# C S 487/519 Applied Machine Learning I
# Project-Stage


**Group members:**

1. Amer Al-Radaideh (800641066) (in-class session)
2. Khandker Mushfiqul Islam (800701988) (online-class session)


## Problem Formulation:

In Aerospace research in general, scientists need to perform computer simulations before conducting any real experimental tests. However, it's difficult to obtain an exact dynamics model of the systems to represent the actual one.

In such research, researchers need to know the dynamics model between the PWM input commands to the drone's motors and the collective thrust force that is generated, see **Figure 1**. The dataset plotted and shown in Figure 2

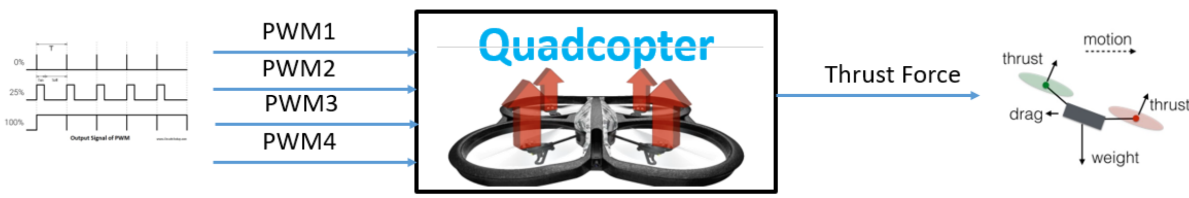


*Figure 1Input-output diagram*

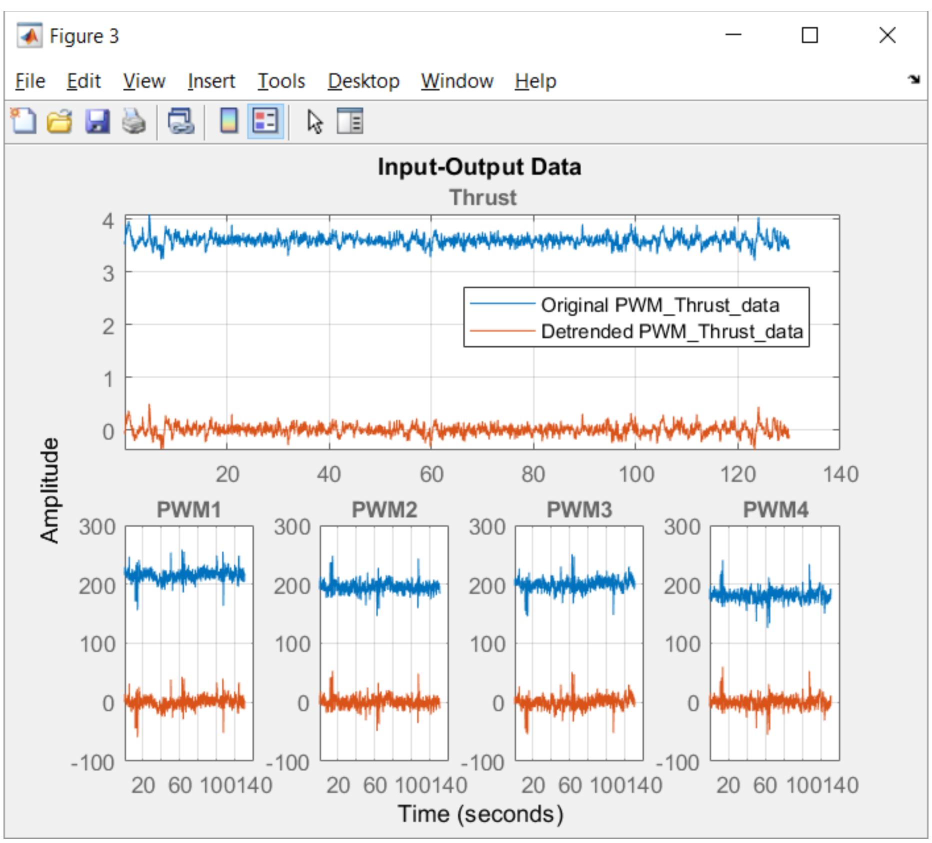


*Figure 2 Input-Output Data plots*

## Motivations:

The motivation behind this project is to be able to get the most possible accurate model that represents the thrust force due to the input PWM commands.

Thrust force estimation for rotary-wing vertical take-off and landing (VTOL) UAVs is challenging due to the difficulty to mount the load cell sensors and get the readings while its flying in the air.

Unlike traditional aerodynamic modeling solutions, in this project, we are looking forward to utilize one of the machine learning-based method which does not require the details of the aerodynamic information to model the Thrust-PWM relation.

The proposed method includes two stages: an off-line training stage and an on-line thrust estimation stage. Only flight data is used for the on-line estimation stage. We use Parrot AR.Drone as the testing quadrotor.

## Preliminary studies:

We have used the system Identification toolbox under Matlab environment to see how good are the results can be. The data were split into 25% for estimation and the remaining 75% for validation (see Figure 3).

Several models (available in the Matlab system Identification toolbox, see Figure 4) were selected to try them. As shown in Table 1, the state space gave the best results.

We are looking forward to try one of the Machine-Learning based techniques to find the best fit model and compare them with the models estimated using Matlab.

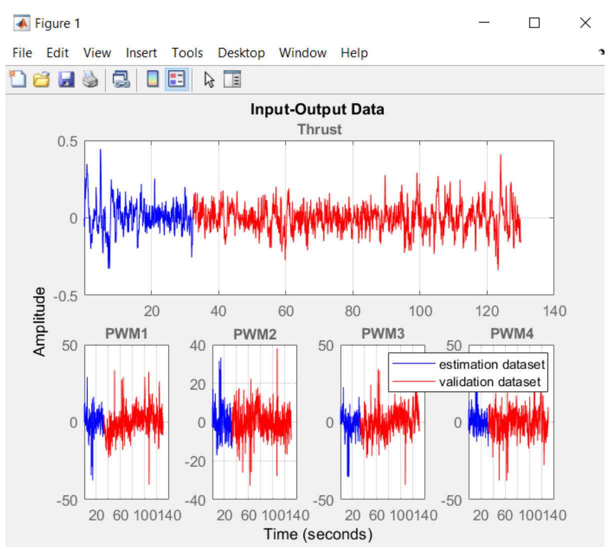


*Figure 3 Data splitting for Estimation and Validation*

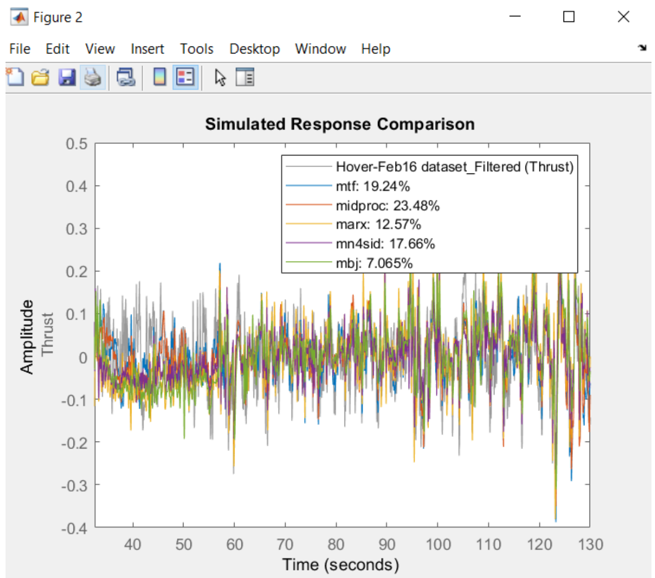


*Figure 4 Several Data models and their best fit percentage*

*Table 1 Models fitting with percentage fit*

| Model name | Best Fit % |
|---|---|
| **Transfer Function (mtf)** | 55.79% |
| **Process Model (midproc0)** | 51.74% |
| **Black-Box model-ARX Model (marx)** | 97.14% |
| **State-Space Models Using (mn4sid)** | 99.45% |
| **Box-Jenkins Model (bj)** | 95% |

## Proposed Solution:

The PWM-Thrust modeling problem is a highly nonlinear and has a mutually coupled terms. Since most of the methodologies that we have learned so far in the class are concerning the linear processes/functions we had to research the available tools to handle fitting multi-variable regression models to a data set with a highly nonlinear and mutually coupled terms as well as handling the noise presence issue. The noise (see Figure 5) makes it hard for the model to be bias-free and it also pushes the model towards overfitting because the model tries to make sense of the noisy data patterns and instead of discovering the real pattern, it fits itself to the noise.

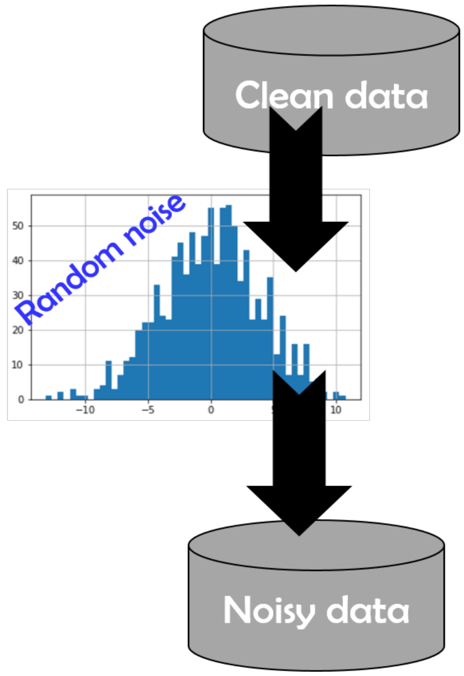


*Figure 5 Dataset with noise*

After a research on the Machine Learning methodologies that can be utilized to solve this problem, we came across few possible strategies for the nonlinear regressions:

1. Kernel Ridge Regression (KRR)
2. Support Vector Regression (SVR)
3. Random Forest Regression

For the Kernel ridge regression (KRR) (Murphy, 2012) combines Ridge regression and classification (linear least squares with l2-norm regularization) with the kernel trick. It thus learns a linear function in the space induced by the respective kernel and the data. For non-linear kernels, this corresponds to a non-linear function in the original space.

The form of the model learned by Kernel Ridge is identical to support vector regression (SVR). However, different loss functions are used: KRR uses squared error loss while support vector regression uses -insensitive loss, both combined with l2 regularization. In contrast to SVR, fitting Kernel Ridge can be done in closed-form and is typically faster for medium-sized datasets. On the other hand, the learned model is non-sparse and thus slower than SVR, which learns a sparse model for $\epsilon > 0$, at prediction-time.

For the last method to be used, a random forest (Raschka, 2015), to deal with the nonlinear relationships, it is an ensemble of multiple decision trees. It can be understood as the sum of piecewise linear functions in contrast to the global linear and polynomial regression models. In other words, via the decision tree algorithm, we are subdividing the input space into smaller regions that become more manageable.

With the research done, we found out that we can take advantage of Python machine learning library, scikit-learn to normalize the data, fit the model, keep the coefficients from becoming too large thereby maintaining bias-variance trade-off, and plot the regression score to judge the accuracy and robustness of each model.

## Dataset:

The following figures show the dataset plots of both the input (Figure 6 PWM dataset from the 4 motorsFigure 6) and the output (Figure 7). The dataset is uploaded to the github folder.

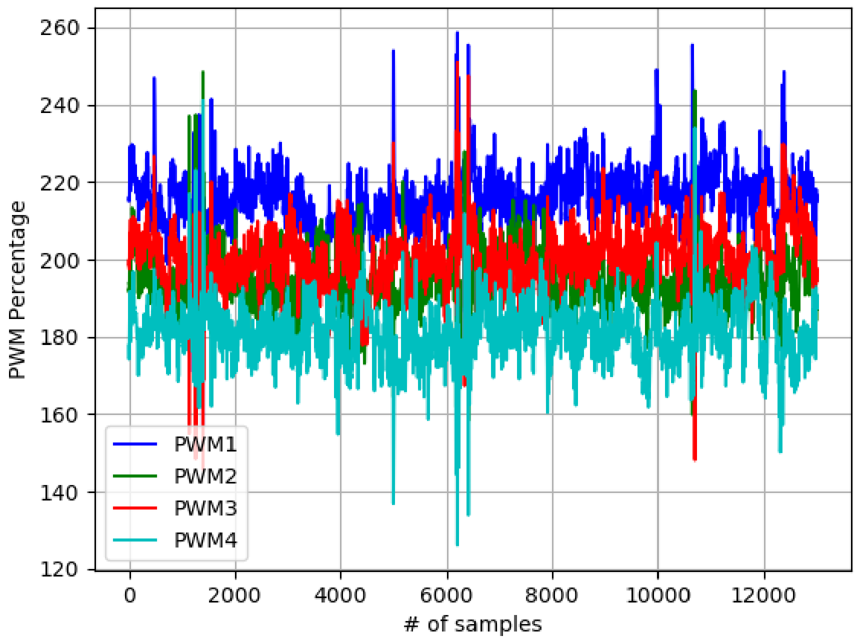


*Figure 6 PWM dataset from the 4 motors*

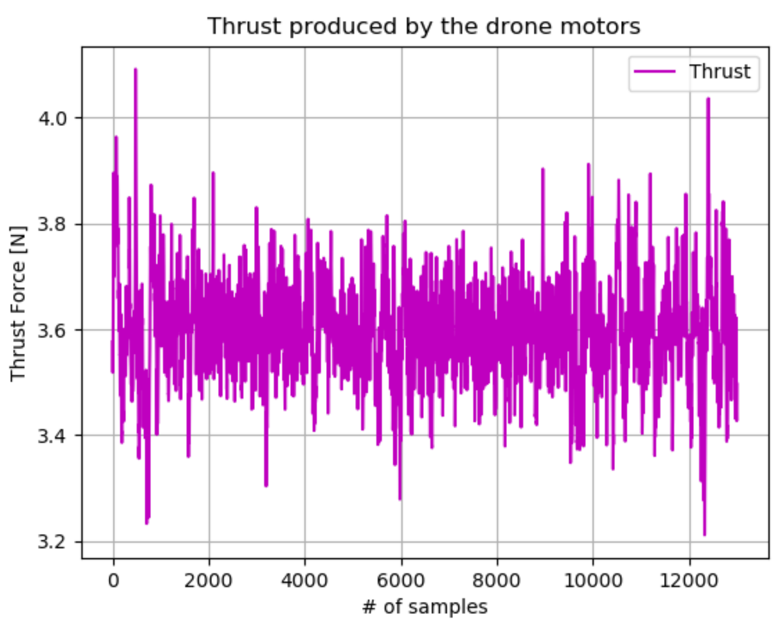


*Figure 7 Thrust dataset estimated from the accelerometer*

The following figures were generated using Matlab, just to make sure that we didn't mix any data.

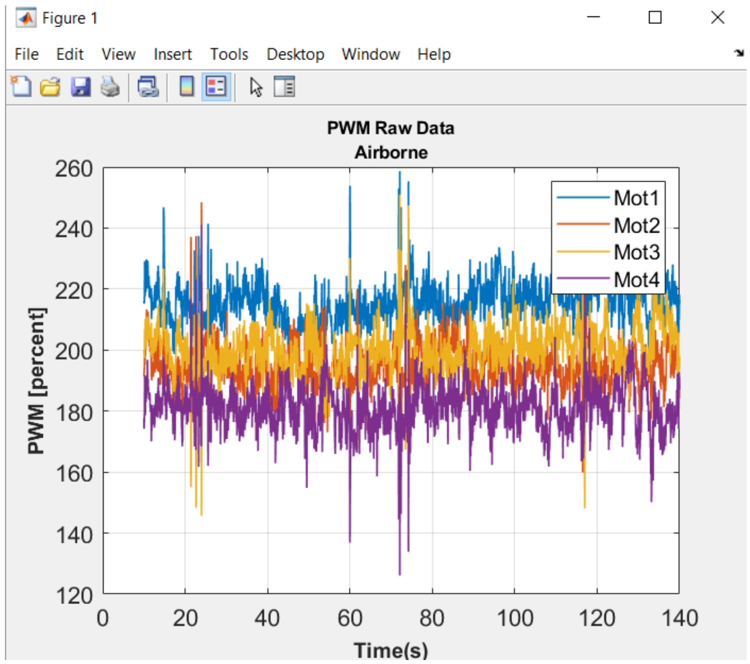


*Figure 8 PWM dataset from the 4 motors*

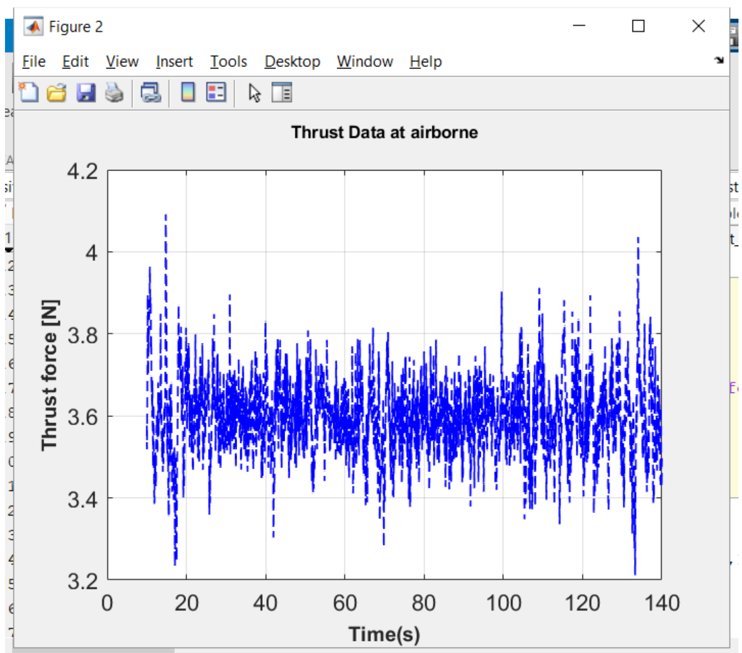


*Figure 9 Thrust dataset estimated from the accelerometer*

First, we will create a scatterplot matrix that allows us to visualize the pair-wise correlations between the different features in this dataset in one place. To plot the scatterplot matrix, we will use the pairplot function from the Seaborn library (http://stanford.edu/~mwaskom/software/seaborn/), which is a Python library for drawing statistical plots based on Matplotlib.

As we can see in the following figure, the scatterplot matrix provides us with a useful graphical summary of the relationships in a dataset:

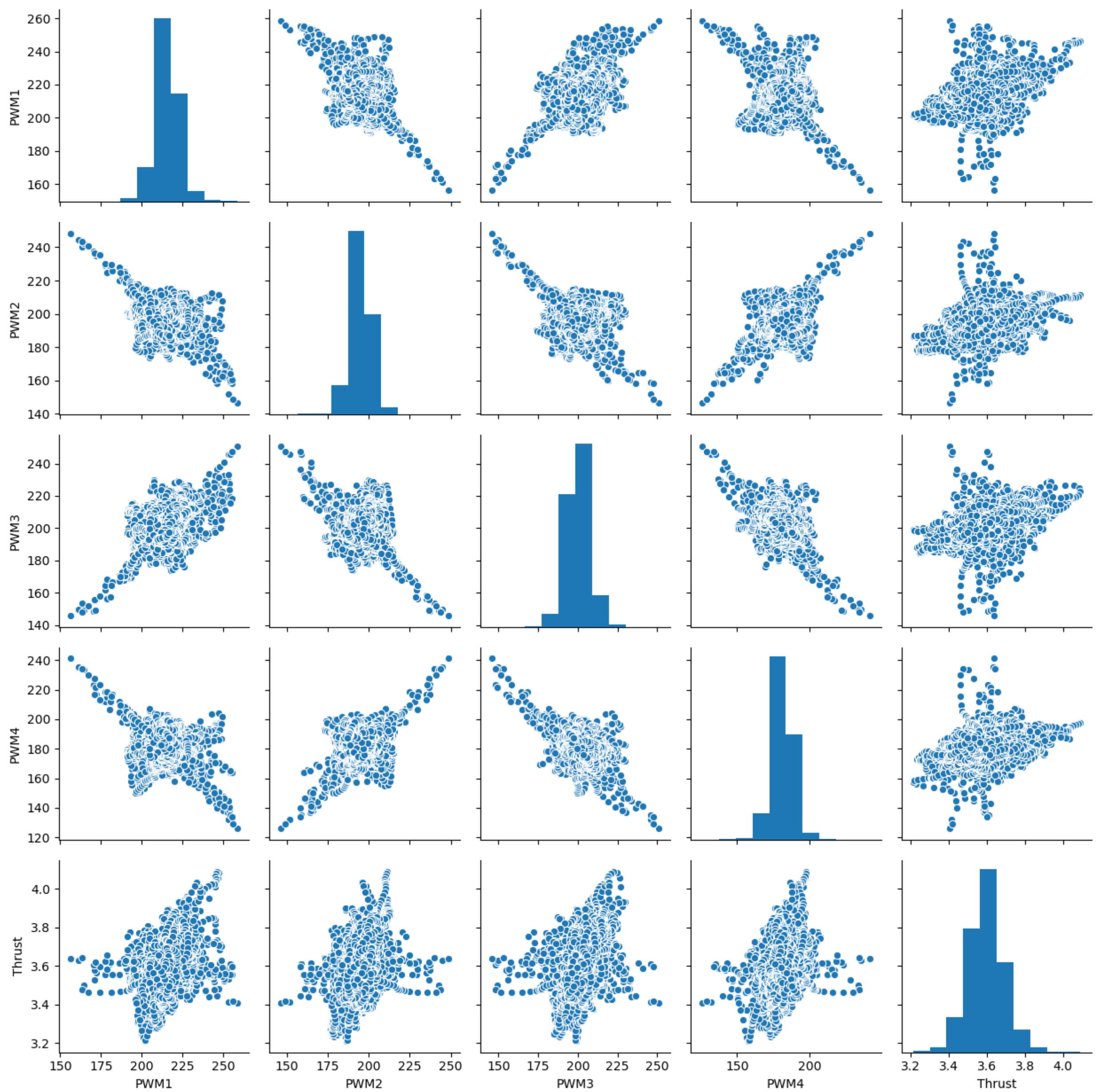

*Figure 10 Pairwise correlation between the features of the dataset*

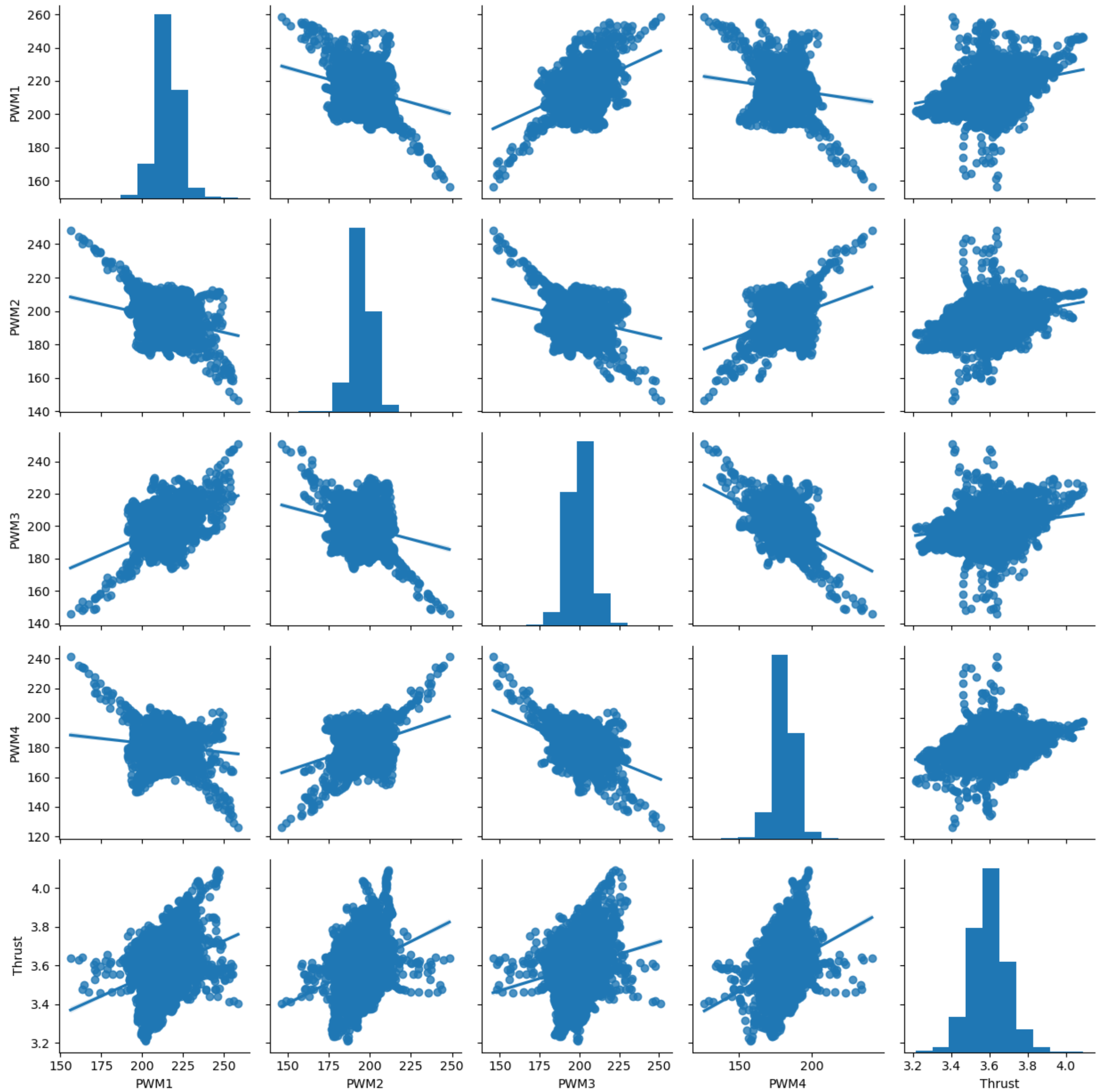


*Figure 11 Pairwise correlation plot with linear regression fitting*

Using this scatterplot matrix, we can now quickly eyeball how the data is distributed and whether it contains outliers. For example, we can see that there is a non-linear relationship between Thrust and all PWM signals (see the last row).
Furthermore, we can see in the histogram—the lower-right subplot in the scatter plot matrix—that the Thrust variable seems to be normally distributed but contains several outliers.

In the following figure, we have used the NumPy's corrcoef function on the five feature columns that we previously visualized in the scatterplot matrix, and we have used the Seaborn's heatmap function to plot the correlation matrix array as a heat map.

As we can see in the resulting figure, the correlation matrix provides us with another useful summary graphic that can help us to select features based on their respective correlations:

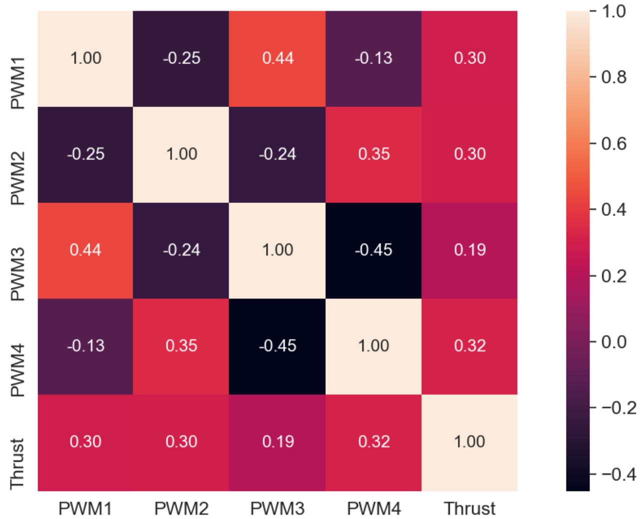


*Figure 12 correlation matrix array as a heat map*

Looking at the previous correlation matrix, we see that our target variable (Thrust) shows a low correlation with the PWM (input variables) and that's a clear nonlinear relationship between them. Based on the above we can introduce the concepts of non-linear regression models that we have proposed earlier.

## References

Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective.* The MIT Press.

Raschka, S. (2015). *Python machine learning.* Packt Publishing Ltd.