Hybrid Mechanistic + Neural Model of Laboratory Helicopter

Christopher Rackauckas[1], Roshan Sharma[2], and Bernt Lie[2]
[1]Massachusetts Institute of Technology, USA; me@ChrisRackauckas.com
[2]University of South-Eastern Norway, Norway; Bernt.Lie@usn.no

Version of July 3, 2020

Contents

1 Introduction **3**

2 Helicopter mechanistic model **5**
2.1 Laboratory helicopter 5
2.2 Geometry of helicopter 5
2.3 Kinetic energy of helicopter 6
2.4 Potential energy of helicopter 7
2.5 Helicopter torques 7
2.6 DAE formulation of model 8
2.7 ODE formulation of model 8
2.8 Model with pivot in body-fixed origo 9
2.9 Simulation of models 9

3 Preliminary model fitting **11**
3.1 Experimental data 11
3.2 Preliminary model fitting 11

4 Hybrid model **14**
4.1 Neural torque with input dependence 14
4.2 Neural torque with input + state dependence 17
4.3 Equation discovery 17

5 Conclusions and Future work **18**

A Mechanistic model development **21**
A.1 Kinetic energy 21
A.2 Lagrangian and momenta gradients 22

B Nominal parameters and operating conditions **23**

References **23**

List of Figures

1 Laboratory helicopter, Sharma (2020). 5
2 Helicopter in side profile, pitch angle θ. 6
3 Helicopter in birds-eye view, yaw angle ψ. 6
4 Pitch angle θ and yaw angle ψ for the operating conditions with DAE model and with ODE model. 9
5 Pitch angular velocity ω_θ and yaw angular velocity ω_ψ for the operating conditions with DAE model and with ODE model. 10
6 Pitch angular momentum p_θ and yaw angular momentum p_ψ for the operating conditions with DAE model and with ODE model. 10
7 Pitch angle θ and yaw angle ψ for the operating conditions with DAE model and with RS model. 11
8 Pitch angular velocity ω_θ and yaw angular velocity ω_ψ for the operating conditions with DAE model and with RS model. 12
9 Pitch motor voltage u_θ and yaw motor u_ψ experiments. 12
10 Pitch angle θ and yaw angle ψ from experimental conditions. 13
11 Pitch angle θ and yaw angle ψ from experimental conditions with nominal model parameters. 13
12 Pitch angle θ: comparing model angle with optimized parameters (blue, solid) and measured experimental angle. 14
13 Yaw angle ψ: comparing model angle with optimized parameters (blue, solid) and measured experimental angle. 15
14 Pitch angle with torque modeled as in Eq. 2 with additive term $\mathrm{FNN}(u;p)$. 16
15 Yaw angle with torque modeled as in Eq. 2 with additive term $\mathrm{FNN}(u;p)$.. 16
16 Pitch angle with $\mathrm{FNN}(u,\mathsf{x},\mathsf{v};p)$. 17
17 Yaw angle with $\mathrm{FNN}(u,\mathsf{x},\mathsf{v};p)$. 18
18 Pitch angle with regression approximation of $\mathrm{FNN}(u;p)$. 19
19 Yaw angle with regression approximation of $\mathrm{FNN}(u;p)$. 19

List of Tables

1 Fittet parameters for laboratory helicopter. 15
2 Parameters for laboratory helicopter. 23
3 Initial operating conditions for laboratory helicopter. 23

Abstract

This document describes some "challenge problems" for Scientific Machine Learning (SciML) and Control in Julia. Key in the challenge problem is a laboratory helicopter model which is attached to the ground; the helicopter can not levitate from the ground. The laboratory helicopter has 2 propellers with independent voltages (control inputs), and the goal is to keep control of the measured pitch angle and yaw angle.

The challenges described in this paper are at three levels. (i) How can existing Julia tools be used to find the best solutions to a number of scientific computing problems? (ii) How can the methods to solve the various challenges be presented in the best pedagogical way, and inspire to further learning of Julia? (iii) How can the challenges inspire for further development in the fields of scientific learning and control?

In the paper, the laboratory helicopter is presented, and a relatively simple dynamic model based on Lagrangian mechanics is detailed. The controller torque is linear in the voltages, and the friction torque is linear in the angular velocities. Nominal model parameters are suggested, and the model/solution in Julia is used to verify that the model is reasonable. Next, experimental data from a laboratory process is presented, and it is shown that the model with nominal parameters gives poor model fit. A global time loss function for model fit is described, and optimal model parameters and initial conditions are suggested. The resulting model gives ok (but not perfect) fit to the experimental pitch angle measurements, but relatively poor fit to the yaw angle measurements. One possible explaination for the poor yaw angle fit is that there may be a deadband in the controller torques. Next, instead of assuming linear controller torque in the voltages, the controller torque is described by a forward neural network block. The result is quite good fit for both the pitch angle and the yaw angle.

With the given initial work and with published experimental data and video description of the helicopter, a number of challenge problems are proposed.

Keywords: Neural differential equations, mechanistic model, hybrid data-driven and mechanistic model, helicopter model, control relevant model

1 Introduction

Today there is an increasing focus on the possibility of developing *data-driven* dynamic models from "Big data". Two main types of models for dynamic systems are data-driven models (regression models, machine learning, empirical models) found by tuning parameters in some abstract mathematical structure so that the structure attains a best fit to the data, and mechanistic models (physics based models) which utilize balance laws based on the mechanicms that drive the change in the system coupled with semi-empiric phenomenologic models (transport laws, reaction rates, etc.). These two model types have their own strengths and weaknesses.

Data-driven models put little emphasis on understanding the system, can easily be automated in computers, but require large amounts of data and do not extrapolate well. Mechanistic models require good understanding of the physics of the system, impose limitations in model generalization, and are more complex to automate in computers. On the other hand, mechanistic models require fewer data, and extrapolate better. Ideally, one would combine the best of both model types. This would imply to use simplified mechanistic models prior to the availability of data, train regression models to describe uncertain phenomenological relations and missing dynamics in the mechanistic model, and this way gradually improve the model description.

Forward neural networks constitute a modern example of nonlinear regression models, i.e., a static mapping from an input signal/feature vector to an output signal which can be fitted to experimental data by modifying weight/bias parameters. Traditional use of neural networks for dynamic models are based on forward neural networks with autoregressive/delayed outputs and moving horizon/delayed inputs. Alternatively, recurrent neural networks include internal feedback paths in layers. Both approaches are based on discrete time data/models.

Realizing that forward neural networks essentially describe a nonlinear mapping, suppose now that both inputs and states of a system are available over a time horizon, together with the derivative of the states. Then one can treat the inputs and states as inputs to the neural network, and the derivatives of the states as outputs from the neural network. By tuning neural network parameters, this enables fitting the neural network to the vector field of the differential equation, leading to Neural Differential Equations (Farrell & Polycarpou 2006, Chen et al. 2018). If all states and state derivatives are known, fitting a neural network to the vector field is straightforward. States and state derivatives may be available in simulation studies for model reduction. However, when considering this idea for real systems with experimental data from a limited number of sensors, two problems are faced: (i) derivatives are not available, and can at best be found by some smoothing/spline fitting, and (ii) derivatives can not be found for all states — only for a reduced order transformation given by measurements. This makes the problem more challenging.

Based on the ideas in Chen et al. (2018), a set of packages for computer language Julia (Bezanson et al. 2017) are combined to fit neural differential equation models (Rackauckas et al. 2019, 2020) to experimental data, leading to differential equations that can be solved by standard differential equation solvers (Rackauckas & Nie 2017). However, the packages aim higher: they allow for a very general mixing of mechanistic models and neural differential equation models in the same framework, with possibilities for the user to choose whether only parameters in the neural network model are tuned, or parameters in both the mechanistic model and the neural differential equation. The advantage of this approach is significant: it allows for the desirable possibility of extending mechanistic models with data-driven elements.

In this paper, we consider a mechanistic model of a laboratory helicopter at University of South-Eastern Norway; the model is used in a course on Model Predictive Control. A mechanistic dynamic model of the system can be developed as in (Gäfvert 2001) using Lagrangian mechanics. Using sets of experimental data, a hybrid model consisting of the mechanistic model extended with neural network blocks is fitted using Julia package DiffEqFlux.jl, and the model is validated. This allows for developing an improved model compared to the purely mechanistic model, and serves as a starting point for challenges related to hybrid models.

The paper is organized as follows. In Section 2 presents the laboratory helicopter, and a mechanistic model developed from Lagrangian mechanics and nominal model parameters. The developed model is verified by simulation in Julia. In Section 3, experimental data are presented, and it is shown that poor knowledge of initial values for the model with the chosen nominal parameters gives low prediction accuracy. Next, a model fitting measure is proposed (loss function), and parameters and initial values are adjusted to minimize the loss function. The model is still imperfect. In

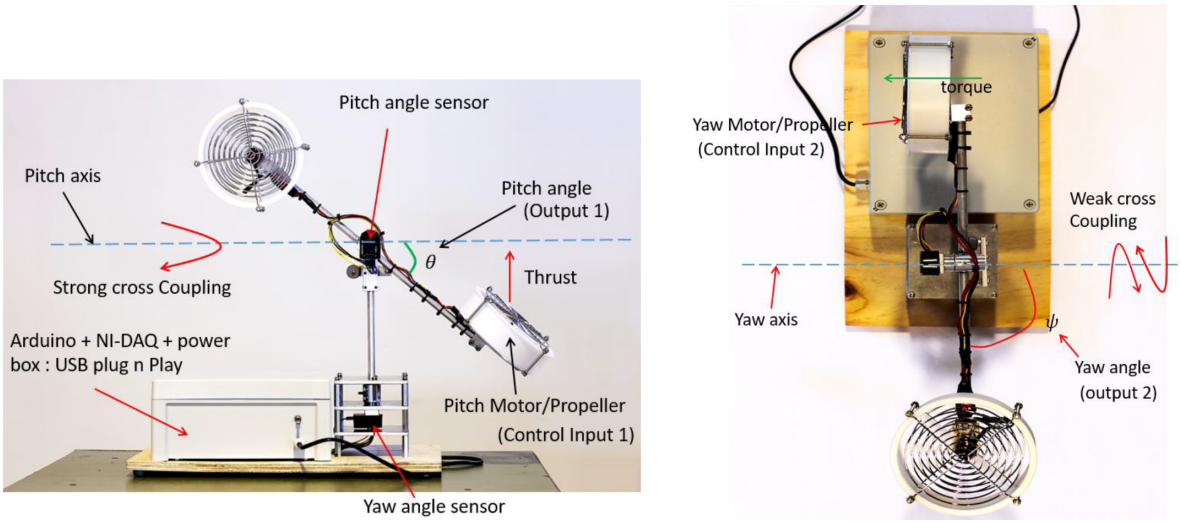

Figure 1: Laboratory helicopter, Sharma (2020).

Section 4, the torque model used in the mechanistic model is replaced by a neural networks block from the controller inputs (voltages) to the controller torques. Finally, in Section 5, some conslucions are drawns, and a number of challenge problems are proposed as future work. Details of the development of the mechanistic model are provided in Appendix A. Nominal model parameters and operating conditions are provided in Appendix B.

2 Helicopter mechanistic model

2.1 Laboratory helicopter

The laboratory helicopter used as a case study, is shown in Fig. 1.

Link https://web01.usn.no/~roshans/mpc/videos/Heli_deadband_effect.mp4 points to a video file detailing the operation of the helicopter, and some elements which makes it complicated to achieve a perfect mechanistic model of the helicopter.

2.2 Geometry of helicopter

Consider the helicopter in Fig. 2, with upward-pointing z axis. Origo of the body-fixed coordinate system is in the longitudinal inertial axis of the helicopter. The laboratory helicopter is hinged to the ground; in Fig. 2, the pivot point is located in the body-fixed origo.

If the helicopter body is elevated a distance h above the pivot point, differential mass $\mathrm{d}m\,(r)$ at position r along the longitudinal axis of the helicopter is given by coordinates z_r and ξ_r,

$$z_r = r\sin\theta + h\cos\theta$$
$$\xi_r = r\cos\theta - h\sin\theta.$$

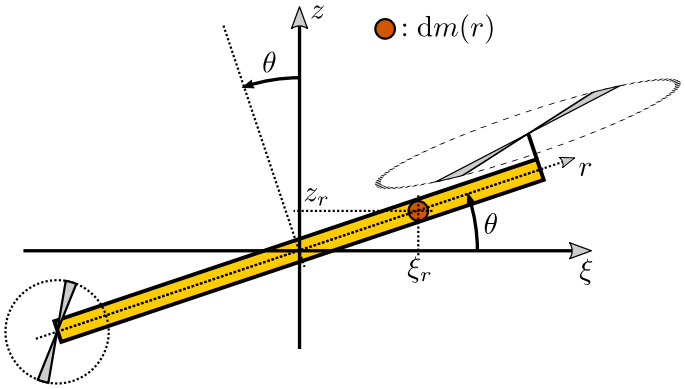

Figure 2: Helicopter in side profile, pitch angle θ.

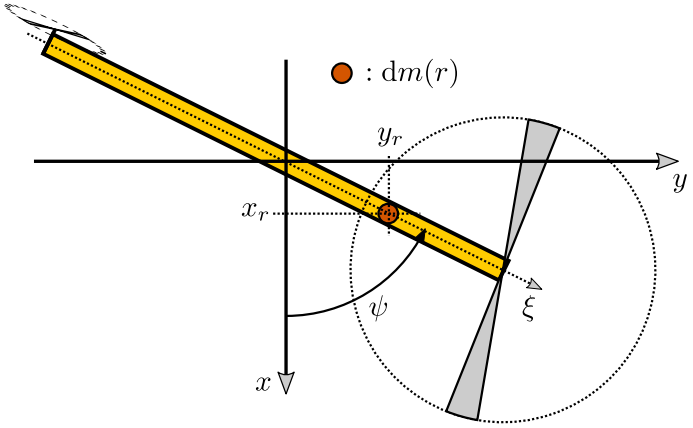

Figure 3: Helicopter in birds-eye view, yaw angle ψ.

The angle of nose raise θ is termed the *pitch* angle. Seen from a birds-eye, the helicopter can also rotate in the plane by the *yaw* angle ψ, Fig. 3.

The positions in the plane are

$$\begin{aligned} x_r &= \xi_r \cos\psi \\ y_r &= \xi_r \sin\psi. \end{aligned}$$

In summary, the position of mass $\mathrm{d}m\,(r)$ is given by Cartesian coordinates (x_r, y_r, z_r)

$$\begin{aligned} x_r &= (r\cos\theta - h\sin\theta)\cos\psi \\ y_r &= (r\cos\theta - h\sin\theta)\sin\psi \\ z_r &= r\sin\theta + h\cos\theta, \end{aligned}$$

where we consider generalized coordinates are $\mathsf{x} = (\theta, \psi)$.

2.3 Kinetic energy of helicopter

With linear velocities $v_x \triangleq \frac{\mathrm{d}x_r}{\mathrm{d}t}$, $v_y \triangleq \frac{\mathrm{d}y_r}{\mathrm{d}t}$, and $v_z = \frac{\mathrm{d}v_z}{\mathrm{d}t}$, and introducing angular velocities $\omega_\theta \triangleq \frac{\mathrm{d}\theta}{\mathrm{d}t}$ and $\omega_\psi \triangleq \frac{\mathrm{d}\psi}{\mathrm{d}t}$, the squared velocity $v_r^2 = v_x^2 + v_y^2 + v_z^2$ can be expressed by the generalized velocities $\mathsf{v} = (\omega_\theta, \omega_\psi)$ as

$$v_r^2 = r^2\left(\omega_\theta^2 + \cos^2\theta\cdot\omega_\psi^2\right) + h^2\left(\omega_\theta^2 + \sin^2\theta\cdot\omega_\psi^2\right) - 2rh\sin\theta\cos\theta\cdot\omega_\psi^2.$$

Combining the rotation of the helicopter body around the pivot point with moment of inertia J_{bc} and mass center distance r_{c}, and rotation around the shaft of moment of inertia J_{s} gies a total kinetic energy expressed as

$$K\left(\mathsf{x},\mathsf{v}\right)=\frac{1}{2}\mathsf{v}^{T}\mathsf{M}\left(\mathsf{x}\right)\mathsf{v}$$

where the mass matrix $\mathsf{M}\left(\mathsf{x}\right)$ is

$$\mathsf{M}\left(\mathsf{x}\right)=\begin{pmatrix} J_{\mathrm{bc}}+mr_{\mathrm{c}}^{2}+mh^{2} & 0\\ 0 & J_{\mathrm{bc}}\cos^{2}\theta+m\left(r_{\mathrm{c}}\cos\theta-h\sin\theta\right)^{2}+J_{\mathrm{s}}\end{pmatrix},$$

see Appendix A.1 for details.

2.4 Potential energy of helicopter

Setting the potential energy to zero in the pivot point, the potential energy of the helicopter becomes

$$\begin{aligned} P&=\int_{m}gz_{r}\mathrm{d}m\left(r\right)\\ &\Downarrow\\ P&=g\int_{m}\left(r\sin\theta+h\cos\theta\right)\mathrm{d}m\left(r\right),\end{aligned}$$

which can be expressed as

$$P\left(\mathsf{x}\right)=mg\left(r_{\mathrm{c}}\sin\theta+h\cos\theta\right).$$

2.5 Helicopter torques

We need the generalized force $\mathsf{F}=(T_{\theta},T_{\psi})$, which is given by some phenomenological relation

$$\mathsf{F}=\mathsf{F}\left(\mathrm{x},\mathsf{v},u\right).$$

Here, $u=(u_{\theta},u_{\psi})$ contains the voltages u_{θ} and u_{ψ} applied to the rotors. In mechanistic models, a proposal for F could be

$$\mathsf{F}=Ku-D\mathsf{v}$$

with a full motor gain matrix K (Sharma 2020)

$$K=\begin{pmatrix} K_{\theta,\theta} & -K_{\theta,\psi}\\ K_{\psi,\theta} & -K_{\psi,\psi}\end{pmatrix},$$

and a diagonal damping friction matrix D

$$D=\begin{pmatrix} D_{\theta} & 0\\ 0 & D_{\psi}\end{pmatrix}.$$

In practice, however, this linear description may be too simple: the system may exhibit deadband thus invalidating the term Ku or nonlinear friction such as stiction or quadratic friction in v thus invalidating the term $D\mathsf{v}$. One reason why deadband may exist is discussed in https://web01.usn.no/~roshans/mpc/videos/Heli_deadband_effect.mp4: power from u is transmitted to the helicopter via wires in the shaft, with twisting of these wires. The system may even exhibit backlash, and thus make it necessary to introduce extra states z with generalized force $\mathsf{F}(\mathsf{x}, \mathsf{v}, z, u)$ Lichtsinder & Gutman (2016, 2019).

2.6 DAE formulation of model

We now introduce the *Lagrangian* L defined as

$$\mathsf{L}(\mathsf{x}, \mathsf{v}) \triangleq K(\mathsf{x}, \mathsf{v}) - P(\mathsf{x}).$$

A DAE formulation of the Euler-Lagrange equation[1] can be posed as

$$\begin{aligned} \frac{\mathrm{d}\mathsf{x}}{\mathrm{d}t} &= \mathsf{v} \\ \frac{\mathrm{d}\mathsf{p}}{\mathrm{d}t} &= \frac{\partial \mathsf{L}}{\partial \mathsf{x}} + \mathsf{F} \\ \mathsf{p} &= \mathsf{M}(\mathsf{x})\,\mathsf{v}. \end{aligned}$$

Here, the momenta p are angular momenta, $\mathsf{p} = (\mathsf{p}_\theta, \mathsf{p}_\psi)$. Term $\frac{\partial \mathsf{L}}{\partial \mathsf{x}} = \left(\frac{\partial \mathsf{L}}{\partial \theta}, \frac{\partial \mathsf{L}}{\partial \psi}\right)$ is given in Appendix A.2.

2.7 ODE formulation of model

If we instead eliminate the momentum, we can rewrite the model as implicit ODEs,

$$\begin{aligned} \frac{\mathrm{d}\mathsf{x}}{\mathrm{d}t} &= \mathsf{v} \\ \mathsf{M}(\mathsf{x})\frac{\mathrm{d}\mathsf{v}}{\mathrm{d}t} &= -\frac{\partial \mathsf{p}}{\partial \mathsf{x}}\mathsf{v} + \frac{\partial \mathsf{L}}{\partial \mathsf{x}} + \mathsf{F}. \end{aligned}$$

In the ODE formulation, we need $\frac{\partial \mathsf{p}}{\partial \mathsf{x}}\mathsf{v}$ in addition to $\frac{\partial \mathsf{L}}{\partial \mathsf{x}}$; $\frac{\partial \mathsf{p}}{\partial \mathsf{x}}$ is given in A.2. The ODE form of the model can then be expressed as

$$\begin{aligned} \left(J_{\mathrm{bc}} + r_{\mathrm{c}}^2 m + h^2 m\right)\frac{\mathrm{d}\omega_\theta}{\mathrm{d}t} &= -\omega_\psi^2\left(\left(J_{\mathrm{bc}} + m\left(r_{\mathrm{c}}^2 - h^2\right)\right)\cos\theta\sin\theta - r_{\mathrm{c}}h\left(\cos^2\theta - \sin^2\theta\right)\right) \\ &\quad - mg\left(r_{\mathrm{c}}\cos\theta - h\sin\theta\right) + T_\theta \end{aligned}$$

$$\begin{aligned} \left(J_{\mathrm{bc}}\cos^2\theta + m\left(r_{\mathrm{c}}\cos\theta - h\sin\theta\right)^2 + J_{\mathrm{s}}\right)\frac{\mathrm{d}\omega_\psi}{\mathrm{d}t} &= 2\omega_\psi\omega_\theta\left(\left(J_{\mathrm{bc}} + m\left(r_{\mathrm{c}}^2 - h^2\right)\right)\cos\theta\sin\theta\right. \\ &\quad \left. + mr_{\mathrm{c}}h\left(\cos^2\theta - \sin^2\theta\right)\right) + T_\psi \end{aligned}$$

[1] The Euler-Lagrange equation is normally expressed as $\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial \mathsf{L}}{\partial \dot{\mathsf{x}}}\right) - \frac{\partial \mathsf{L}}{\partial \mathsf{x}} = \mathsf{F}$; here, Leibniz' derivative notation is used instead of Newton's notation, and we have introduced the generalized momentum $\mathsf{p} = \frac{\partial \mathsf{L}}{\partial\left(\frac{\mathrm{d}\mathsf{x}}{\mathrm{d}t}\right)}$.

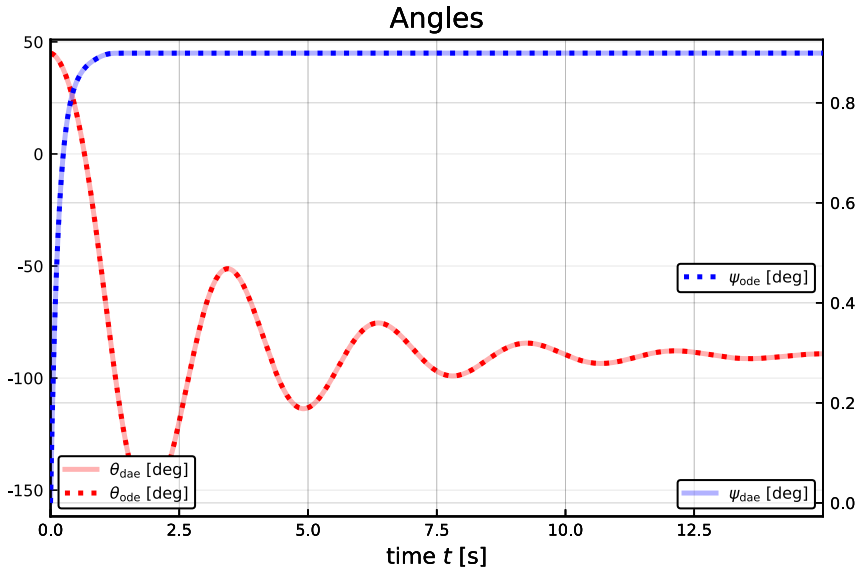

Figure 4: Pitch angle θ and yaw angle ψ for the operating conditions with DAE model and with ODE model.

with the trivial additional equations

$$\frac{\mathrm{d}\theta}{\mathrm{d}t} = \omega_\theta$$
$$\frac{\mathrm{d}\psi}{\mathrm{d}t} = \omega_\psi.$$

2.8 Model with pivot in body-fixed origo

If the pivot point is located in the helicopter origo, i.e., if $h \equiv 0$, the model becomes somewhat simpler; in the ODE form:

$$\left(J_{\mathrm{bc}} + mr_{\mathrm{c}}^2\right)\frac{\mathrm{d}\omega_\theta}{\mathrm{d}t} = -\omega_\psi^2\left(J_{\mathrm{bc}} + mr_{\mathrm{c}}^2\right)\cos\theta\sin\theta - mgr_{\mathrm{c}}\cos\theta + T_\theta$$
$$\left(\left(J_{\mathrm{bc}} + mr_{\mathrm{c}}^2\right)\cos^2\theta + J_{\mathrm{s}}\right)\frac{\mathrm{d}\omega_\psi}{\mathrm{d}t} = 2\omega_\psi\omega_\theta\left(J_{\mathrm{bc}} + mr_{\mathrm{c}}^2\right)\cos\theta\sin\theta + T_\psi.$$

2.9 Simulation of models

Nominal model parameters and operating conditions are chosen based on values in Sharma (2020), see Appendix B Tables 2 and 3, respectively. Figure 4 displays the pitch and yaw angles, θ and ψ, respectively, for the DAE formulation and for the ODE formulation. Figure 5 displays the pitch and yaw angular velocities, ω_θ and ω_ψ, respectively, for the DAE formulation and for the ODE formulation. Figure 6 displays the pitch and yaw (angular) momenta, p_θ and p_ψ, respectively, for the DAE formulation and for the ODE formulation. We see that the solutions are identical for the DAE and the ODE formulations, as they should be — since the DAE formulation and the ODE formulation are based on identical assumptions.

It is of interest to compare the solution of the models developed here, with that of Sharma (2020), see Section 2.8. Figure 7 displays the pitch and yaw angles, θ

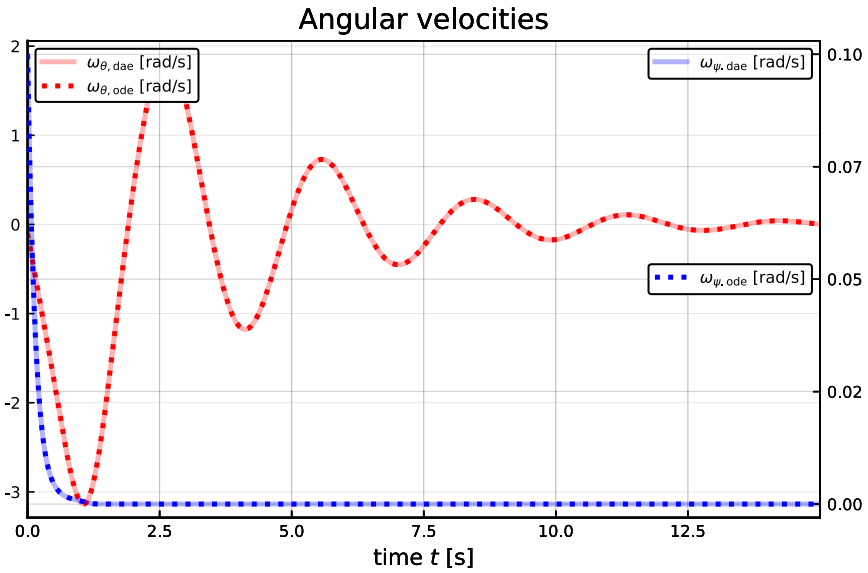

Figure 5: Pitch angular velocity ω_θ and yaw angular velocity ω_ψ for the operating conditions with DAE model and with ODE model.

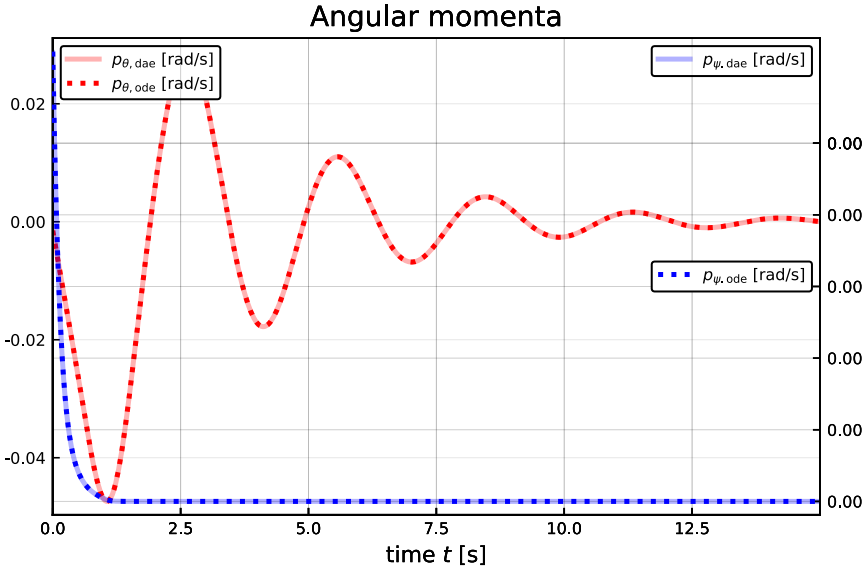

Figure 6: Pitch angular momentum p_θ and yaw angular momentum p_ψ for the operating conditions with DAE model and with ODE model.

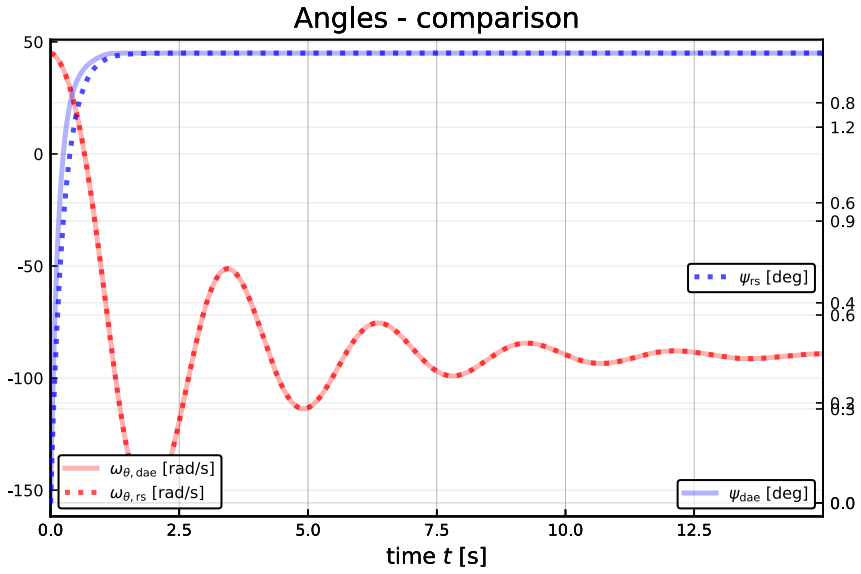

Figure 7: Pitch angle θ and yaw angle ψ for the operating conditions with DAE model and with RS model.

and ψ, respectively, for the DAE formulation developed here and for the RS model of Sharma (2020).

Figure 8 displays the pitch and yaw angular velocities, ω_θ and ω_ψ, respectively, for the DAE formulation developed here and for the RS model of Sharma (2020). The model of Sharma (2020) does not specify the angular momenta. As seen in Figs. 7 and 8, the true model and (DAE) and the approximate model (RS) are quite similar, although not identical in the case of the the yaw angle/yaw angular velocity. The similarity of the two models will depend somewhat on operating conditions.

3 Preliminary model fitting

3.1 Experimental data

Voltages for pitch motor u_θ and yaw motor u_ψ in experiments, Fig. 9.

Pitch angle θ and yaw angle ψ from experiments, Fig. 10.

Pitch angle θ and yaw angle ψ from experiments with nominal model parameters, Fig. 11.

3.2 Preliminary model fitting

We now tune model parameters p to minimize the loss function V

$$V(p) = \frac{1}{N_\mathrm{d}} \sum_{k=1}^{N_\mathrm{d}} \left(\theta_k - \theta_k^\mathrm{d}\right)^2 + \left(\psi_k - \psi_k^\mathrm{d}\right)^2 \tag{1}$$

where θ_k is the model pitch angle and ψ_k is the model yaw angle, while θ_k^d and ψ_k^d are logged data of the same angles. The model angles vary with the model parameters p; $\theta_k = \theta_k(p)$ and $\psi_k = \psi_k(p)$.

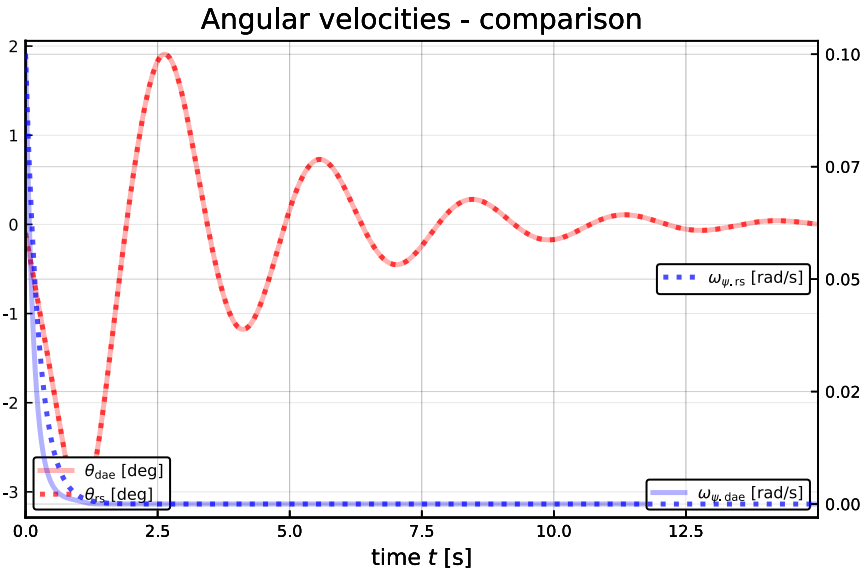

Figure 8: Pitch angular velocity ω_θ and yaw angular velocity ω_ψ for the operating conditions with DAE model and with RS model.

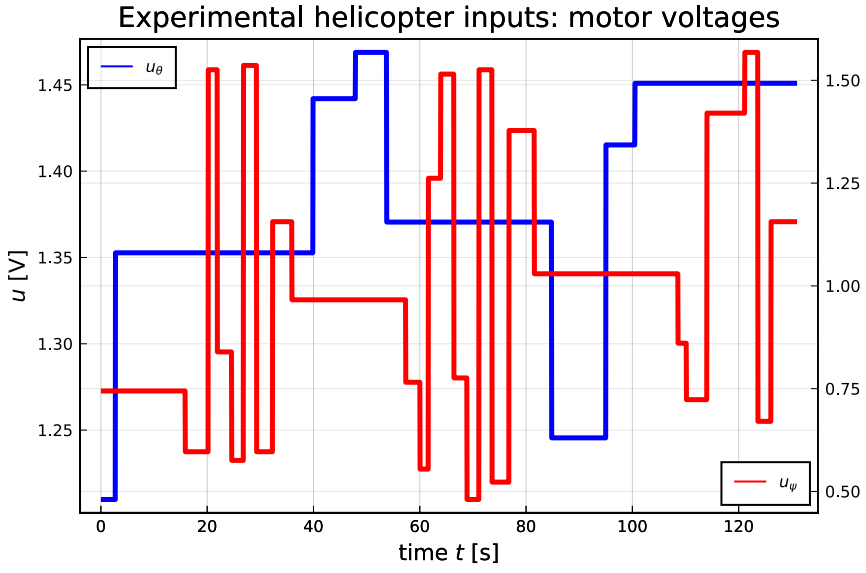

Figure 9: Pitch motor voltage u_θ and yaw motor u_ψ experiments.

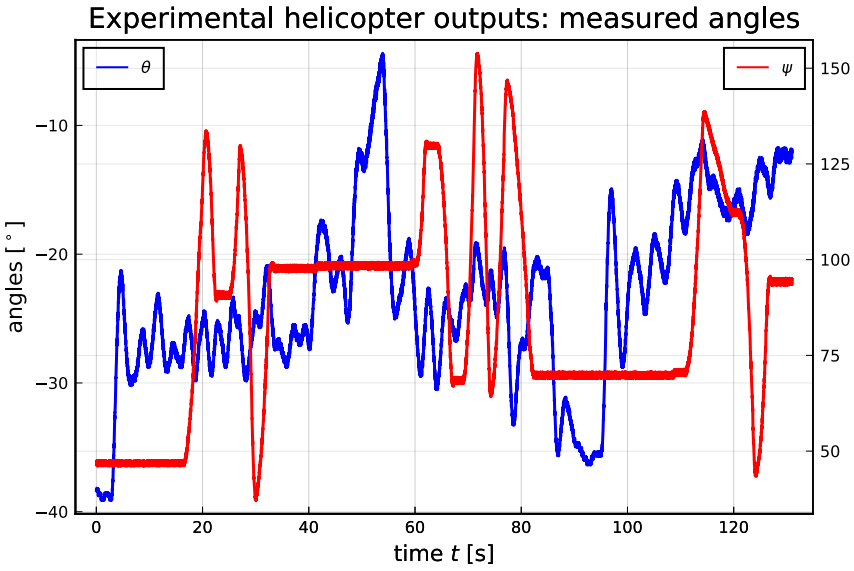

Figure 10: Pitch angle θ and yaw angle ψ from experimental conditions.

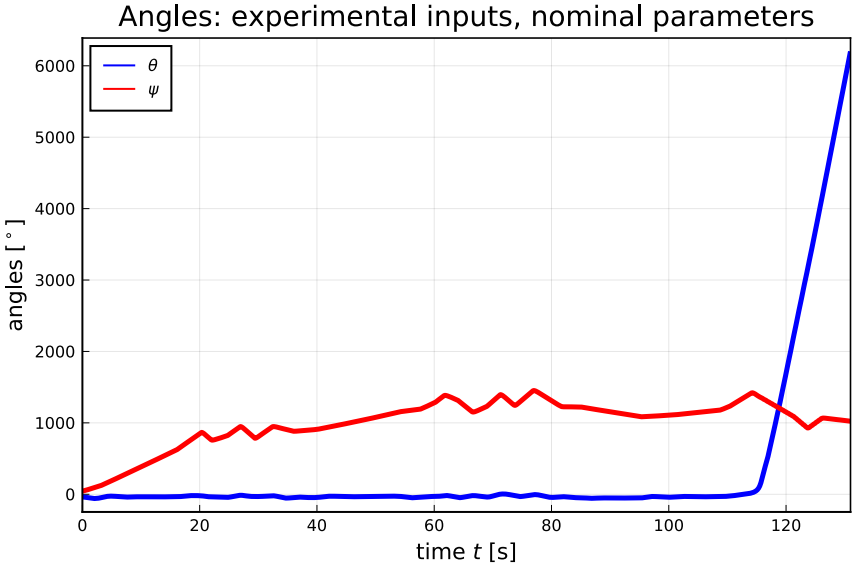

Figure 11: Pitch angle θ and yaw angle ψ from experimental conditions with nominal model parameters.

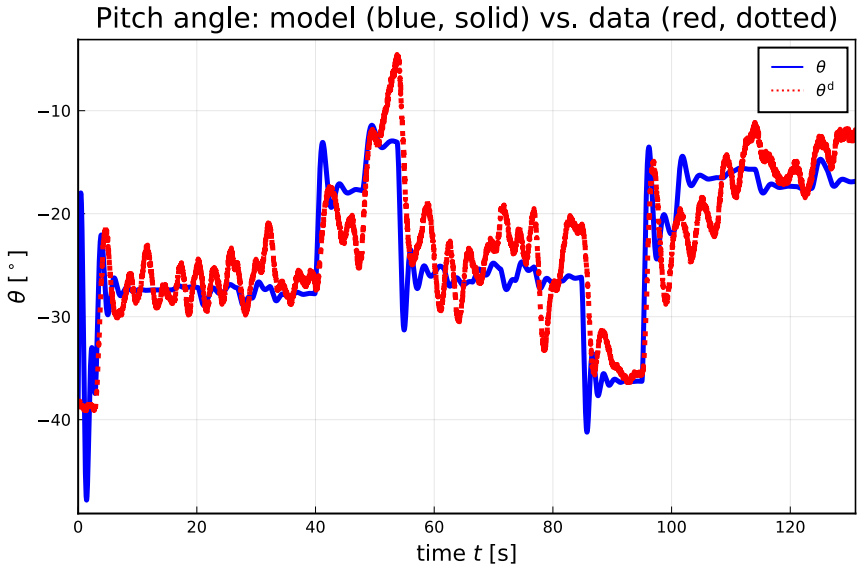

Figure 12: Pitch angle θ: comparing model angle with optimized parameters (blue, solid) and measured experimental angle.

Pitch angle θ from the model and from experimental data with optimized model parameters, Fig. 12.

Yaw angle ψ from model and from experimental data with optimized model parameters, Fig. 13.

Table 1 shows the fitted values of model parameters.

4 Hybrid model

4.1 Neural torque with input dependence

In Section 2.5, we considered a helicopter torque given as $\mathsf{F} = Ku - D\mathsf{v}$. We now modify this torque expression to

$$\mathsf{F} = Ku - D\mathsf{v} + \mathrm{FNN}\,(u; p) \tag{2}$$

where we keep the optimal parameter values in Table 1, but re-fit the model by tuning parameters p in $\mathrm{FNN}\,(\cdot)$ consisting of weights and biases in a two layer Feedforward Neural Net with 2 inputs, 16 nodes in the hidden layer and a $\tanh^{-1}(\cdot)$ activation function, and 2 outputs in the linear output layer. The tuning of the neural net is carried out in two phases: in the first phase, gradient search is done using an ADAM method of Julia package DiffEqFlux.jl, using the global loss function in Eq. 1 of Section 3.2. In a second, polishing step, we use a quasi-Newton method BFGS method to fine tune the parameters of the neural net.

After fitting the neural network, the pitch angles are as in Fig. 14 — which should be compared to Fig. 12.

The model fitting of the yaw angle is as in Fig. 15 — which should be compared to Fig. 13.

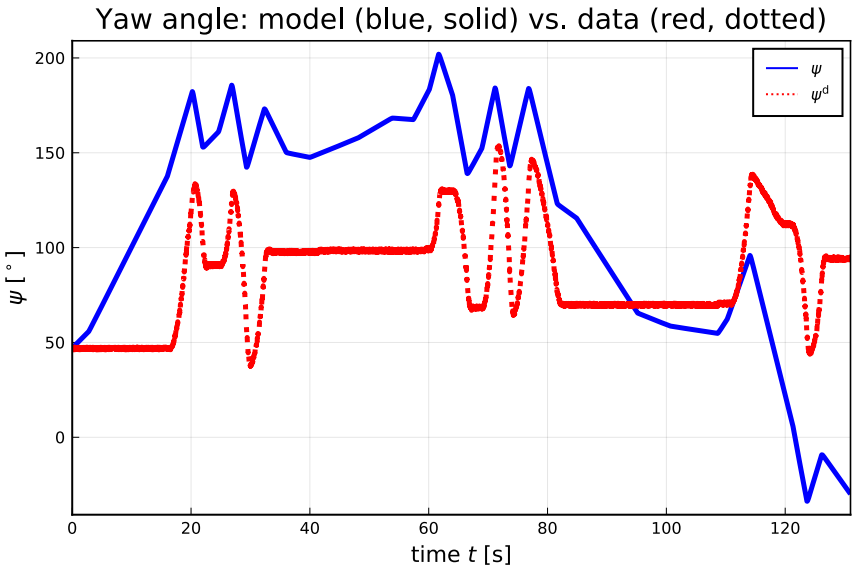

Figure 13: Yaw angle ψ: comparing model angle with optimized parameters (blue, solid) and measured experimental angle.

Table 1: Fittet parameters for laboratory helicopter.

Parameter	Nominal value	Fitted value
r_{c}	1.5 cm	2.2 cm
h	0 cm	0.1 cm
m	0.5 kg	0.72 kg
J_{bc}	0.015 kg m^2	0.01 kg m^2
J_{s}	0.005 kg m^2	0.0017 kg m^2
g	9.81 m/s^2	9.81 m/s^2
$K_{\theta,\theta}$	0.055 Nm/V	0.105 Nm/V
$K_{\theta,\psi}$	0.005 Nm/V	0.0017 Nm/V
$K_{\psi,\theta}$	0.15 Nm/V	0.03 Nm/V
$K_{\psi,\psi}$	0.20 Nm/V	0.043 Nm/V
D_θ	0.01 Nm/(rad/s)	0.014 Nm/ (rad/s)
D_ψ	0.08 Nm/(rad/s)	0.11 Nm/ (rad/s)

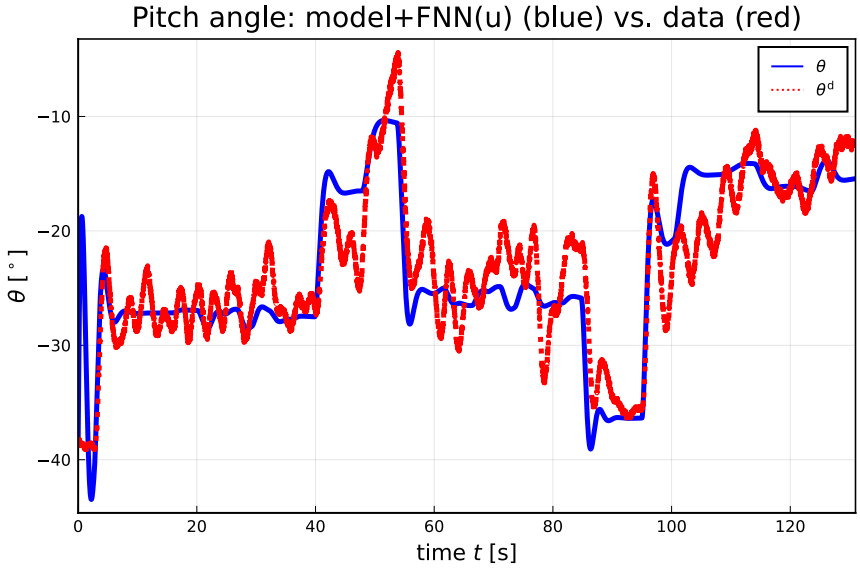

Figure 14: Pitch angle with torque modeled as in Eq. 2 with additive term FNN $(u; p)$.

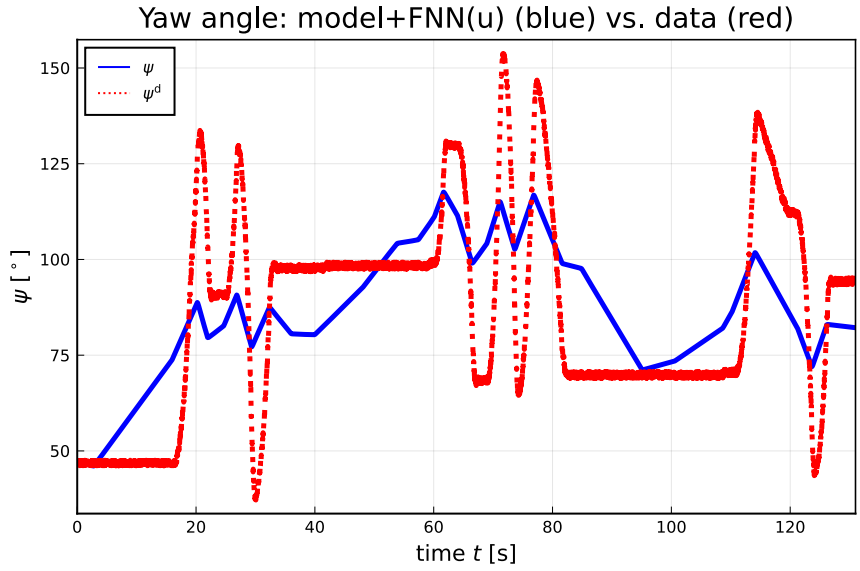

Figure 15: Yaw angle with torque modeled as in Eq. 2 with additive term FNN $(u; p)$..

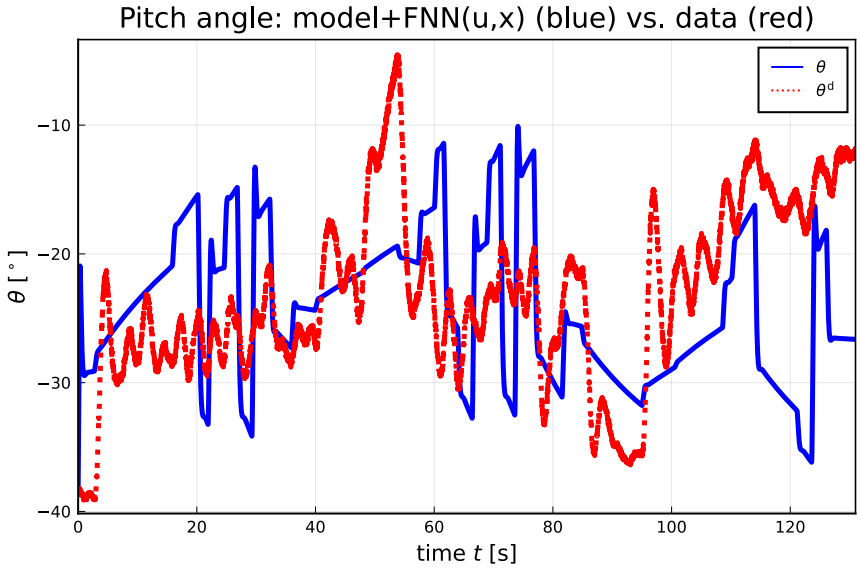

Figure 16: Pitch angle with FNN $(u, \mathsf{x}, \mathsf{v}; p)$.

4.2 Neural torque with input + state dependence

Next, we modify the torque expression to include a neural network which both depends on input voltage u and states x, v:

$$\mathsf{F} = Ku - D\mathsf{v} + \text{FNN}\,(u, \mathsf{x}, \mathsf{v}; p)\,. \tag{3}$$

Now we have a neural network with 6 inputs (u, x, v) and two outputs x, with a hiddenlayer with 16 nodes. We procede as in Section 4.1.

After fitting the neural network, the pitch angles are as in Fig. 16 — which should be compared to Figs. 12, 14.

The model fitting of the yaw angle is as in Fig. 17 — which should be compared to Figs. 13, 15.

4.3 Equation discovery

When finding the model augmented with a neural network addition to the torque F, we know the experimental input u and compute the resulting solution x. We can then post process the solution and compute FNN $(u; p)$ or FNN $(u, x; p)$. When considering the case of FNN $(u; p)$, this gives us the time series of u and FNN $(u; p)$, which we will use for "equation discovery". The idea is simply to propose a regression model between u and FNN $(u; p)$, and then fit this regression model. In such a regression model, we will postulate a large number of basis function which may have a physical origin. As an *example*, we could postulate a regression model of form

$$\text{FNN}\,(u; p) = c_1 + c_2 u_\theta + c_3 u_\psi + c_4 u_\theta^2 + c_5 u_\psi^2 + c_6 u_\theta u_\psi \cdots \tag{4}$$

In this case, we have a linear regression model that easily be fitted to the data set $(u, \text{FNN}\,(u; p))$. Some of the dataset can be used for training the model, and some of it can be used for validation. This way, it is possible to "discover" which of the basis

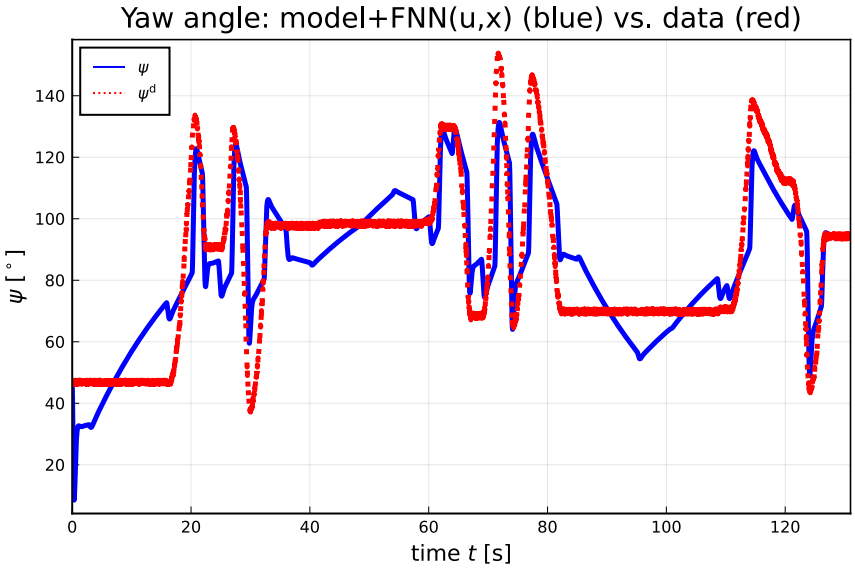

Figure 17: Yaw angle with FNN $(u, \mathsf{x}, \mathsf{v}; p)$.

functions/equations that are relevant to describe the fitted neural network part of the torque.

To automate this process, package `DataDrivenDiffEq.jl` is used. A set of basis functions are proposed in line with the regression equation in Eq. 4 using the `Basis()` constructor of package `DataDrivenDiffEq.jl`. Next, the optimizer `SInDy` (Sparse Identification of nonlinear Dynamics) of this package is used to "discover" the relevant equations that explain the neural network. Finally, this set of discovered equations can replace the neural network in the expression for F.

We find that

$$\begin{aligned}\mathrm{FNN}\,(u;p)_1 &\approx -4.37 \cdot 10^{-4} \cos(u_\psi) + 4.02 \cdot 10^{-4} \sin(u_\psi)\\ \mathrm{FNN}\,(u;p)_2 &\approx -1.35 \cdot 10^{-2} \cos(u_\psi) + 7.74 \cdot 10^{-3} u_\psi^2.\end{aligned}$$

By replacing the neural network with the approximate, "discovered" equations, the model fit is as in Figs. 18 and 19.

When comparing Figs. 14 and 15 to Figs. 18 and 19, observe that the pitch approximation is almost identical, while the yaw approximation is good — but a little bit off that from FNN $(u; p)$.

5 Conclusions and Future work

The key problem of fitting mechanistic models to experimental data has been considered, based on a simple model of a laboratory helicopter and experimental data from the lab rig. As in all engineering practice, the model structure of the original mechanistic model limits how well it is possible to fit the model to real data. Still, a mechanistic model gives important information about the system and may often be used for control design with some success, even when a perfect model fit is not

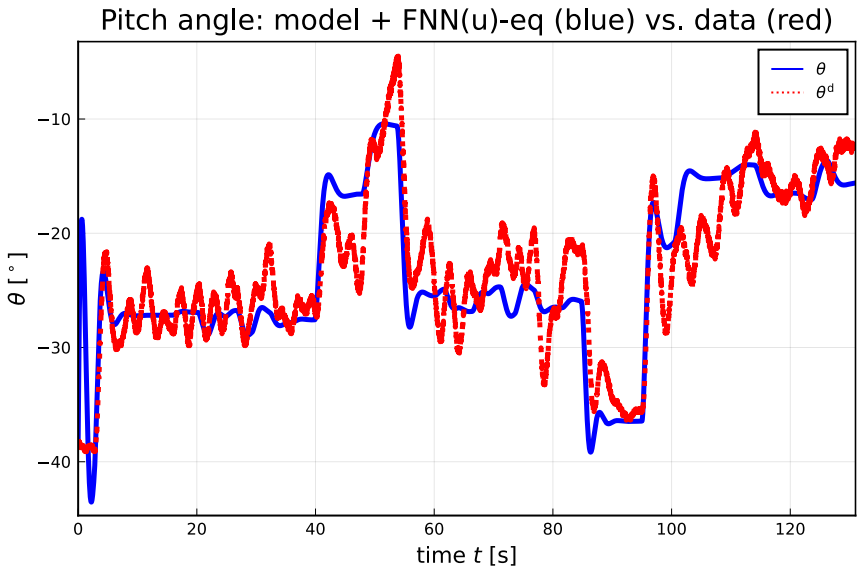

Figure 18: Pitch angle with regression approximation of FNN $(u; p)$.

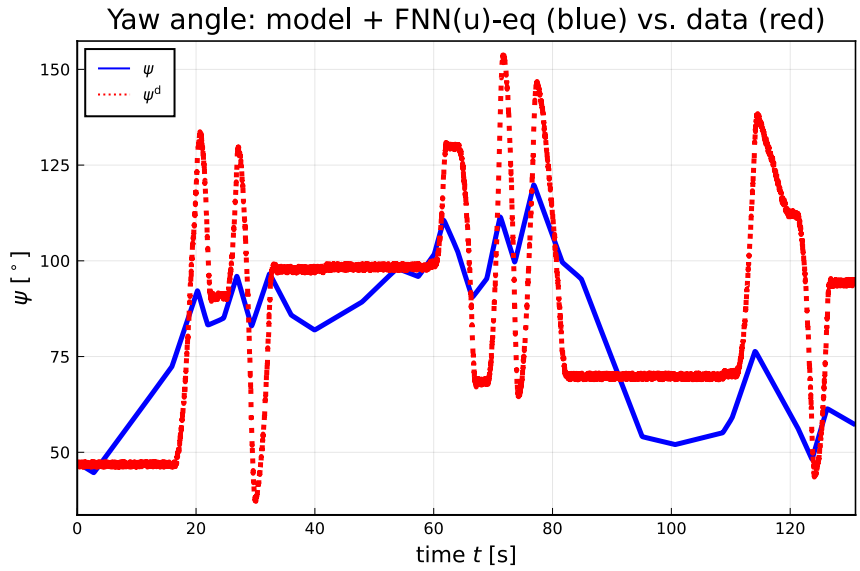

Figure 19: Yaw angle with regression approximation of FNN $(u; p)$.

possible. The reason why control design works for imperfect models is that feedback to some degree corrects for the effect of model error.

But it is of interest to *improve* on a mechanistic model as more data becomes available. Thus, we do not want to replace the mechanistic model by a completely data driven model, but instead augment the mechanistic model with data driven elements as more information becomes available. An obvious way to do this is to add regression type blocks in the mechanistic model, both in phenomenological laws and also in adding more states to the system with regression type vector fields. One possible data driven model structure is a neural network. The challenge with including regression blocks in dynamic models is how to integrate these in an efficient way so that differential equation solvers can handle them, so that it is possible to take advantage of GPUs, and how to make accessible model gradients for training the network within the dynamic model. In this paper, we have used packages within the DifferentialEquations.jl package and the Flux.jl package eco systems to achieve this. The original, fitted mechanistic model is good for pitch angle predictions, but poor for yaw angle predictions. The results show that yaw angle predictions improve considerable by adding a torque $\mathrm{FNN}\,(u;p)$ computed via a trained neural network. Furthermore, the results show that there is not much to gain from including states in the torque model, i.e., $\mathrm{FNN}\,(u,\mathrm{x},\mathrm{v};p)$ doesn't really improve on the yaw angle predictions.

For practical implementation, a neural network is not the most convenient model, as it is rather convoluted with many parameters. It is therefore of interest to see whether the neural network can be replaced by a regression approximation using more physically relevant basis functions. This is what the Equation discovery part of the Julia eco-system allows for. As seen, such equation discovery can lead to dramatically reduced model complexity compared to a neural network.

Overall, the tools offered in Julia give a first glimpse into what future modeling practice will look like: one starts by developing an efficient, and relatively simple mechanistic model for design/control synthesis/... As data become available, the model can be improved by adding unstructured data driven terms (e.g., neural networks) which subsequently are given a more physical interpretation by equation discovery.

The experience with model fitting of the helicopter model is a *starting point* for exploring these new, more *universal* (hybrid) models. The helicopter example also suggests a number of *challenges* that can be proposed as future work, in order to streamline the process of building such universal models, including:

1. Tools for (automatically) carrying out experiments and collecting data, including
   (a) The possibility to build GUIs with on-line graphics, buttons, sliders, etc. — as in https://web01.usn.no/∼roshans/mpc/videos/Heli_deadband_effect.mp4.
   (b) Experiment design for dynamic systems, e.g., optimize control inputs so that the condition number of sensitivity matrices are improved, etc.
2. For universal models:
   (a) Tools for analysing parameter identifiability for mechanistic models,

(b) Further improving tools and interfaces for model fitting of mechanistic models (choice of subset of parameters to tune, definition of loss function, optimization algorithms with parameter constraints, constraints on variables, etc.),

(c) Further improving tools and interfaces for adding data driven models within mechanistic models,

(d) Further improving tools for parameter uncertainty and prediction uncertainty description.

3. For the helicopter model:

(a) Improved model for the helicopter, including torque description (is there a tool for automating model development based on Lagrangian mechanics?) — the model described above is imperfect,

(b) Description of parameter uncertainty and prediciton uncertainty,

4. Documentation:

(a) Good presentations of work on the helicopter model will help popularize the existing tools,

(b) Documentation of problems with tools will help improve tools.

A Mechanistic model development

A.1 Kinetic energy

The kinetic energy of the helicopter *body* is

$$\begin{aligned} K_{\mathrm{b}} &= \frac{1}{2}\int_m v_r^2 \mathrm{d}m\left(r\right) \\ &\Downarrow \\ K_{\mathrm{b}} &= \frac{1}{2}\left(\omega_\theta^2 + \cos^2\theta\cdot\omega_\psi^2\right)\int_m r^2\mathrm{d}m\left(r\right) \\ &\quad + \frac{1}{2}h^2\left(\omega_\theta^2 + \sin^2\theta\cdot\omega_\psi^2\right)\int_m \mathrm{d}m\left(r\right) \\ &\quad - \frac{1}{2}\cdot 2h\sin\theta\cos\theta\cdot\omega_\psi^2\int_m r\mathrm{d}m\left(r\right). \end{aligned}$$

Let $r_{\mathrm{c}} \triangleq \frac{1}{m}\int_m r\mathrm{d}m\left(r\right)$ be the position of the mass center. From Steiner's Theorem, the Moment of Inertia of the helicopter *body* is $J_{\mathrm{b}} = \int_m r^2\mathrm{d}m\left(r\right) = J_{\mathrm{bc}} + mr_c^2$, where J_{bc} is the Moment of Inertia of the helicopter mass center. Then we can express K_{b} as

$$K_{\mathrm{b}} = \frac{1}{2}\omega_\theta^2\left(J_{\mathrm{bc}} + mr_{\mathrm{c}}^2 + mh^2\right) + \frac{1}{2}\omega_\psi^2\left(J_{\mathrm{bc}}\cos^2\theta + m\left(r_{\mathrm{c}}\cos\theta - h\sin\theta\right)^2\right).$$

In the laboratory setting, the helicopter is attached to the pivot point via a shaft. This shaft may also have a rotational kinetic energy K_s,

$$K_\mathrm{s} = \frac{1}{2} J_\mathrm{s} \omega_\psi^2.$$

The total kinetic energy is thus

$$K = K_\mathrm{b} + K_\mathrm{s}.$$

With the generalized coordinates $\mathsf{x} = (\theta, \psi)$ and corresponding generalized velocities $\mathsf{v} = (\omega_\theta, \omega_\psi)$, kinetic energy can be written as

$$K(\mathsf{x}, \mathsf{v}) = \frac{1}{2} \mathsf{v}^T \mathsf{M}(\mathsf{x}) \mathsf{v}$$

where the mass matrix $\mathsf{M}(\mathsf{x})$ is

$$\mathsf{M}(\mathsf{x}) = \begin{pmatrix} J_\mathrm{bc} + m r_\mathrm{c}^2 + m h^2 & 0 \\ 0 & J_\mathrm{bc} \cos^2\theta + m \left(r_\mathrm{c} \cos\theta - h \sin\theta\right)^2 + J_\mathrm{s} \end{pmatrix}.$$

A.2 Lagrangian and momenta gradients

In Section 2.6, we need the term $\frac{\partial \mathsf{L}}{\partial \mathsf{x}} = \left(\frac{\partial \mathsf{L}}{\partial \theta}, \frac{\partial \mathsf{L}}{\partial \psi}\right)$, which is given as

$$\begin{aligned} \frac{\partial \mathsf{L}}{\partial \theta} &= -\omega_\psi^2 \left(\left(J_\mathrm{bc} + m\left(r_\mathrm{c}^2 - h^2\right)\right) \cos\theta \sin\theta - r_\mathrm{c} h \left(\cos^2\theta - \sin^2\theta\right)\right) \\ &\quad - mg\left(r_\mathrm{c} \cos\theta - h \sin\theta\right) \\ \frac{\partial \mathsf{L}}{\partial \psi} &= 0. \end{aligned}$$

In Section 2.7, we need $\frac{\partial \mathsf{p}}{\partial \mathsf{x}}$ in addition to $\frac{\partial \mathsf{L}}{\partial \mathsf{x}}$:
Here, we have

$$\frac{\partial \mathsf{p}}{\partial \mathsf{x}} = \begin{pmatrix} \frac{\partial \mathsf{p}_\theta}{\partial \theta} & \frac{\partial \mathsf{p}_\theta}{\partial \psi} \\ \frac{\partial \mathsf{p}_\psi}{\partial \theta} & \frac{\partial \mathsf{p}_\psi}{\partial \psi} \end{pmatrix}$$

where

$$\begin{aligned} \mathsf{p}_\theta &= \left(J_\mathrm{bc} + m r_\mathrm{c}^2 + m h^2\right) \omega_\theta \\ \mathsf{p}_\psi &= \left(J_\mathrm{bc} \cos^2\theta + m \left(r_\mathrm{c} \cos\theta - h \sin\theta\right)^2 + J_\mathrm{s}\right) \omega_\psi. \end{aligned}$$

We find

$$\begin{aligned} \frac{\partial \mathsf{p}_\theta}{\partial \theta} &= 0 \\ \frac{\partial \mathsf{p}_\theta}{\partial \psi} &= 0 \\ \frac{\partial \mathsf{p}_\psi}{\partial \theta} &= -2\left(J_\mathrm{bc} \cos\theta \sin\theta + m\left(r_\mathrm{c} \cos\theta - h \sin\theta\right)\left(r_\mathrm{c} \sin\theta + h \cos\theta\right)\right) \omega_\psi \\ &= -2\left(\left(J_\mathrm{bc} + m\left(r_\mathrm{c}^2 - h^2\right)\right) \cos\theta \sin\theta + m r_\mathrm{c} h \left(\cos^2\theta - \sin^2\theta\right)\right) \omega_\psi \\ \frac{\partial \mathsf{p}_\psi}{\partial \psi} &= 0. \end{aligned}$$

Table 2: Parameters for laboratory helicopter.

Parameter	Value	Description
r_{c}	$1.5\,\mathrm{cm}$	Distance between pivot point and center of mass.
h	$0\,\mathrm{cm}$	Elevation of helicopter from pivot point.
m	$0.5\,\mathrm{kg}$	Mass of helicopter
J_{bc}	$0.015\,\mathrm{kg\,m^2}$	Moment of Inertia of body about center of mass.
J_{s}	$0.005\,\mathrm{kg\,m^2}$	Moment of Inertia of shaft.
g	$9.81\,\mathrm{m/s^2}$	Acceleration of gravity.
$K_{\theta,\theta}$	$0.055\,\mathrm{Nm/V}$	Torque constant on pitch coordinate from pitch motor.
$K_{\theta,\psi}$	$0.005\,\mathrm{Nm/V}$	Torque constant on pitch coordinate from yaw motor.
$K_{\psi,\theta}$	$0.15\,\mathrm{Nm/V}$	Torque constant on yaw coordinate from pitch motor.
$K_{\psi,\psi}$	$0.20\,\mathrm{Nm/V}$	Torque constant on yaw coordinate from yaw motor.
D_θ	$0.01\,\mathrm{Nm/(rad/s)}$	Friction torque damping coefficient in pitch coordinate.
D_ψ	$0.08\,\mathrm{Nm/(rad/s)}$	Friction torque damping coefficient in yaw coordinate.

Table 3: Initial operating conditions for laboratory helicopter.

Quantity	Value	Description
$\theta\,(0)$	$\frac{\pi}{4}\,\mathrm{rad}$	Initial pitch angle.
$\psi\,(0)$	$0\,\mathrm{rad}$	Initial yaw angle.
$\omega_\theta\,(0)$	$-0.1\,\mathrm{rad/s}$	Initial pitch angular velocity.
$\omega_\psi\,(0)$	$0.1\,\mathrm{rad/s}$	Initial yaw angular velocity.
$u_\theta\,(t)$	$0\,\mathrm{V}$	Pitch motor voltage.
$u_\psi\,(t)$	$0\,\mathrm{V}$	Yaw motor voltage.

B Nominal parameters and operating conditions

Nominal parameters for the mechanistic model are given in Table 2.

Nominal operating conditions for simulations are chosen as in Table 3.

References

Bezanson, J., Edelman, A., Karpinski, S. & Sha, V. B. (2017), 'Julia: A Fresh Approach to Numerical Computing', *SIAM Review* **49**(1), 65–98.

Chen, T. Q., Rubanova, Y., Bettencourt, J. & Duvenaud, D. (2018), 'Neural ordinary differential equations', *arXiv CoRR* **abs/1806.07366**. http://arxiv.org/abs/1806.07366.

Farrell, J. A. & Polycarpou, M. M. (2006), *Adaptive Approximation Based Control. Unifying Neural, Fuzzy and Traditional Adaptive Approximation Approaches*, Wiley-Interscience, Hoboken, New Jersey.

Gäfvert, G. (2001), Modelling of the eth helicopter laboratory process, Technical Report TFRT-7596, Department of Automatic Control, Lund Institute of Technology (LTH).

Lichtsinder, A. & Gutman, P.-O. (2016), 'Closed-form sinusoidal-input describing functionfor the exact backlash model', *IFAC-PapersOnLine* **49**(18), 422–427.

Lichtsinder, A. & Gutman, P.-O. (2019), 'On the dual properties of friction and-backlash in servo control systems', *IFAC-PapersOnLine* **52**(16), 340–345.

Rackauckas, C., Innes, M., Ma, Y., Bettencourt, J., White, L. & Dixit, V. (2019), 'Diffeqflux.jl - A julia library for neural differential equations', *arXiv CoRR* **abs/1902.02376**. http://arxiv.org/abs/1902.02376.

Rackauckas, C., Ma, Y., Martensen, J., Warner, C., Zubov, K., Supekar, R., Skinner, D. & Ramadhan, A. (2020), 'Universal differential equations for scientific machine learning', *arXiv CoRR* **abs/2001.04385**. https://arxiv.org/abs/2001.04385.

Rackauckas, C. & Nie, Q. (2017), 'DifferentialEquations.jl — A Performant and Feature-Rich Ecosystem for Solving Differential Equations in Julia', *Journal of Open Research Software* **5**(15).

Sharma, R. (2020), Two degrees of freedom (2-dof) helicopter, Laboratory problem in course iia 4117 model predictive control, University of South-Eastern Norway.