École Nationale Supérieure de l'Énergie, l'Eau et l'Environnement

MASTER Automatique et Systèmes Intelligents

Stage Assistant Ingénieur - 2021

Rapport de Stage

Development of simplified models of complex mechatronic systems in vehicles with hybrid modeling

Deutsches Zentrum für Luft- und Raumfahrt (DLR)

Münchener Strasse 20, 82234 Wessling, Deutschland

Internship Supervisors :

Caspar Bieri (Researcher Assistant at DLR)

Johannes Ultsch (Researcher Assistant at DLR)

School Tutor :

Olivier Sename

Student:

Victor Sanchez

Contents

1 Acknowledgments

I would like to thank Mr Caspar Bieri and Mr Johannes Ultsch, research assistants at the DLR, who were my supervisors during this internship. Even it was fully remote, they knew how to come with ambition, benevolence, seriousness and kindness.
Finally, I would like to thank Dr. Jonathan Brembeck who gave me the possibility to realize my internship in the laboratory and also Mr Olivier Sename, Professor at Grenoble INP - ENSE3, who created the contact between me and Mr Brembeck.

2 Abstract

2.1 English Version

Our everyday life remains on physics law. Isaac Newton tried to understand why everything was attracted by earth and created in 1687 the physical model about gravity that we all know. Even nowadays lots of the basics model are either empirical or based on physics approximation. But it appears that in certain case, these approximation are not sufficiently accurate. Based on data and artificial intelligence this paper explains how to create new accurate models, also called "hybrid models". The research involves the using of neural network to create these scientific machine learning models that are in parts physical and in parts based in data. The application was carried out around the mechanical system that makes the interaction between road and cars : Vehicle Suspension. Two different algorithm were studied and applied to Quarter Vehicle Model to illustrates this new hybrid model. The findings reveal that Neural Network could adapt themselves accurately to physical models such as Ordinary Differential Equation. Also it is shown that gradient descent methods as used with neural networks is an efficient way to determine the constant parameters of the vehicle itself. An extension has thus been studied with Stochastic Differential Equation or with a bayesian approach not only based on data but also on randomness and probability. On that final point, further research is necessary so as to establish the efficiency of the approach on other data and other model.

2.2 French Version

Notre vie quotidienne repose sur les lois de la physique. Isaac Newton a essayé de comprendre pourquoi tout était attiré par la terre et a créé en 1687 le modèle physique de la gravité que nous connaissons tous. Aujourd'hui encore, de nombreux modèles de base sont soit empiriques, soit basés sur des approximations physiques. Mais il apparaît que dans certains cas, ces approximations ne sont plus pertinentes. En se basant sur les données et l'intelligence artificielle, cet article explique comment créer de nouveaux modèles précis, également appelés "modèles hybrides". La recherche implique l'utilisation de réseaux neuronaux pour créer ces modèles scientifiques d'apprentissage automatique qui sont à moitié physiques et à moitié basés sur des données. L'application a été réalisée autour du système mécanique qui assure l'interaction entre la route et les voitures : la suspension du véhicule. Deux algorithmes différents ont été étudiés et appliqués au modèle de quart de véhicule pour illustrer ce nouveau modèle hybride. Les résultats révèlent que les réseaux neuronaux peuvent s'adapter parfaitement aux modèles physiques tels que les équations différentielles ordinaires. Il est également démontré que la descente de gradient utilisé avec des réseaux de neurones est un moyen efficace de déterminer les constantes du véhicule lui-même. Une extension a donc été étudiée avec des équations différentielles stochastiques et avec une approche bayésienne basée non seulement sur les données mais aussi sur un carctère aléatoire et les probabilités. Sur ce dernier point, des recherches supplémentaires sont nécessaires afin d'établir l'efficacité de l'approche sur d'autres données et d'autres modèles.

3 Introduction

Due to covid crisis, I had to realize the internship remotely and work with my own computer.
In this internship report I will first expose the laboratory and how I worked remotely. Then I will present the work I have done and the results I have obtained. Finally I will make a feedback on my contribution.

4 The laboratory

The DLR or Deutsches Zentrum für Luft- und Raumfahrt (1) is a German research laboratory founded in 1997 that resulted from the fusion of separate research organizations after the reunification of Germany.
Today, it has about 8000 employees in 33 institutes and establishments grouped in the 13 DLR centers in Germany. It also has offices in France and the United States.
The sectors of this laboratory are rather varied: Aeronautics, Aerospace, Transport, Energy, Security and Defense.
The laboratory I have worked at is based in Wessling near the city of Munich.

5 Internship's work management

5.1 Objectives

The first main objective was to make documentary research on the techniques of Hybrid Modelling and more globally Scientific Machine Learning (2) . Then, the goal was to apply numerically the techniques found to a specific mechanical system model which is the Quarter Vehicle Model (QVM).

5.2 Work Management

During the first weeks I learned how to program in Julia in Visual Studio Code and I made some researches on Hybrid Modelling and Scientific Machine Learning applications. Then I studied more in depth Neural Network in Julia and particularly two different training techniques. After that I produced algorithms that implement those techniques and I applied them to data given by my supervisor. Finally I made some research on other approaches of parameter estimation.

5.3 Organisation with working remotely

When I began my internship its been merely one year and a half that the covid crisis started. Thus the laboratory had developed lots of tools to work remotely and it was really easy for me.

First I had a computer in the lab which I had to connect to a software so as to realise my work. To count the hour of work, there was a platform where we could check in and check out. Also I had access to a github repository where I could exchange the program files with my supervisor. Finally, to communicate I had a special mail address and a skype business account to make video calls. In order to present the work I have done, there were two meetings per week. One on the Monday afternoon and one on the Thursday morning.

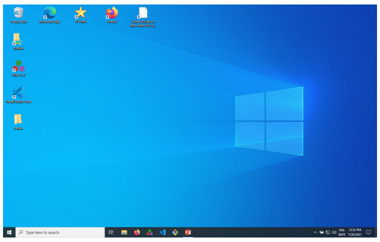

Figure 1: Desktop

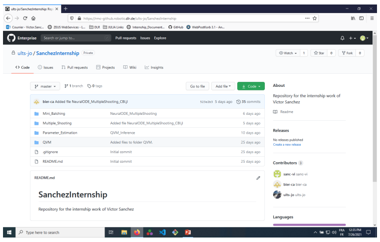

Figure 2: Github

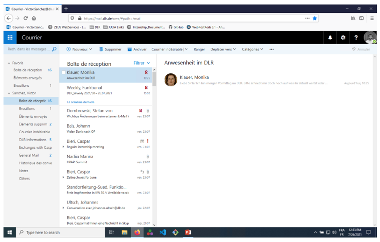

Figure 3: Mail

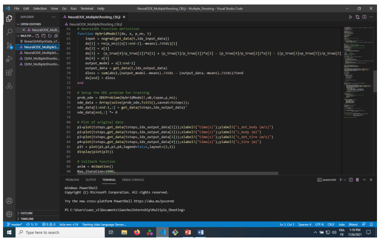

Figure 4: VScode

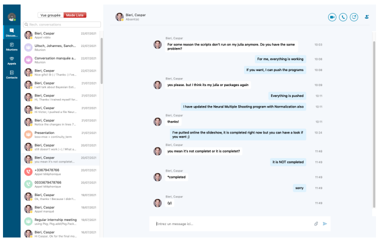

Figure 5: Skype

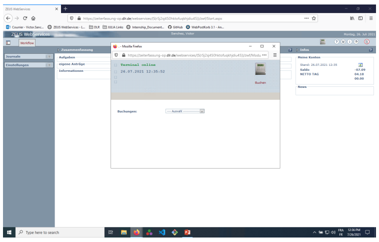

Figure 6: Remote Checking

6 Scientific Work

6.1 Quarter Vehicle Model

During the study we consider the mechanical system that creates the interaction between road and vehicle : The system of suspension that we approximate with a Quarter Vehicle Model (QVM) (3) like the following picture :

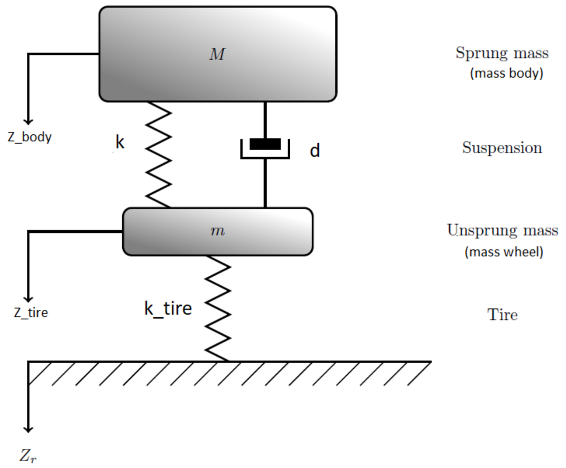

Figure 7: Quarter Vehicle Model

From this part we will have the following notation :

Z_{body}=z : position of the body ; Z_{tire}=z_t : position of the tire ; z_r : road profile

6.1.1 Physical Model

This mechanical system is ruled by the following equations :

$$\begin{cases} \ddot{z} = \frac{k}{M}(z - z_t) - \frac{c}{M}(\dot{z} - \dot{z}_t) \\ \ddot{z}_t = \frac{k}{m}(z - z_t) + \frac{c}{m}(\dot{z} - \dot{z}_t) - \frac{k_t}{m}(z_t - z_r) \end{cases} \tag{1}$$

From these equations we consider the state $Z = [\dot{z} \quad z \quad \dot{z}_t \quad z_t]^T$ and we obtain the following state space model :

$$\begin{bmatrix} \ddot{z} \\ \dot{z} \\ \ddot{z}_t \\ \dot{z}_t \end{bmatrix} = \begin{bmatrix} -\frac{c}{M} & -\frac{k}{M} & \frac{c}{M} & \frac{k}{M} \\ 1 & 0 & 0 & 0 \\ \frac{c}{m} & \frac{k}{m} & -\frac{c}{m} & \frac{-k-k_t}{m} \\ 0 & 0 & 1 & 0 \end{bmatrix} . \begin{bmatrix} \dot{z} \\ z \\ \dot{z}_t \\ z_t \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \frac{k_t}{m} \\ 0 \end{bmatrix} .z_r \tag{2}$$

6.1.2 Hybrid Model

For the hybrid model, we define the right-hand side of the first equation of (1) to be a neural network. Thus it will give us the following system of equation :

$$\begin{cases} \ddot{z} = F_{Neural\_Network}(Z) \\ \ddot{z}_t = \frac{k}{m}(z - z_t) + \frac{c}{m}(\dot{z} - \dot{z}_t) - \frac{k_t}{m}(z_t - z_r) \end{cases} \tag{3}$$

The Neural Network is used as a non linear function of the state which will allow the model to fit given training data by adjusting its parameters. To calculate the gradients necessary for training the neural network, the ODE has to be solved after each parameter adjustment. (4)

6.1.3 Loss Computation

Usually we can compute it in a discrete way with the following equation :

$$\sum ||data_Z - prediction_Z|| \tag{4}$$

6.2 Numerical working environment

In order to manipulate differential programming capabilities necessary for the gradient calculation for training. To do so the work has been realized in the high level language Julia 1.6.1 (5).This programming language is used in numerical analysis and computer science.

6.3 Training Techniques

For both model that we have seen before the general structure of the algorithm is merely the same at the beginning. The mathematical problem is defined with Ordinary Differential Equation and solved with Runge Kutta algorithm (order five) (6).

Here we will expose two training techniques that have been studied more in depth. (7). We are going to develop then two applications for each training techniques. The first one is to estimate the parameters of the QVM model and the second one is to train the neural network of the hybrid model.

6.3.1 Multiple Shooting

In this algorithm, we split the data set into overlapping intervals that has a defined group size. These groups are solved in parallel to fit the training data. The training objective is split in two loss functions : The first one computes the Mean Square Error (MSE); The second one, called continuity loss function, ensures that each group coincide with the others and that the final result is a continuous predicted set. The following picture illustrates what the algorithm is doing.

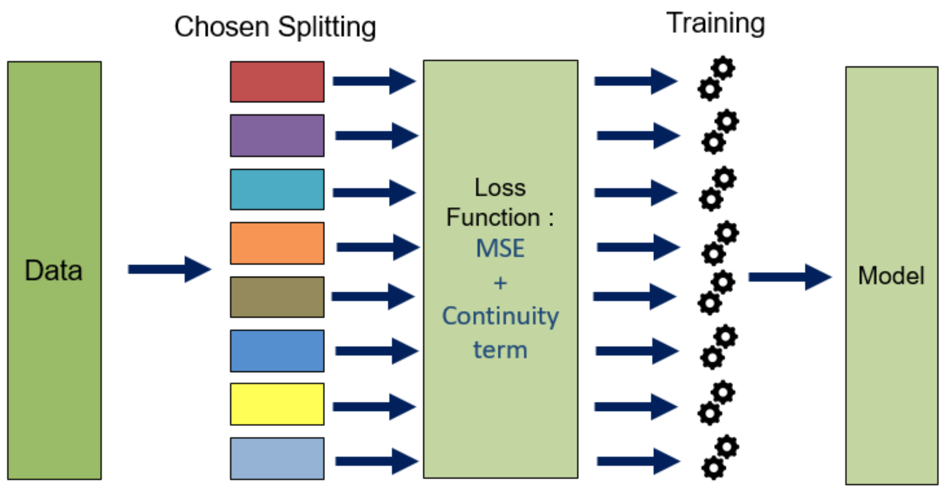

Figure 8: Multiple Shooting Explanation

6.3.2 Mini-Batching

That algorithm (9) proposes an interesting approach in term of accuracy. In each loop iteration, it selects some data points randomly so as to contribute to the loss function. The Mini-Batching algorithm does not use a continuity term in the loss function but only the MSE for the current batch.

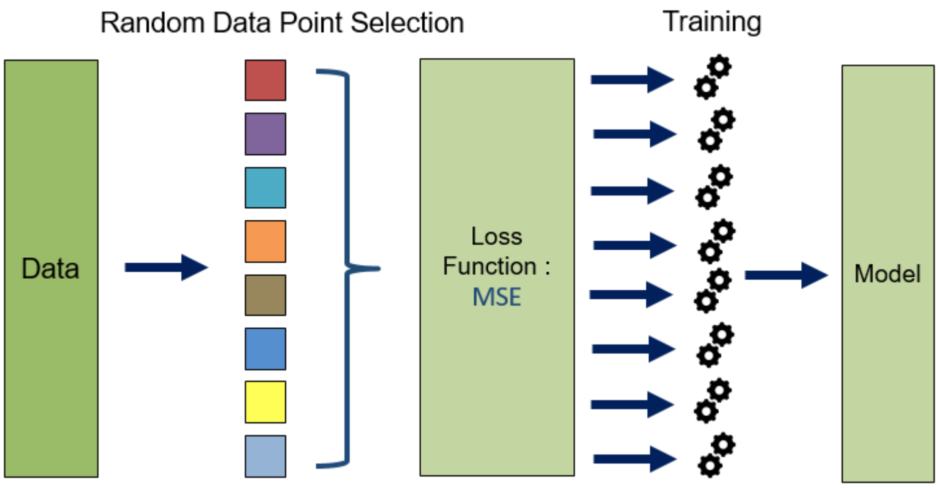

Figure 9: Mini Batching Explanation

6.3.3 Comparison

Mini-Batching provides a solution where the computation is made serial while Multiple Shooting is in parallel.

	Multiple Shooting	Mini-Batching
Small Group or Batch Size	Increase computing time Mandatory for high frequency dynamics	Increase computing time Lower risk of local minimum
Large Group or Batch Size	Decrease computing time Problem with high frequency dynamics	Decrease computing time Higher risk of global minimum

Table 1: Comparison of Multiple Shooring and Mini Batching

The choice of the algorithm has to be made regarding the data we have. Thus we are now going to analyze the form of the data.

6.4 Application to Synthetic Data

6.4.1 Reduced Physical Model

We first focused on the physical model but with $z_r = 0$.

To observe the evolution of the system we must put non zero initial condition because the equation are linearized around a certain equilibrium point. We thus choose : $Z_{l0} = [0 \quad 0,2 \quad 0 \quad 0 \quad 0]$.

We have the to following plots :

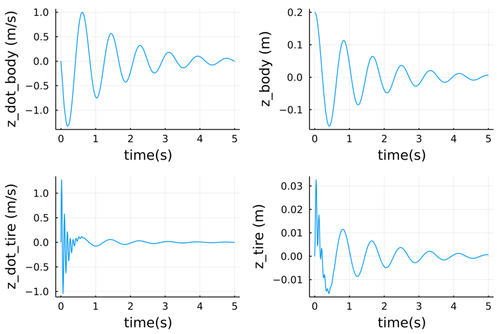

Figure 10: Reduced Physical Model Solution

As expected, the system amortize itself and converge to the equilibrium point.

6.4.2 Complete Physical Model

Then it has been provided a MAT file that contains both the input and the output of the QVM model. A sine-sweep for Z_r is used as input. We thus have this plots after solving the ODE Problem :

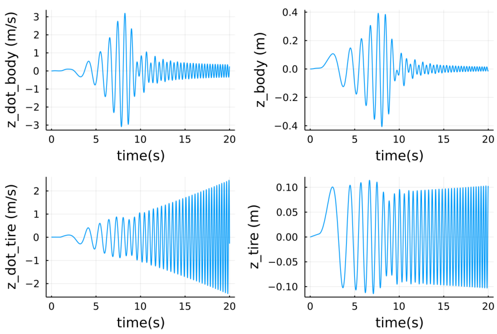

Figure 11: Complete Physical Model Solution

We can observe between 5s and 10s a phenomenon of resonance which is created by the body of the car. After that we observe the effect of the sine-sweep.

For the rest of the study, we will keep that complete physical model because it is more interesting to analyze.

6.5 Other Approaches on Parameter Estimation

In that final part we propose two other leads so as to obtain an estimation of the parameters for a given dynamic model. The research on that part would require additional test so as to see the efficiency of these techniques.

6.5.1 Stochastic Differential Equation

Basically a Stochastic Differential Equation (SDE) is the same as an ODE but the difference is that some coefficient are randomly chosen or are noisy.

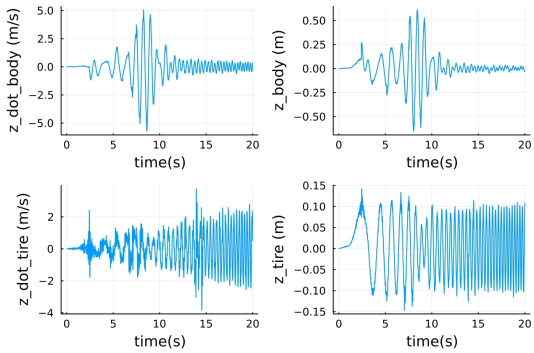

Figure 12: SDE Solution Plot

6.5.2 Bayesian Estimation

An other approach called "Bayesian Estimation" propose to estimate knowing only some boundary conditions.

7 Results obtained

7.1 Parameter Estimation

In a first step, so as to get used to applying the algorithms we focused on fitting of the QVM parameters model only.

7.1.1 Applied Multiple Shooting

We choose here a group size that last 2.5 seconds, we can observe that at the beginning the learning is not that easy but that at the end, it fits perfectly.

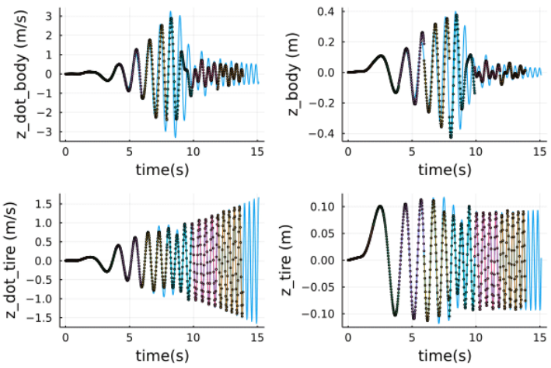

Figure 13: Fitting result at half of iteration maximum

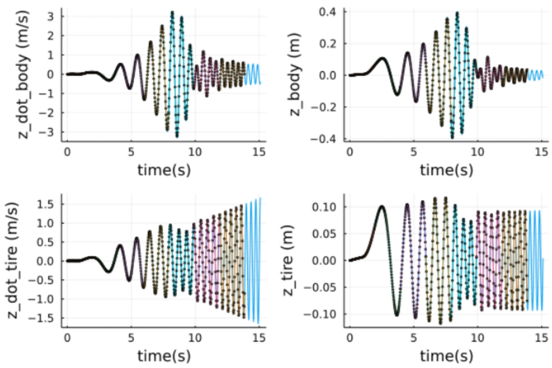

Figure 14: Fitting result at iteration maximum

Additionally, we can notice that this size of group is efficient. This situation is interesting because it allows to accumulate the advantages of both situation of Table 1.

Finally, the training time is around 20 minutes

7.1.2 Applied Mini-Batching

We can first see on figure 15 that the shape of the prediction data look a like a lot the real data at half of iteration maximum. This is due to the fact that the algorithm is choosing random point with Gaussian distribution at each epoch.

Also we can notice that the training time is around 5 minutes, which is much shorter. (8)

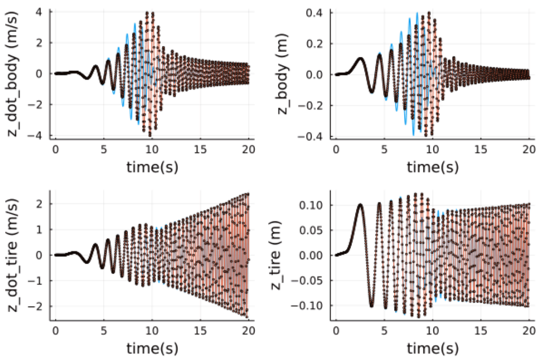

Figure 15: Fitting result at half of iteration maximum

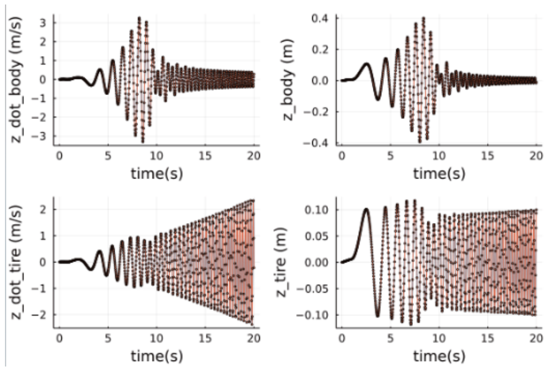

Figure 16: Fitting result at iteration maximum

7.2 Hybrid Modelling

As the equation (3) is showing we implement the Neural Network as the first state equation in the ODE System. The solving will then train the neural network that will fit the $\dot{z}$ variable.

7.2.1 Applied Multiple Shooting

We keep a group size that last 2.5 seconds and we obtain the following plots :

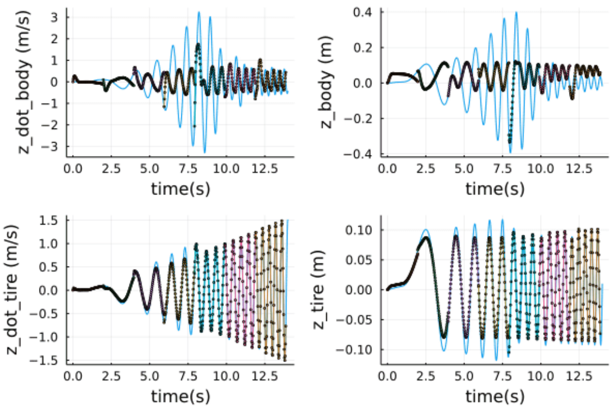

Figure 17: Fitting result at one third of iteration maximum

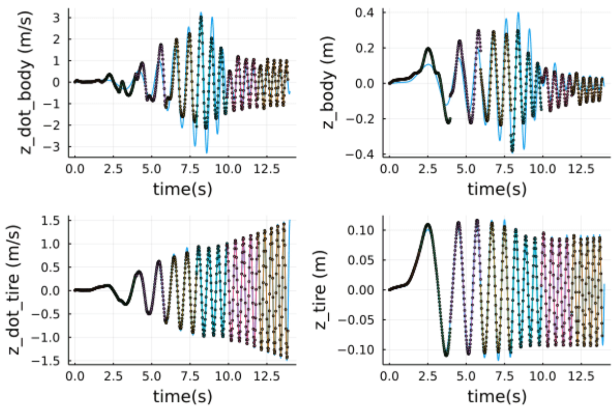

Figure 18: Fitting result at two third of iteration maximum

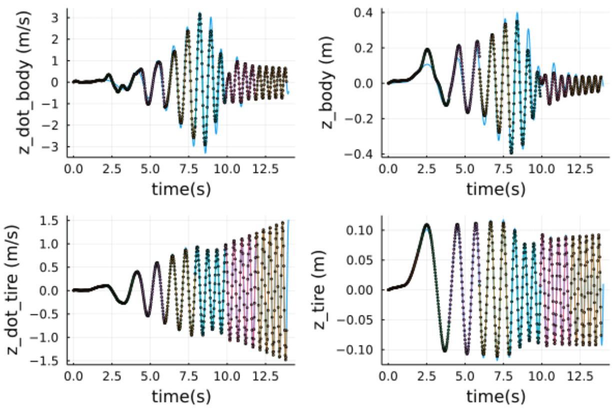

Figure 19: Fitting result at iteration maximum

Firstly, we observe at the beginning that the third and fourth states are solved rapidly which is logical because the equations are conserved. Also, these states are fitted quite perfectly regarding the form of the first and second state that has an influence on it.

Secondly, as expected the Neural Network is not efficient at the beginning nonetheless, it is learning step by step to finally fit the data.

Finally, the solving training time was approximately of 4 hours, which is due to the number of parameter to train and fit. In fact the NN has around 200 parameters, so computing the gradient for each parameter at each iteration is taking a long time.

7.2.2 Applied Mini-Batching

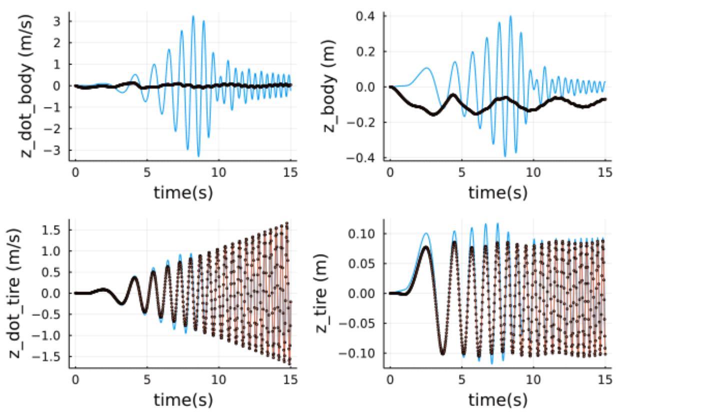

Figure 20: Fitting result at one third of iteration maximum

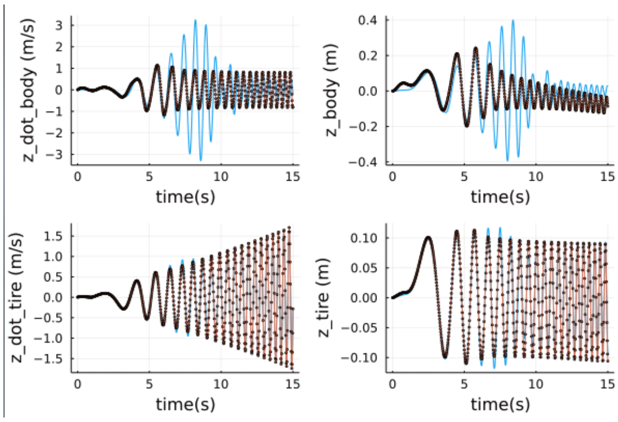

Figure 21: Fitting result at two third of iteration maximum

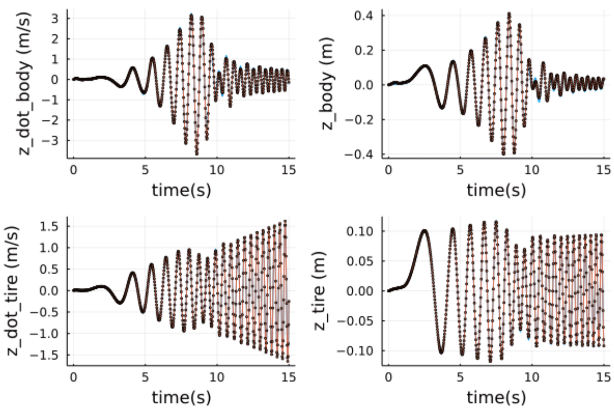

Figure 22: Fitting result at iteration maximum

We notice here that the learning of the neural network seems to be more efficient than before because the computing time was only about 30 minutes.

This shows that the mini batching has a better chance to reach a good minimum.

8 Contribution of the internship to the laboratory

The work we have accomplished will be useful for the researchers for there future work.
Nevertheless, even if we built algorithms which are efficient in the data we used, this does not mean that they are efficient for every type of data from real measurements.. That is why several research may be compulsory if they use other data.

9 Internship's personal contribution

First, I would say that the learning of Julia Programming Language in one week taught me how to catch the basics of a knowledge and to focus on what's important for my use of it. My supervisor is working for a while with NN so he taught me some "dos and dont's" that will be definitely useful. I have also learned a lot in term of NN Computation (classical programming) because until now I only manipulated NN on an interface in MATLAB.

Secondly, as the internship was fully remote it taught me how to work on my own and to understand the challenges of working from home. Of course, I had a first experiment with lockdowns and remote course at Ense3, but here that was different. I had to manage my daily schedule on my own so as to be as efficient as possible and not getting tired after four hours of work. For example, I used to take my meal at different hours each day so as to change my way of work and not getting use to routine.

Finally, before my internship I thought that it was impossible to work fully remote due to several known reasons such as: tiredness of staying seated all day, only knowing people via webcam and skype or also not being enough motivated. Guess what, I was totally wrong. In fact, as I was curious and ambitious about that internship, working fully remote was just a detail and it helped me to develop other skills that I never thought about.

10 Conclusion

As a conclusion I would say first that this internship brought me a lot on both scientific knowledge and work management. I also learned a lot on myself facing the work remote.
I am sure that the knowledge I have acquire on Julia programming, Neural Network and Parameter Estimation will be very useful for my study project.

11 References

References

[1] : DLR Official website https://www.dlr.de/

[2] : Official Website for Scientific Machine Learning in Julia : https://sciml.ai/

[3] : Semi-Active Suspension Control Design for Vehicles by S.M. Savaresi ; C. Poussot-Vassal ; C. Spelta ; O. Sename ; L. Dugard

[4] : Universal Differential Equations for Scientific Machine Learning by Christopher Rackauckasa, Yingbo Mac, Julius Martensend, Collin Warnera, Kirill Zubove, Rohit Supekara, Dominic Skinnera, Ali Ramadhana, and Alan Edelmana.

[5] : https://en.wikipedia.org/wiki/Julia_(programming_language)

[6] : Andreas Rößler. Runge–kutta methods for the strong approximation of solutions of stochastic differential equations. SIAM Journal on Numerical Analysis, 48(3):922–952, 2010.

[7] : Doing Scientific Machine Learning with Julia's SciML Ecosystem - Chris Rackauckas

[8] : Neural Ordinary Differential Equations by Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, David Duvenaud ; University of Toronto, Vector Institute

[9] : Scientific Machine Learning - Improvements to Neural and Universal Differential Equations - https://cn.julialang.org/jsoc/gsoc/sciml/