MACHINE LEARNING FOR DYNAMICAL MODELING OF A FLEXIBLE INVERTED PENDULUM SYSTEM.

M. A. DuPuis[1], N. A. Janmohamed[2], [1]NASA Kennedy Space Center, NE-L6, Cape Canaveral, FL, USA, [2]Santa Monica College, Santa Monica, CA, USA

Introduction: The inverted pendulum system, a canonical example of an unstable mechanical system, is often used to model the control problems encountered in the flight of rockets in the initial stages of launch, when the airspeed is too small for aerodynamic stability [1]. A system with a flexible pendulum is a variant that more accurately simulates rocket flight nonlinearities (particularly, the flex modes of the rocket).

To increase NASA capability of modeling dynamical systems for which closed form solutions are not clear or easily developed, this project aims to provide a machine learning approach that produces a learned dynamical model of the *Penny* robot (a rover with a flexible inverted pendulum) from operational data. The developed approach can then be generalized to other complex dynamical systems, including but not limited to rockets and other robotic systems.

Rationale: Traditional mathematical and control approaches can be used to develop system models, but in any real-world system, there are myriad nonlinearities that may be insufficiently described in the system model. If sufficient training data can be gathered from operation, a model could be learned that provides a representation of the system dynamics. This learned model can then be used to test different controllers, both in simulation and on the hardware.

Methods: The state-space representation of a linear time invariant discrete time dynamical system is given by $x(t+1) = \mathbf{A}x(t) + \mathbf{B}u(t)$. $\mathbf{A}$ is the state evolution matrix (how $x(t)$ influences $x(t+1)$), $x(t)$ is the state at time t, $\mathbf{B}$ is the control matrix (how $u(t)$ influences $x(t+1)$), and $u(t)$ is the control input at time t [2]. If $u(t)$ is zero, this equation simplifies to $x(t+1) = \mathbf{A}x(t)$. Data collected from letting a system initialized with random states and no control input evolve through time (until specified failure conditions) was used to train a neural network representation of the $\mathbf{A}$ matrix.

OpenAI's gym cartpole environment [3] was used as a testbed for developing a model to learn from collected operational data. This environment was modified to use system parameters representative of *Penny* (e.g. pendulum length, pendulum mass, cart mass, etc.). A model describing the 4-state system as specified by Barto et al. [4] was developed first, after which a model describing the linearized 6-state system as specified by DuPuis, Okasha [5] was developed.

Two datasets, *curr* and *next* were created that paired the measurement at time t with the measurement at $t+1$ (i.e. *curr[i]* $= x(t)$ and *next[i]* $= x(t+1)$). A multilayer perceptron architecture was used to learn a model of the system; given $x(t)$, the model predicts $x(t+1)$.

This approach will be applied to real operational *Penny* data to learn a model of the system dynamics.

Results: The developed pipeline produced a learned $\mathbf{A}$ matrix of the dynamics present in the simulation that evolves states through time with greater than 99.99% accuracy on unseen data. The simulation uses the semi-implicit Euler integrator, which may positively influence the model accuracy. At the time of this writing, not enough operational *Penny* data has been collected to test the ability of the pipeline to learn an accurate model of the *Penny* bot.

Conclusions: Though work remains to be done regarding developing a model for a physical system, the results are promising as it has been shown that a neural network architecture trained solely on operational data can learn a dynamical model that accurately represents the state evolution of a simulated nonlinear system.

References: [1] KH Lundberg, TW Barton. "History of inverted-pendulum systems." *IFAC Vol.* 42, 131-135, 2010. [2] KM Hangos et al. *Intelligent Control Systems an Introduction with Examples.* Kluwer, 254, 2004. [3] G Brockman, et al. OpenAI Gym. arxiv:1606.01540, 2016. [4] C Anderson, A Barto, R Sutto. "Neuronlike Adaptive Elements That Can Solve Difficult Learning Control Problem". IEEE, 1983. [5] DuPuis, M., Okasha, H. "Flex Dynamics Modeling for *Penny* Robot." Unpublished, 9-10, 2020.