Memory Effect of Aerodynamic Force

Machine Learning (CS-433) Project 2

Tommaso Farnararo, Francesca Venturi, Girolamo Vurro
Department of Computer Science, EPFL, Switzerland

***Abstract*—In this work we study the problem of computing the aerodynamic force acting on a gas bearing supported rotor given the past trajectory of the geometric center of the rotor. This is usually achieved by numerical integration of the pressure distribution in the bearing, which is expensive to compute. The goal of this project is twofold: first, we aim to sidestep the PDE integration of the pressure with ML techniques; secondly, we investigate the memory effect, i.e. the dependence of the aerodynamic force on the past of the system. In particular, we successfully implement ANNs able to predict the force with a high degree of accuracy and measure the depth of the memory effect.**

I. Introduction

Gas lubricated bearings are fluid bearings where the fluid between two surfaces is a gas. These are classified depending on the bearing surface and the pressurization system [1]. This study is conducted on the Herringbone Grooved Journal Bearing (HGJB) lubricated with R-134a, a non-flammable refrigerant gas, which supports a rotor system. These machines are complex to operate due to the small available clearance between the rotor and the bushing (our model has a clearance of 5 μm). Their main application is high-speed rotating machinery, thanks to some properties such as temperature resilience and low friction [2]. The theory behind it, called NGT (Narrow Groove Theory), aims at computing the fluid film pressure distribution $\bar{P}(t)$ inside a HGJB, that follows the modified Reynolds equation:

$$\begin{gathered}\partial_\theta\left[\bar{P}\left(f_\theta \partial_\theta \bar{P}+f_c \partial_{\bar{z}} \bar{P}\right)\right]+\partial_{\bar{z}}\left[\bar{P}\left(f_c \partial_\theta \bar{P}+f_z \partial_{\bar{z}} \bar{P}\right)\right] \\ +c_s\left[\sin \beta \partial_\theta\left(\bar{P} f_s\right)-\cos \beta \partial_{\bar{z}}\left(\bar{P} f_s\right)\right]= \\ \Lambda \partial_\theta\left(\bar{P} f_v\right)+\sigma \partial_t\left(\bar{P} f_v\right)\end{gathered}$$

with suitable boundary and initial conditions. Once the pressure distribution is computed by solving the Reynolds equation with numerical methods, it is used to determine the bearing forces in the x and y direction:

$$\left[\begin{array}{c}\mathbf{F}_x(t) \\ \mathbf{F}_y(t)\end{array}\right]=2 \int_0^{L/D} \int_0^{2\pi}(\bar{P}-1)\left[\begin{array}{c}\cos \theta \\ -\sin \theta\end{array}\right] d\theta\, d\bar{z}$$

$\mathbf{F}_x(t)$ and $\mathbf{F}_y(t)$ are vectors, which contain directional forces for each bearing. To simplify the rotordynamics equations, the model only assumes cylindrical motions of the rotor, removing the tilting degrees of freedom. Consequently, the total aerodynamic force is twice the one generated by a single bearing.

II. Goal of the Project

The goal of the project is to study the memory effect of the aerodynamic force $\mathbf{F}(t)$ during its whole temporal evolution, i.e. the number of lagged instants d prior to t necessary to obtain a reliable prediction of the aerodynamic forces $F_x(t)$ and $F_y(t)$. The parameter d is called *delay parameter*. Currently, the integrals above are solved considering $\bar{P}(s)$ for $s \in [0, t]$, namely the whole evolution of the pressure distribution up to time t: this leads to enormous costs in terms of time and computational complexity. Clearly, the larger the amount of lagged observations, the higher the precision. However, the test error is expected to stabilize at low values once the minimum threshold for the delay parameter is exceeded, thus highlighting the well-known *elbow trend*.

III. Dataset Analysis

Our data are composed of $N = 500$ trajectories in the 2D cartesian system (each of them with a different starting point and characterized by $T = 2001$ time instants) and the corresponding forces. These trajectories refer to the position of geometrical center of the rotor as it turns inside the bearing, exposed to the aerodynamic force. Their values are stored in four different datasets: two of them contain the $\mathbf{X}$ and $\mathbf{Y}$ coordinates of the trajectory, which are the input, while the other two are the directional forces $\mathbf{F}_x$ and $\mathbf{F}_y$, the output. Hence, in total we are provided with four matrices $N \times T$, where each row represents the evolution in time of the corresponding quantity. Notice that every quantity is normalized over the characteristic dimensions of the system, hence the data are dimensionless.

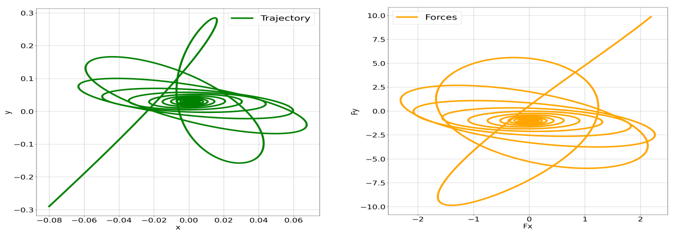

Fig. 1: 40^{th} trajectory and corresponding forces

One immediately observes that the rotor approaches a stationary equilibrium around the center of the bearing and the aerodynamic force eventually balances the vertical weight of the rotor, reaching the equilibrium at $(\bar{F}_x, \bar{F}_y)' = (0, -1)'$. Since the radius of the bearing is $R = 1$, we can create for each path a new variable $\mathbf{C}$ for the clearance parameter of the bearing, i.e. $\mathbf{C} = R - \sqrt{\mathbf{X}^2 + \mathbf{Y}^2}$. This quantity represents, at every time step, the distance between the trajectory and the edge of the bearing. This variable is introduced mainly because of the dynamics of the physical system in question: the smaller the clearance parameter, the larger the aerodynamic force.

IV. Models

For our purpose, we implement three different models: Linear Regression (LR), Feedforward Neural Network (FNN) and Convolutional Neural Network (CNN). LR is implemented to have a baseline for our research, then improving it with more sophisticated models, such as Neural Networks.

A. Linear Regression

The linear model aims at investigating in broad terms whether it is worth to search for a memory effect in the physical system. Indeed, it is expected that even a naif model, such as the linear one, will capture the dependence of the forces at time t on a limited number d of instants prior to it. Moreover, as far as the actual estimation of the forces is concerned, one expects the prediction to improve as the trajectory approaches the stationary point. Given a fixed delay parameter d, we implement two different linear models, which we present below.

1) Clearance Model: The first model takes the following form:

$$\tilde{F}_k^t = w_0 + \sum_{i=1}^{d} w_i C^{t-i} \qquad k = x, y \quad t = d+1, \dots, T \tag{1}$$

where the aerodynamic force F along k-direction at time t is predicted by means of the rotor positions C at d previous time steps, resulting in a design matrix of shape $N \times (d+1)$ and a vector of weights $\mathbf{w} \in \mathbb{R}^{d+1}$.

2) Auto-regressive Model: The second model adds some more information to the regression process. Indeed, not only it does encode the dependence of the aerodynamic force on the clearance parameter, but it also exploits the forces themselves as features: this is the reason why it is called *auto-regressive*. All these features are again analyzed for d time steps prior to the current t. Therefore, the model becomes:

$$\begin{gathered}\tilde{F}_k^t = w_0 + \sum_{i=1}^{d} (w_i C^{t-i} + w_{i+1} F_x^{t-i} + w_{i+2} F_y^{t-i}) \\ k = x, y \quad t = d+1, \dots, T\end{gathered} \tag{2}$$

where $\mathbf{w} \in \mathbb{R}^{3d+1}$ is the weights vector and all the other parameters follow the same aforementioned notation.

Except for the architecture of the feature matrix, these two models share several steps in the regression process:

- cross validation is applied, using 80% of the trajectories as training set and 20% of them as a testing set;
- as shown in formulas (1) and (2), the time evolution of the aerodynamic forces is estimated by means of $2000 - d$ models for $\tilde{\mathbf{F}}_x$ and $\tilde{\mathbf{F}}_y$, respectively, which are computed separately. This results in a process which is computationally heavy and time consuming;
- the cost function that we use is the Mean Square Error. Given a trajectory and fixed the delay parameter d, we then obtain two vectors of $2000 - d$ errors, corresponding to the models' losses for each time step;
- the predictions of F_x^t and F_y^t for $t = 1, \dots, d$ are not performed since the first d forces are already given by construction.

Due to the dimensionality of the problem, it emerges that it is possible to investigate the memory effect up to $d_1^* \sim 350$ lag observations in the first case and $d_2^* \sim 120$ in the second one, otherwise overfitting would happen. Although this constraint is quite restrictive for an exhaustive investigation, it still turns out to be sufficient to hit the target. Indeed, the two linear models aim at proving the existence of an optimal parameter d that reduces the loss while verifying the condition $d \ll T$. Therefore, memory effect is analyzed with a grid search across different values of d, each of them associated to the corresponding test error, which is obtained as the mean over the losses of all the tested trajectories. Plotting these results, we obtain the following curve:

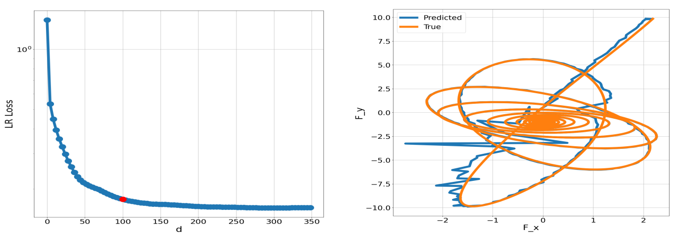

Fig. 2: Linear Regression Clearance Model - Semilogy Elbow trend of the MSE and 40th prediction

Fig. 2 shows that our intuitions about the error curve turn out to be correct: the error keeps decreasing as the lag parameter increases, but from approximately 100 lags on it gets flatter, looking similar to an *elbow curve*.

The error of the *Autoregressive Model* follows the same trend, being obviously way smaller than the first one, since the information added is extremely significant. However, the *autoregressive model* has been implemented having as features the forces $\mathbf{F}_x$ and $\mathbf{F}_y$, that are actually the output we want to predict, so they are not available in real-life simulations. It gets clear that the latter model, despite being performative, is not applicable: given a brand new dataset of trajectories ($\mathbf{X}$ and $\mathbf{Y}$), it would not be able to predict the output $\tilde{\mathbf{F}}_x$ and $\tilde{\mathbf{F}}_y$. In particular, to predict $\tilde{F}_x^t$ and $\tilde{F}_y^t$ for $t = 1, ..., T$ the model would require F_x^k and F_y^k for $k < t$, which are not available. Thus, this would lead to using $\tilde{F}_x^k$ and $\tilde{F}_y^k$, the forces predicted by the model itself during the previous instants. However, the error made in the the very first steps of the evolution would propagate, making the loss explode. On the other hand, while being qualitatively correct for the memory effect study, the *clearance* model is not sufficiently powerful for prediction tasks. This is why we decide to step into neural networks.

B. Feedforward Neural Network

1) Dataset creation: For the entire execution of the following studies, 80% of the trajectories are selected as training set, 10% as validation set, and the remaining 10% of observations are excluded a-priori and used as the testing set, to evaluate the performance of the network. Therefore, from the original dataset, we construct a new one as follows: fixed a delay parameter d, for each trajectory we build $T-d$ samples, each one relative to the step $t=d+1,...,T$. A single observation is then made of the features

$$X^{t-\tilde{d}} \quad Y^{t-\tilde{d}} \quad C^{t-\tilde{d}} \quad (and \quad potentially \quad F_x^{t-\tilde{d}} \quad F_y^{t-\tilde{d}})$$

for $\tilde{d} = 1,...,d$. As a consequence, a single input sample takes the form of $\mathbf{x}_i \in \mathbb{R}^{5d}$ for $i = 1,...,N_{tr+v}(T-d)$ (where $N_{tr+v} = 450$ trajectories) and is characterized by the information at d instants prior to the time step it refers to. The output is generated accordingly, so it is a vector $\mathbf{F} \in \mathbb{R}^{N_{tr+v}(T-d)\times 2}$.

2) Models description: Since we are also interested in the actual prediction of forces, it makes sense to implement 3 different models:

- *Coordinates FNN*: the first one considers as features the coordinates of the trajectories and the clearance parameter (3 features)
- *True FNN*: the second one, similarly to the linear case, adds the information of the true value of the aerodynamic forces (5 features)
- *Predicted FNN*: the third one exploits the predictions of the model itself at the previous instants (5 features)

Again, we recall that the second model is inapplicable at testing time: it is not used, but to have an ideal reference to compare performances to. Please notice that the last network represents exactly the model mentioned at the end of the previous paragraph.

Clearly, one would hope that the simplest model (i.e. *Coordinates FNN*) achieves the same accuracy as the ideal one (*True FNN*), presumably in slightly dilated times. As for the third model, it is the most difficult to implement, which is why it would be preferred only in case of significantly better performance than the first one.

3) Training process: The training process goes through several cascading steps.

As a preliminary inspection, several simulations with varying morphology of the FNN are run to set up the architecture, getting to optimality with three hidden layers, batches of 500 observations and 10 epochs. Throughout the following analysis, these parameters are kept constant. The activation function is *Relu* for each layer, the optimizer is *Adam's optimizer* and the loss function is the *Mean Square Error*. Finally, because of the large number of epochs, it is convenient to apply a *learning rate scheduler*, which we choose to be *CosineAnnealingLR*: this expedient encourages the learning rate to adapt over the epochs, sensibly improving the predictive performance.

Secondly, the parameters that are left to be chosen are the initial learning rate and the number of neurons per hidden layer. While keeping these values fixed, it is observed that *Coordinates FNN* actually achieves the same error as *True FNN*: in fact, by keeping the loss trend monitored, it is observed that the former simply takes a few more epochs to hit the same order of accuracy as the latter. The third model, instead, predicts the output considerably worse, again due to the propagation of the initial error. A more detailed presentation of numerical results will be held in the specific section.

Finally, once the *Coordinates* model is chosen, we proceed with a cross-validation over the parameters that are still to be set: initial learning rate and amount of neurons per hidden layer. The combination of parameters that minimizes the error is: `lr` = 0.01, `hidden1` = 256, `hidden2` = 256 and `hidden3` = 64.

To conclude, we apply a grid-search on the optimal delay parameter: again, the elbow-trend occurs. Notice that for the following study we consider $d = 100$, even if a much smaller delay parameter could be selected to have good results. This choice will be discussed in V.

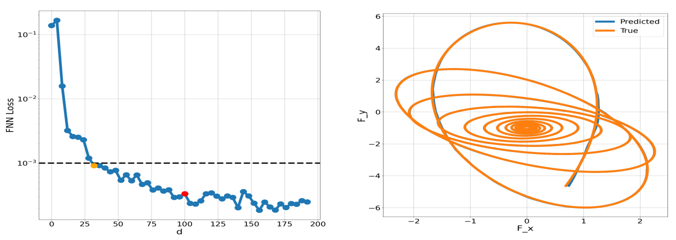

Fig. 3: Coordinates FNN - Semilogy Elbow trend of the MSE and 40th prediction

C. Convolutional Neural Network

1) Data creation: Although the results obtained so far are definitely satisfying, a model that approaches the problem with Convolutional Neural Networks (CNN) may be appropriate, hopefully leading to more accurate results. In particular, the application of a 1D convolution fits well the behavior of a times series. Moreover, even if the CNN reduces the amount of parameters to be learned, it also adds some more information. Due to its structure, a CNN requires a slightly different dataset. While in the FNN a single observation consists of a vector, now it is necessary to reshape the elements of that vector into 3 (or potentially 5, depending on the model) input channels, thus obtaining a tensor $\mathbf{x}_i \in \mathbb{R}^{3\times d}$. This means that each observation is represented as 3 (or 5) vectors: each one corresponds to an input channel and is scrolled by the convolutional kernel along its unique non-zero dimension (i.e., a 1D convolution).

2) Models description: As for the networks, they are similar to the FNN models:

- *Coordinates CNN* takes as input only the $\mathbf{X}$, $\mathbf{Y}$ and the clearance parameter $\mathbf{C}$ (3 features)
- *True CNN* also exploits the true forces (5 features)
- *Predicted CNN* uses the forces predicted by the model (3 features)

3) Training Process: CNN's training follows the same logical steps as FNN's. Once again, the *Coordinates* model ends up being the most appropriate. To be coherent with the whole study, we set the batch size of 500, 10 epochs and 3 convolutional layers. The same activation function (*ReLu*), optimizer (*Adam's*), learning rate scheduler (*CosineAnnealingLR*) and

loss function (*MSE*) are used. At the very end of the network, a linear fully connected layer is applied in order to fit the data to the output size. Furthermore, we add a skip connection between the input and the third layer to ensure that the identity function is learned by the network.
Again, the choice of optimal values for the initial learning rate, the kernel size and the feature maps of each convolutional layer are selected by cross-validation. This results in: `lr` = 0.01, `feature map1` = 64, `feature map2` = 32 and `feature map3` = 64.
Finally, in total analogy with the previous models, a grid-search on the optimal delay parameter highlight an *elbow trend*. Once again, for sake of coherence, we select $d = 100$, even if a much smaller one would be sufficient.

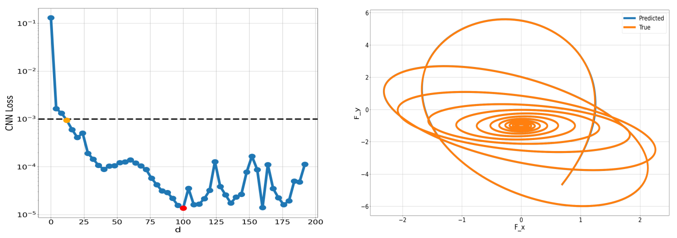

Fig. 4: Coordinates CNN - Elbow trend of the MSE and 40^{th} prediction

V. Results

In the previous section we announced which models would be qualitatively better and why we expect a well-defined hierarchy of performances. In this section we deal with a quantitative analysis of the results. Table I below summarizes the errors for the models:

Model	MSE	RE
Pred. FNN	2.71e−3 ± 1.56e−5	51.95% ± 1.58%
Pred. CNN	2.76e−3 ± 3.95e−5	53.09% ± 7.02%
Clearance LR	2.4e−2	35.32%
Coord. FNN	1.3e−5 ± 1.5e−6	2.02% ± 1.00%
Coord. CNN	**5.2e−6 ± 1.2e−6**	**1.29% ± 0.49%**
Autoregr. LR	3.2e−4	9.37%
True FNN	2.83e−6 ± 6.86e−7	0.89% ± 0.43%
True CNN	1.34e−6 ± 5.65e−7	0.81% ± 0.38%

TABLE I: Mean Squared Errors and Relative Errors (95% CI)

We also indicate the *Relative Errors* of the models: even if it is not really significant for our purpose, it is physically relevant. Coherently with the qualitative analysis, it is evident that the models *Coordinates* outperform their counterparts *Predicted*. The explanation is straightforward: in the *Predicted* models, we insert biased information, leading to a model which uses an input that is actually wrong. Similarly, we also see that the *Coordinates* networks achieve accuracy levels comparable to the ideal *True* model. Finally, we also observe that the Convolutional network is better than a more standard network such as Feedforward.
To compare the performance of the models we voluntarily choose to keep the delay parameter constant, i.e. $d = 100$. However, it is worth specifying that, in terms of the memory effect study, we choose a tolerance on the MSE ($tol = 10^{-3}$) and the minimum delay parameter d that fulfills that threshold. Fig. 3 and 4 show the parameters for which that tolerance is satisfied: $d_{FNN} = 32, d_{CNN} = 12$. As expected, the CNN Coord model minimizes the d parameter, again reinforcing the thesis that it is the best model.

VI. Robustness against different physical parameters

Once we ascertain that the final model (i.e., *Coordinates CNN*) achieves satisfactory results, the CSQI Lab proposes to test the robustness of that model on a set of observations generated under different conditions. While the original dataset comes from simulations having the same parameters of *pressure* at bearing ends and *rotor speed*, the one provided at this stage contains trajectories from 9 different combinations of these two parameters. In this regard, 2250 trajectories are provided, divided into 250 for each set of parameters. For computational reasons, we consider 70% of trajectories as training and validation sets and 30% as testing set. Despite the limited number of observations for each parameter set, the results obtained are considerably good: the model generalizes properly the behavior of aerodynamic forces, and errors are sufficiently low for each set. The *rotor speed* is in Krpm, while the *pressure* is in KPa.

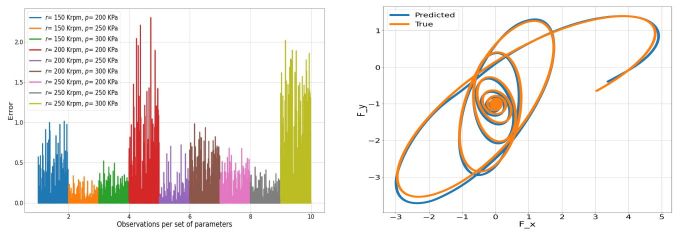

Fig. 5: MSEs for every parameters set and visual comparison of the predited and the true forces

VII. Conclusions and Future work

The project reveals that, independently of the model chosen, a delay parameter $d = 100$ is sufficient to obtain a satisfactory prediction of the aereodynamic forces, so the memory effect actually occurs. However, Neural Networks strongly outperform the linear regression: test errors are extremely low and, graphically, predictions and real forces almost overlap. Hence, the strength of Neural Networks is clear when prediction comes into play. Although the networks used are elementary and the computational tools limited, the results obtained are quite satisfactory: this is the reason why insights for future work are wide. In particular, we would enjoy being able to tackle this problem by means of Physics Informed Neural Networks (PINNs), which we are sure could make big improvements regarding the robustness of the model against the variation of physical parameters. Lastly, it would be challenging to analyse a possible correlation between the delay parameter and the intrinsic periodicity of the problem.

References

[1] L. Gu, E. Guenat, and J. Schiffmann, “A Review of Grooved Dynamic Gas Bearings,” *Applied Mechanics Reviews*, vol. 72, no. 1, 10 2019, 010802. [Online]. Available: https://doi.org/10.1115/1.4044191

[2] W. Liu, P. Bättig, P. H. Wagner, and J. Schiffmann, “Nonlinear study on a rigid rotor supported by herringbone grooved gas bearings: Theory and validation,” *Mechanical Systems and Signal Processing*, vol. 146, p. 106983, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0888327020303691