6), 422–440.

Karumuri, S., Tripathy, R., Bilionis, I. and Panchal, J. (2020), 'Simulator-free solution of high-dimensional stochastic elliptic partial differential equations using deep neural networks', *Journal of Computational Physics* **404**, 109120.

KEPCOINC. (2018), *BOP (M) (D) 100W, 200W, 400W BIPOLAR OPERATIONAL POWER SUPPLY*, FLUSHING, NY. 11355 U.S.A.
URL: $https://www.kepcopower.com/support/bop-operator-r8.pdf$

Lagaris, I., Likas, A. and Fotiadis, D. (1998), 'Artificial neural networks for solving ordinary and partial differential equations', *IEEE Transactions on Neural Networks* **9**(5), 987–1000.

Lu, L., Pestourie, R., Yao, W., Wang, Z., Verdugo, F. and Johnson, S. G. (2021), 'Physics-informed neural networks with hard constraints for inverse design', *SIAM Journal on Scientific Computing* **43**(6).

MaxonAG (2021), *Catalog: Re 25 ø25 mm, precious metal brushes CLL, 10 watt*.
URL: $https://www.maxongroup.com/medias/sys_master/root/8881621893150/EN-21-144.pdf$

Mishra, S. and Molinaro, R. (2021), 'Estimates on the generalization error of physics-informed neural networks for approximating a class of inverse problems for pdes', *IMA Journal of Numerical Analysis* **42**(2), 981–1022.

Nageshrao, S., Lopes, G., Jeltsema, D. and Babuška, R. (2014), 'Interconnection and damping assignment control via reinforcement learning', *IFAC Proceedings Volumes* **47**(3), 1760–1765.

Nunna, K., Sassano, M. and Astolfi, A. (2013), 'Constructive interconnection and damping assignment for port-controlled hamiltonian', *2013 American Control Conference* .

Ortega, R. and Spong, M. W. (2000), 'Stabilization of underactuated mechanical systems via interconnection and damping assignment', *IFAC Proceedings Volumes* **33**(2), 69–74.

Ortega, R., van der Schaft, A., Maschke, B. and Escobar, G. (2002), 'Interconnection and damping assignment passivity-based control of port-controlled hamiltonian systems', *Automatica* **38**(4), 585–596.

Raissi, M., Perdikaris, P. and Karniadakis, G. (2019), 'Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations', *Journal of Computational Physics* **378**, 686–707.

Romero, J. G., Donaire, A. and Ortega, R. (2013), 'Robust energy shaping control of mechanical systems', *Systems 'I&'amp; Control Letters* **62**(9), 770–780.
URL: *https://www-sciencedirect-com.proxy-ub.rug.nl/science/article/pii/S0167691113001199*

Sanchez-Escalonilla, S., Reyes-Baez, R. and Jayawardhana, B. (2021), 'Total energy shaping with neural interconnection and damping assignment – passivity based control'.
URL: *https://arxiv.org/abs/2112.12999*

Sanchez-Escalonilla, S., Reyes-Baez, R. and Jayawardhana, B. (2022), 'Stabilization of underactuated systems of degree one via neural interconnection and damping assignment – passivity based control'.

Schaft, A. v. d. and Jeltsema, D. (2014), *Port-Hamiltonian Systems theory:an introductory overview*, Vol. 1, Now Foundations and Trends.

Uzair, M. and Jamil, N. (2020), Effects of hidden layers on the efficiency of neural networks, *in* '2020 IEEE 23rd International Multitopic Conference (INMIC)', pp. 1–6.

van der Schaft, A. (2006), 'Port-hamiltonian systems: An introductory survey', *Proceedings of the International Congress of Mathematicians Madrid, August 22–30, 2006* p. 1339–1365.
URL: *http://www.ems-ph.org/doi/10.4171/022-3/65*

Venkatraman, A. (2010), *Control of Port-Hamiltonian Systems: Observer Design and Alternate Passive Input-output Pairs*, GrafiMedia, University of Groningen, The Netherlands.

Vu, N. T. and Lefèvre, L. (2018), 'A connection between optimal control and ida-pbc design', *IFAC-PapersOnLine* **51**(3), 205–210.

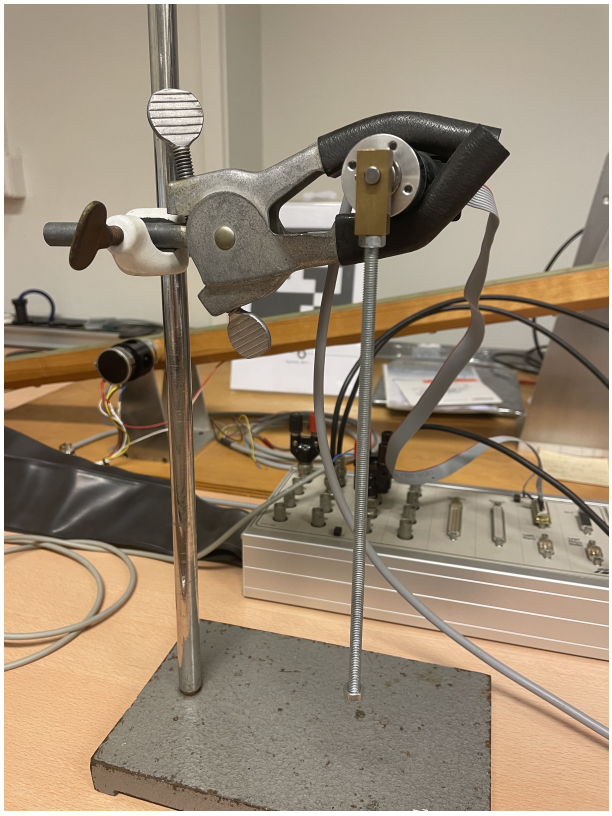
A H_a MATLAB function

```
function Ha= fcn(q, data1, data2, data3, p, data7, data6, data4, data5, data8)
% Simulink function for calculating Ha based NN data

firstLayer = zeros(1,20);
secondLayer = zeros(1,20);
thirdLayer = zeros(1,20);

for n=1:20
    firstLayer(n) = tanh(q*data1(1,n)+p*data1(2,n)+data2(n));
end
for i=1:20 %from neuron i in the second layer
    dummy = 0;
    for j = 1:20 %to neuron j in the first layer
        dummy = dummy + firstLayer(j)*data3(j,i); % weight from neuron
        % i (1st layer) to neuron j (2nd layer) multiplied by the output
        % of neuron i plus the bias in neuron j
    end
    secondLayer(i) = tanh(dummy + data4(i));
end
for i=1:20
    dummy = 0;
    for j = 1:20
        dummy = dummy + secondLayer(j)*data5(j,i);
    end
    thirdLayer(i) = tanh(dummy + data6(i));
end
dummy = 0;
for n=1:20
    dummy = dummy + thirdLayer(n)*data7(n);
end
a = (dummy + data8);
Ha = a; % k is for p, l is for q
```

B Experimental setup



(a) Pendulum connected to Maxon DC motor

(b) Digital Encoder

Figure 16: Pendulum and Encoder

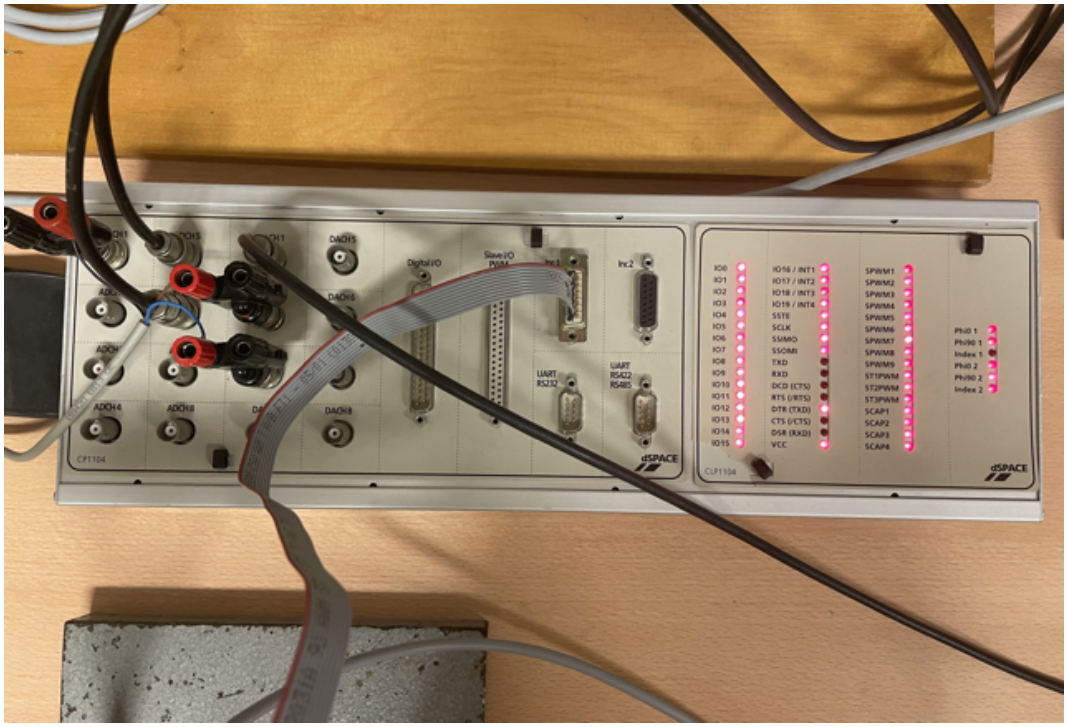


Figure 17: dSpace

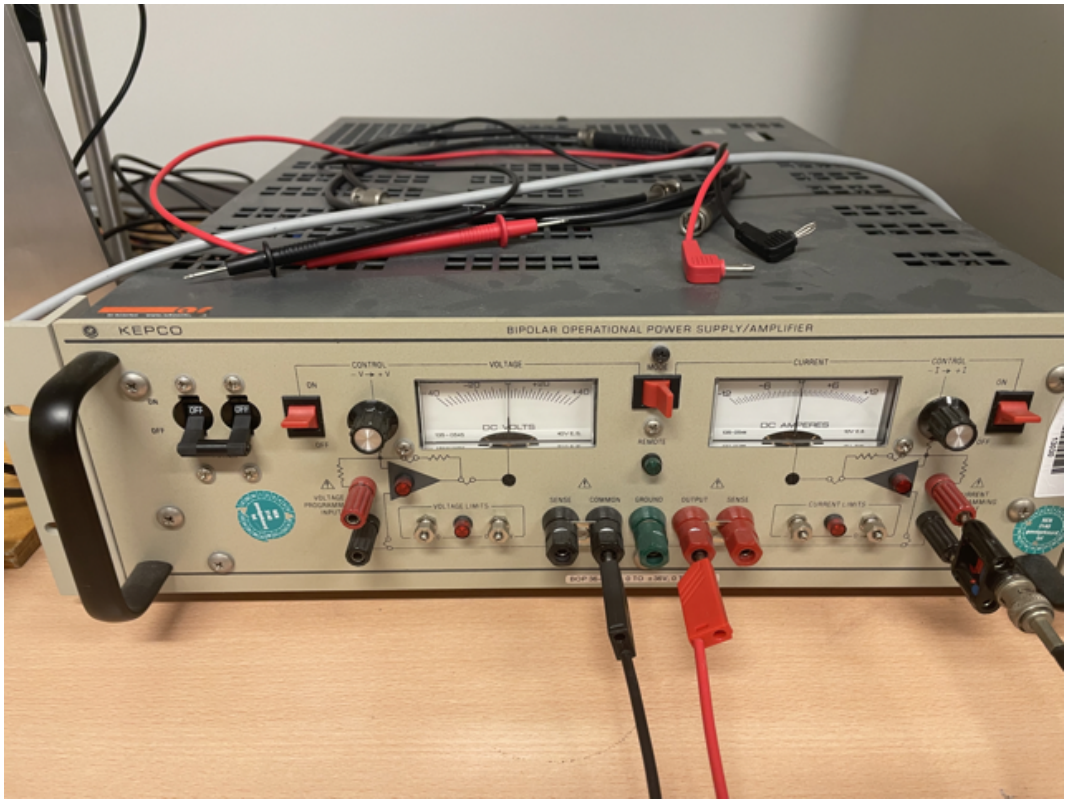


Figure 18: KEPCO Op-Amp

C Python code for Data Fitting

```
"""
Created on Mon May 30 09:35:18 2022
@author: veronika
"""
import numpy as np
import matplotlib.pyplot as plt
import scipy.io
from scipy.integrate import odeint
from lmfit import minimize, Parameters, Parameter, report_fit

dataFitting = scipy.io.loadmat('VoltageExperiment.mat')
t = dataFitting['t'][0]
y = dataFitting['Angle'][0]
# Plot measured Data
plt.figure(dpi=300)
plt.plot(t, y, label='Measured data')

def f(y, t, paras):
    """
    Port-Hamiltonian system
    """
    g = 9.81

    x1 = y[0]
    x2 = y[1]

    d = paras['d'].value  # damping
    m = paras['m'].value  # mass
    l = paras['l'].value  # length
    a = paras['a'].value  # coupling

    # States x=[q,p] Angle & Gen momentum
    f1 = x2/(m*l**2)  # q
    f0 = - m*g*l*np.sin(x1) - d*x2/(m*l**2)+ a*1.75 #p
    return [f1, f0]

def g(t, x0, paras):
    """
    Get the solutinon for ODE
    """
```

```
41      x = odeint(f, x0, t, args=(paras,))
42      return x
43
44  def residual(paras, t, data):
45      """
46      Compute the residual
47      """
48      x0 = [data[0], 0]
49      model = g(t, x0, paras)
50      x2_model = model[:, 0]
51      return (x2_model - data).ravel()
52
53  # initial conditions
54  x10 = 0
55  x20 = 0
56  y0 = [x10, x20]
57
58  # Set the boundaries for parameters (constraints are based on measurements)
59  params = Parameters()
60  params.add('l', value=0.208, min=0.104, max=0.3 ) #vary=False
61  params.add('m', value=0.042, vary =False)
62  params.add('d', value=0.01, min=0.0, max=2)
63  params.add('a',value=0.03, min = 0,max=5)
64
65  #  Model fitting
66  result = minimize(residual, params, args=(t, y), method='leastsq')
67  x0 = [y[0], 0]
68  x = g(t, x0, result.params)
69
70  # Plot fitted data over Measured
71  plt.plot(t, x[:,0], label='Fitted dat')
72  plt.xlabel('time [s]')
73  plt.ylabel('Angle [rad]')
74  plt.legend()
75
76  d = round(result.params['d'].value,3)
77  m = round(result.params['m'].value,3)
78  l = round(result.params['l'].value,3)
79  a = round(result.params['a'].value,3)
80
81  # Print the results
82  print(f"mass = {m}, length = {l}, damping = {d}, coupling={a}")
```

D SIMULINK simulation models

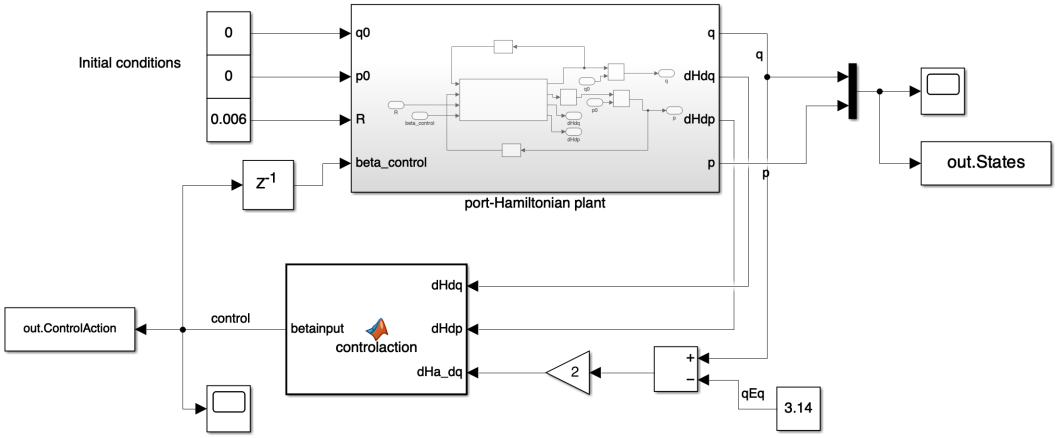


Figure 19: Simulink model for analytical solution

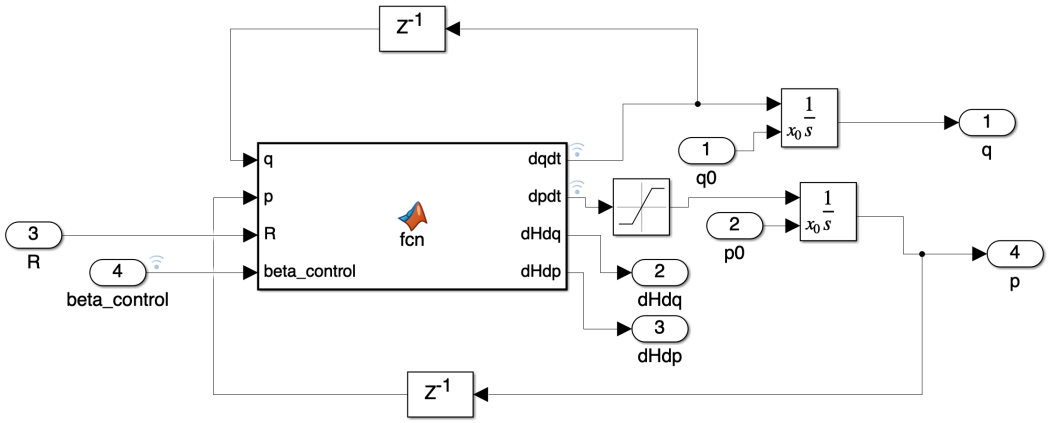


Figure 20: Simulink model for analytical solution: port-Hamiltonian plant

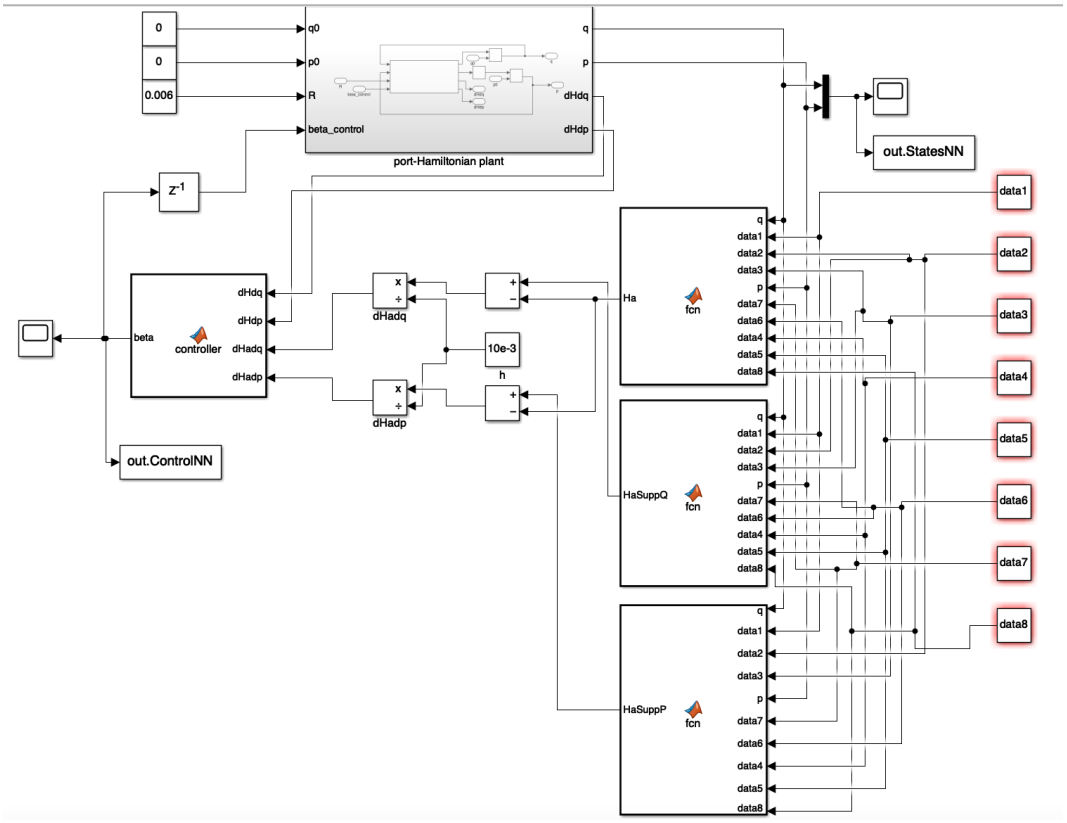


Figure 21: Simulink model for Neural IDA-PBC: port-Hamiltonian plant