DATS 598: Data Science Capstone

Journal Entries and Progress Documentation

A Physics-Informed Neural Network Approach to Parameter Estimation with User-Driven Exploration and Visualization

Eric Crisp
University of Pennsylvania
ecrisp@upenn.edu

Wednesday 10th September, 2025

Contents

Journal Entry - Week 2 - Main Proposal

Date: Wednesday 10th September, 2025
Duration: 10 hours

Objective

Synthesyze the literature review and understand a path forward with respect to the project on PINN, or pivot to alternative project (truth detection).

Activities Completed

- Whitepaper review of generating simulations from PINN-derived systems of equations deemed too sophisticated for a 12-week project.
- Simplification of the proposed PINN project includes removing the equation generation step and focusing on the subset problem space of Parameter Estimation for systems of ODEs.
- Systems of ODEs, or even PDEs if time allows (i.e., heat equation), can be converted from code, to symbolic representations with SymPy, converted to tensors for PyTorch the entire space of linear equations to be considered.
- ODEs with algebraic loops need to introduce stale parameters which can be captured symbolically and defined as the i - 1 value (i.e., lagged parameters).

Key Findings

While the full-blown PINN project that was intiated last week was determined to be too expensive and likely unfeasbile, the new task is a simplification and a narrowing of scope. The goal is to now take a system of equations and symbolically convert them to a matrix of coefficients of terms (linear, non-linear differential equations are out of scope), use this system to estimate the parameters via PINN with loss functions tied to fundamental physics and user-defined regularizations.

The use case for this kind of application is quite significant. When the Space Shuttle, or any manrated vehicle is flying, the sensor data stored is significantly less than the data stored on the test stand. So, when models are made and controllers are developed, the operation in the field relies on much less although critical flight data. Depending on the speed, reliability, and many other factors, a PINN-trained model in the loop could help define concepts such as:

- Remaining Product Life
- Likelihood of Catostrophic Failure
- Predictions of Unmeasured Data

For this project, we will start with a simple model and build to a more complex model that relies on my background in real-time modeling physical systems. The goal is to show how a trained model can predict, or connect the dots when information is missing, if the parameters estimated are sound. For example, imagine a web app with a diagram of a rocket engine. You have 15 sensors available to "include" in the training, but when flying, you must choose only 5. The predictions (i.e., likelihood of failure, or of the sensors not selected) can vary wildly depending on which 5 are selected, the loss functions included in training, and other factors. This can be visualized by simulation accuracy in the time domain, as well as an overall accuracy in the steady state domain.

Challenges Encountered

- Leverage aerospace background and create two systems of ODEs that can be used for testing parameter estimation:
  - trial and experimentation (fluid flow in a pipe with a chemical reaction)
  - full-scale implementation of application (i.e., rocket engine system)
- Create the generic systems of equations, convert them to tensors with SymPy, ensure algebraic loops are broken, and apply automatic differentiation.
- Create underlying pipeline from Python model to deployed (i.e., inference) on web app with parameter selection and vizuations.

Journal Entry - Week 2 - Secondary Proposal

Date: Wednesday 10th September, 2025
Duration: 10 hours

Objective

AI text generation can be incorrect in multiple ways. This project considers a simple, toy process for detecting whether the facts presented in a prompt (i.e., generated response) are likely true or not.

Activities Completed

- The FEVER dataset [1] contains a large number of samples scraped from Wikipedia that are considered to be "true" along with modifications to them that are considered "false".
- This process of truth detection would include significant semantic analysis, natrual language processing techniques, as well as data wrangling.
- Ideally, this would be turned into a web-app where the user could type in a statement, or question, and the result would be a likelihood that it is true.
- Extensions of this could be tied into responses from LLM to give the user a degree of confidence external to the text-generating tool itself.

Key Findings

The project timeline for this would be feasible as many of the tools are open source, the beginning dataset is well-known, and there is room to expand by including things like RAG and web scraping instead of relying on the information in FEVER.

Challenges Encountered

- NLP algorithms for understanding what is the fact that under consideration.
- Understanding what defines "trustworthy" relative to web data (i.e., what is "truth" and at what point when does X% "truth" becomes fake, or false).
- Data wrangling and pipeline process, more than just FEVER, incorporating a database consisting of cleaned, preprocessed web data.

[1] https://fever.ai/dataset/fever.html is a dataset commonly used for training truth detection.

Next Steps

1. Create clean-cut skeleton of the process, tooling, dataset, and libraries needed with timelines associated to implement neccesary applications, or extensions.

Reflections

As the scope of this project is being honed, the two viable paths that remain are an NLP, web scraping and data wrangling project and a project centered around deep neural networks that overlaps with differential equations. Both projects involve heavy, yet distinctly different, aspects of data science and machine learning. The main takeaway of the NLP project

Aspect	**PINN Parameter Estimation**	**NLP Truth Detection**
Model(s)	PINN	Physics-Informed NN
Training Data	Synthetic (simulation)	FEVER, web-scraping, RAG
Loss Function	Physics-based, MSE	MSE, Logistic Regression
Tooling and Overhead	Significant, excluding the additional web app interface	Minimal, except web app
Tech Stack	Python, FastAPI, PyTorch, Pandas	Python, JavaScript (Node, React)

Table 1: Comparison of PINN Parameter Estimation and NLP-based Truth Detection

A Systems of Equations for PINN Implementation

A. Simple Case: Mass Spring Damper System

Generically, this is a well understood system and includes the states and learned parameters.

$$\begin{aligned} x_1 &= x(t), \quad x_2 = \dot{x}(t) \\ \dot{x}_1 &= x_2 \\ \dot{x}_2 &= \frac{1}{m}\left(-cx_2 - kx_1 + F(t)\right) \end{aligned} \tag{1}$$

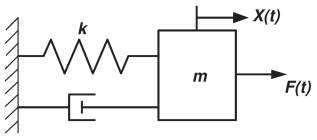

Figure 1: Simple Mass-Spring-Damper System

This sets the parameters to be estimated as

$$\theta = \{m, c, k\}$$

B. Complex Case: Liquid Rocket Engine Systeme Model

The RS-25, or Space Shuttle Main Engine, is a well known rocket engine that is relatively simple to simulate. A system of equations can be created to model the components. There are N parameters and M states where a parameter can be learned (i.e., line resistances, discharge coefficients, etc.) and a state is defined as a variable that dictates a state within the engine (i.e., chamber pressure, turbopump shaft speed, etc.). The system of equations is produced below and will be converted into matrix format after simulations show that numeric stability as the timeseries data here is used as samples for PINN training.

PCA and other dimensional reduction methods can, and should be, used to determine which sensors (states) are most critical to the sytem.

$$f_1(x, y) = 0 \tag{2}$$
$$f_2(x, y) = 0 \tag{3}$$

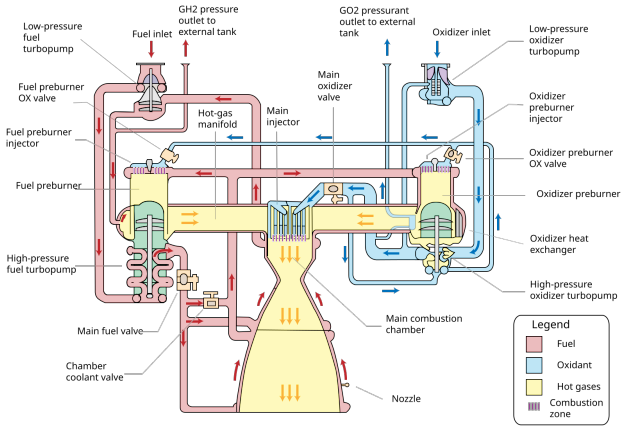

Figure 2: Space Shuttle Main Engine (RS-25) Engine Diagram