# PROCEEDINGS OF SPIE

SPIEDigitalLibrary.org/conference-proceedings-of-spie


# A comparison between regular and physics-informed neural networks based on a numerical multibody model: a test case for the synthesis of mechanisms


F. M. Bono, L. Radicioni, S. Cinquemani



F. M. Bono, L. Radicioni, S. Cinquemani, "A comparison between regular and physics-informed neural networks based on a numerical multibody model: a test case for the synthesis of mechanisms," Proc. SPIE 12489, NDE 4.0, Predictive Maintenance, Communication, and Energy Systems: The Digital Transformation of NDE, 124890L (25 April 2023); doi: 10.1117/12.2657981

Event: SPIE Smart Structures + Nondestructive Evaluation, 2023, Long Beach, California, United States

# A comparison between regular and physics-informed neural networks based on a numerical multibody model: a test case for the synthesis of mechanisms

Bono F.M[a], Radicioni L.[a], and Cinquemani S.[a]

[a]Politecnico di Milano, Milano, Italy

## ABSTRACT

Since 2019 researchers in the field of deep learning have been exploring the possibilities of Physics Informed Neural Networks (PINN). The training of regular neural networks (NNs) involved an optimization where the loss function depends exclusively on the dataset available. In PINN this loss function takes into account also the physics of the problem, if it is known and the governing equations are given. This paper explores the advantages of the use of PINNs with respect to regular NNs, in the privileged case where a multibody model is available. However, there is still uncertainty around how much weight should be associated with each of the two losses (data-driven loss and physics loss). Therefore, different weights for the two losses are considered and their effect on the performance of the model is evaluated. The research focuses on the synthesis of a four-bar mechanism for trajectory planning of a point belonging to the connecting rod. The objective is to generate a tool that synthesizes the mechanism topology given the desired trajectory. This preliminary study shows how PINN are suitable to automatize the synthesis of mechanisms, where regular NN would generally fail. Numerical analyses also demonstrate that a PINN learns relations from a physical numerical model in a more efficient way than a traditional NN.

**Keywords:** Physics informed neural networks, Synthesis of mechanism, Four-bar mechanism, Trajectory generation, Generative networks

## 1. INTRODUCTION

Among the synthesis of mechanisms problems, the *trajectory generation* involves the choice of a mechanical system to be used, such as cams or linkage mechanisms, and the definition of its geometrical parameters to achieve a desired motion law. In the past, the mechanisms synthesis for *trajectory generation* was obtained by using atlases, namely books recording the trajectory generated by many combinations of mechanical systems in different arrangements and geometries.[1] This procedure is regarded as a "graphical method", since the engineer needed to find the best arrangement by visual similarity among motion laws.

Another method involves the definition of certain *precision points* that must belong to the generated motion law to solve the problem in closed form. However, by doing so, only a limited number of precision points can be defined.[2] The computational power made available by modern calculators allows for generating and testing motion laws iteratively so that the mechanisms' synthesis can be addressed as an optimization problem.[3] All the methods just described requires the engineer to have a certain familiarity with the mechanisms and optimization problems. However, having assessed the outstanding performance achieved by artificial intelligence (AI) in the field of image, sound and text generation,[4–6] it seems worth exploring the possible use cases of such models for the generation of mechanisms. The aim of this paper is consequently the definition of a model which takes in input a desired motion law, and yields as output the geometrical properties of the mechanism to realize it. The main advantage offered by this solution would be having a fast and flexible tool, that can help designers in the synthesis of mechanisms. This study must be seen as a first step in that direction.

Moreover, to generate such a model, the well-known property of feed-forward neural networks to be universal approximators is exploited.[7] To enhance the performance of the neural network, the training procedure will account for the physics of the process, obtaining a form of "Physics-informed neural network".[8] At every

Further author information: Francesco Morgan Bono: E-mail: francescomorgan.bono@polimi.it

NDE 4.0, Predictive Maintenance, Communication, and Energy Systems: The Digital Transformation of NDE, edited by Norbert G. Meyendorf, Christopher Niezrecki, Ripi Singh, Proc. of SPIE Vol. 12489, 124890L · © 2023 SPIE · 0277-786X · doi: 10.1117/12.2657981

training iteration, the output of the network is used to run a simulation on multi-body software. This procedure yields a neural network able to approximate the behaviour of the multi-body simulation tool.

The paper is arranged in the following way: in section 2, the problem of trajectory generation is formally defined. In section 3, the multibody model is introduced, which is used for the generation of the training set and to physic-inform the network. In section 4, the physics-informed training is defined and explained. In section 5, the results are compared and discussed for networks trained with different logics. Finally, in section 6, some conclusions are drawn and future research scenarios are suggested.

## 2. FOUR-BAR MECHANISMS

*Linkage mechanisms* are systems composed of rigid bodies connected by rotational and prismatic pairs only. When the axes of the rotational pairs are all parallel to each other and orthogonal to the prismatic pairs axes, the linkage mechanism is said to be *planar*.[9] Among the planar linkage mechanisms, great effort has been historically dedicated to the study and characterization of the *four-bar* mechanism. In fact, most planar mechanisms commonly adopted can be tracked back to four-bar. For instance, the *slider-crank* or the *swinging glyph* are both regarded as special cases of four-bar. This is why the synthesis of four-bar mechanisms gained so much relevance in the past.
The kinematic chain of the four-bar mechanism is usually represented by four *links* connected by means of hinges (figure 1). The link $\overline{O_1O_2}$ is grounded and takes the name of *frame*. The opposite link $\overline{AB}$ is usually named *connecting rod*. The links contiguous with the frame, $\overline{O_1A}$ and $\overline{O_2B}$, are called *cranks* if they can perform a complete rotation, or *rockers* if they can only span a partial rotation. According to the relative lengths of the four links, a four-bar system can feature two cranks, two rockers or either one crank and one rocker. Specific details about four-bar mechanism types and classifications fall beyond the scope of the paper, and it is suggested to refer to dedicated literature for more information. It is worth noting that is commonly preferable to work with four-bar systems that have at least one crank. In this way, the system can be driven by a motor or a rotating shaft directly connected to the crank, whereas acting on a rocker is evidently more complicated.
The motion of the connecting rod plays a significant role in studying four-bar mechanisms, which corresponds to the *rototranslation* of a rigid body with two points bounded to circular paths. Of particular interest are the trajectories traced by the coupling rod's points (figure 2), which are $6^{th}$ degree curves and might substantially vary for different links arrangements.

### 2.1 Synthesis of mechanisms for trajectory generation

The synthesis of mechanisms is formally defined as follows: given a *desired* motion law $y = f(\alpha)$, where $\alpha$ refers to the master angle, the objective is the generation of a curve $y\prime = g(\alpha)$, which approximates the function $f$. Evidently, the two curves cannot be perfectly equal, and so this is achieved by imposing that the *structural error*

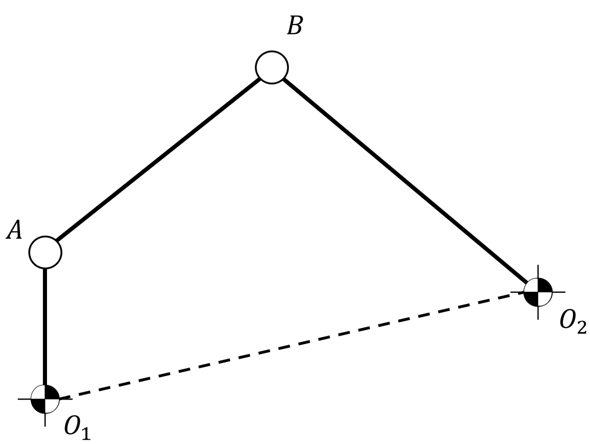


Figure 1. Four-bar mechanism.

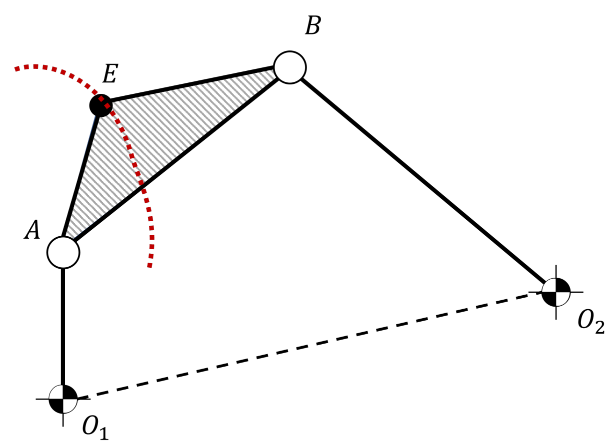


Figure 2. Example of Arc of trajectory traced by point $E$, belonging to the connecting rod.

$e(\alpha) = f(\alpha) - g(\alpha)$ remains below a certain value.
The procedures for the synthesis of mechanisms are usually classified as *direct* or indirect methods, depending on the number of iterations required to solve the synthesis problem. If an algorithm defines the mechanism in one passage, the method is *direct*. The *precision points* method is the most famous direct method.[10] The procedure involves the nullification of the structural error in correspondence of certain points only, which take consequently the name of *precision points* (figure 3). However, this method requires a manual check of the outcome, since the procedure does not account for the behaviour of the curve $g(\alpha)$ in points different from the *precision* ones. Moreover, the problem can be solved in closed form for a limited number of precision points only, which corresponds to the number of degrees of freedom of the design problem.

Given a *desired trajectory*, synthesising a four-bar mechanism whose coupling rod follows the desired curve is called *trajectory generation*. The other common problems related to the synthesis of mechanisms are the *function generation* and the *planar motion generator*,[11] which are not treated here for simplicity. In the case of four-bar mechanisms, the problem involves 9 unknowns, that is, the position of $O_1$ and $O_2$ with respect to the reference frame, and then the length of the links $\overline{O_1A}$, $\overline{O_2B}$, $\overline{AB}$, $\overline{AE}$ and $\overline{BE}$ (Figure 2). This means that a maximum of 9 conditions can be defined. One can therefore introduce 9 precision points or fewer precision points and the tangent condition is some of them, so that the overall number of conditions imposed is 9.
Under the assumption that the $\overline{O_1A}$ rotates with constant angular velocity, the problem involves 10 unknowns (the 9 just introduced and the initial angle of $\overline{O_1A}$), but only 5 conditions can be introduced. In fact, each condition counts twice (the position of point E for a given value of the angle $\overline{O_1A}$). The conditions can involve precision points, velocity vectors or acceleration vectors, but the overall number of conditions cannot be greater than 5.[9]

In the case of *indirect* methods, the optimal design of mechanisms is achieved by successive evaluations of the system. The synthesis of the four-bar mechanism can consequently be addressed as an optimization problem.[12] Once a *objective function* is introduced to *quantify* the distance between the current trajectory with respect to the desired one, the 9 (or 10) parameters of the mechanism are progressively tuned and the objective function is evaluated at every iteration. Eventually, an optimal solution (*global* or *local*) is achieved.

The success of generative networks in recent years for the generation of images, texts and sounds, is driving researchers to train and test models able to artificially produce objects in the most diverse fields. This motivated us to evaluate the construction of a generative model also in the context of the synthesis of mechanisms, and in particular for the *trajectory generation* problem.

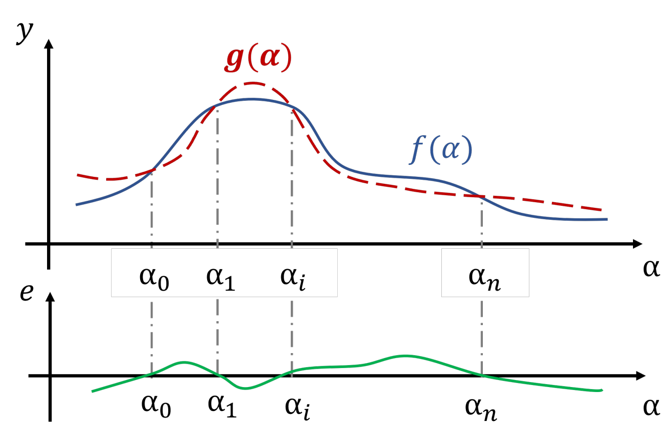


Figure 3. The precision points methods imposes the nullification of the structural error $e(\alpha)$ in correspondence of the precision points.

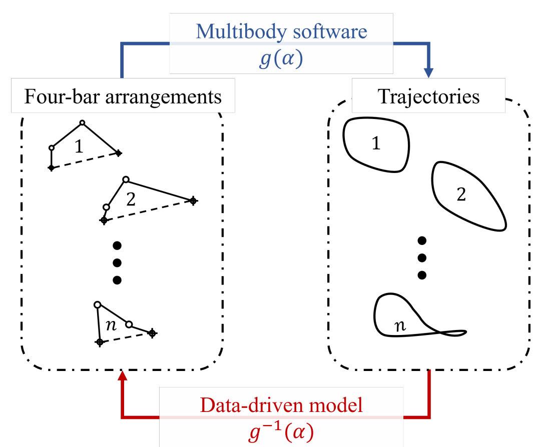


Figure 4. A multibody software generates the trajectories for different four-bars arrangements. A data-driven model can be train to do the opposite.

## 3. TRAINING SET AND MULTIBODY MODEL

Multibody softwares are extremely versatile tools that assist engineers in systems design. In the context of the synthesis of mechanisms, they allow fast prototyping by computing the required physical variables in the desired time span. If the problem considered is the trajectory generation of a four-bar, whose link is driven by a constant angular velocity, the multibody tool takes as input the geometrical parameters of the four-bar, and it yields in output the trajectory generated by the point $E$ of the connecting rod. This means that the software applies the function $g(\alpha)$ so that:

$$x_E(\alpha), y_E(\alpha) = g(a_1, a_2, \cdots, a_{10}, \alpha) \tag{1}$$

Where $a, b, \cdots, a_{10}$ are the 10 parameters introduced in section 2.1 and $\alpha$ the master angle of the driver link. However, the synthesis involves the definition of the four-bar parameters given the desired function $x_E(\alpha), y_E(\alpha) = f(\alpha)$, which means that the function in 1 must be inverted, so that to obtain the mechanism topology:

$$a_1, a_2, \cdots, a_{10}, \alpha = g^{-1}(x_E(\alpha), y_E(\alpha)) \tag{2}$$

But the function $g$ is not invertible. The objective of this research is consequently the inversion of the function $g$.

### 3.1 Data-driven modelling

One possible approach for the definition of the function $g^{-1}(\alpha)$ might require the use of a parametric model fitted onto a dataset containing many possible trajectories and the corresponding parameters of the four-bar that generated them. In the past, this collection of curves took the name of *atlases*. Atlases were used for the synthesis of mechanisms by visual similarity between the desired curve and the ones reported in the atlas. Today, the help of calculators allows for generating many possible combinations of four-bars by varying the parameters iteratively. The dataset collected in this way might then be used to train a data-driven model, which takes in input the curves and estimates the parameters of the four-bars to generate them (figure 4). To do so, the properties of neural networks as universal approximators are exploited,[13] whereas a multibody dynamics software (MSC ADAMS) is used for the collection of the training set.

### 3.2 Multi-body model

The model considered to simulate the behaviour of the four-bar mechanism is the one in figure 5, implemented in the software MSC ADAMS (figure 6). A constant angular velocity is imposed onto the link $\overline{O_1A}$ so that the number of parameters to define the system is 10. However, to avoid the problem known as the *curse of dimensionality*, the number of parameters to be defined has been reduced. Briefly, the curse of dimensionality states that the volumetric density of data points rapidly reduces as the number of dimensions of the space increases.[14] The immediate consequence is that the dataset collected becomes sparse and not representative

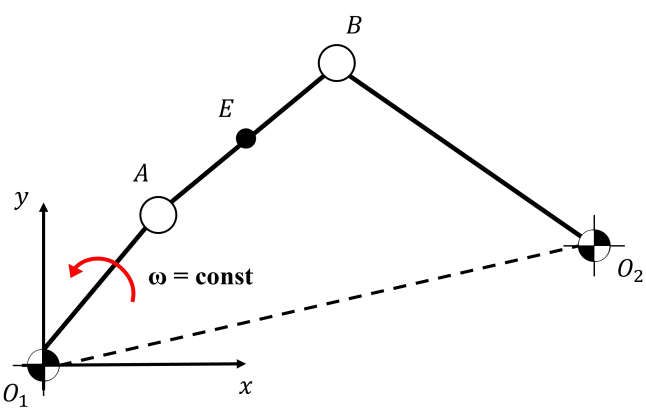

Figure 5. Model of the four-bar mechanism used for the collection of the training set.

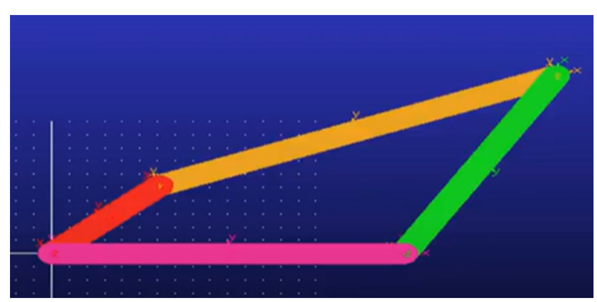
Figure 6. Four-bar mechanism in ADAMS view.

| | $X_A$ | $Y_A$ | $X_B$ | $Y_B$ | $X_{O_2}$ | $Y_{O_2}$ |
|---|---|---|---|---|---|---|
| Initial value [mm] | 0 | 50 | 200 | 50 | 200 | 0 |
| End value [mm] | 140 | 190 | 340 | 190 | 340 | 90 |
| N°points evaluated | 15 | 15 | 15 | 15 | 15 | 10 |

Table 1. Candia Bridge - Correlation between Inclinometers

anymore of the phenomenon under study.
The hinge $O_1$ is centred in the origin of the reference frame, while the point of the coupling rod $E$ is located in the middle of $\overline{O_1A}$. This reduced the number of parameters of the system by 4. The number of remaining parameters is then 6, which are the coordinates of $O_2$, $A$ and $B$ (or equivalently, the coordinates of $O_2$, the length of $\overline{O_1A}$, $\overline{AB}$, $\overline{O_2B}$ and the initial angle of $\overline{O_1A}$).
Every single variable is set to vary in the ranges indicated in table 1, with a pace of 10 mm, whereas all the other five parameters are kept constant. The master angle $\alpha$ of the link $\overline{O_1A}$ performs a rotation of 30 degrees for 500 integration steps. Eventually, by multiplying the number of points evalauted (last row of table 1) one gets the number of data points "explored": $15^5 \times 10 = 7,593,750$.
The training set collected should count therefore 7,593,750 observations. However, all the simulations that ended up in singularity points have been ignored, so the number of observations in the training set is lower. Every observation is characterized by the six parameters of the four-bar and the correspondent trajectory generated by point $E$ of the connecting rod, namely an array of size $2 \times 500$ (the value of x and y for every alpha).

One might argue that the in case of four-bar mechanisms, the use of a multibody tool is not strictly necessary, since the equations of the curve defined by the point $E$ as a function of all the four-bar parameters can be explicitly defined. However, to propose a method as general as possible, it has been decided to adopt a multibody software to generate the training set. In this way, the same approach can be easily extended to other sets of problems that involve multibodies, and not necessarily for the *trajectory generation* only.

### 3.3 Neural network structure

Once the training set has been generated, the data-driven model must be defined. As anticipated, the model chosen is a neural network. The package adopted is TensorFlow, and the network is assembled with Keras API. The model counts five layers:

1. A Separable Convolutional 1D layer. This layer takes in input a tensor of size (2, 500), namely the trajectories, and it applies 1D-convolutional windows separate on every channel. Then, it mixes the channel by a point-wise multiplication

2. A Maxpooling layer, to reduce the space of the convolutional map.

3. Two Gated Recurrent Unit layers are applied in succession. The first counts 64 cells, the second 16 cells. The application of convolutional plus recurrent layers is proved to be particularly efficient in the analysis of time series.[15]

4. The last is a six-neuron Dense layer. Every neuron contains the coordinates of points $O_2$, $A$ and $B$. These are the targets of the training, namely the variables that the network must estimate.

The training of the model can finally be carried out. However, in this way, the model is fully data-driven, which means that only the objects belonging to the training set "drive" the gradient descend.[16] Nevertheless, to generate the training set, a *physical* model has been used, which means that we have access to the physics of the problem. It has been proven that it is possible to *inform* the network of the physical law during training, improving in such a way the performance of the trained model.

# 4. PHYSICS INFORMED TRAINING

The definition Physics Informed Neural Networks (PINN) have been first introduced by Raissi et al.[8] to refer to neural networks obtained by implementing the system partial differential equations in training, taking then the derivatives with respect to the input coordinates. This approach seemed to introduce a regularization mechanism into the backpropagation algorithm, allowing the training of neural networks even with small datasets.

In this paper the training process of the neural network is carried out by backpropagating two losses: the *data-driven* loss, which directly derives from the dataset collected (section 3.2) and the *physics-based* loss. The latter is obtained at every batch by evaluating the input trajectories with respect to the ones generated by the parameters estimated by the neural network in training:

$$(\hat{a}_{i1}, \cdots, \hat{a}_{i6}) = h_\theta(x_{i1}, \cdots, x_{i500}, y_{i1}, \cdots, y_{i500}) \tag{3}$$

$$\mathcal{L}_{data} = \frac{1}{N}\sum_{i=1}^{N}\left(\sum_{j=1}^{6}(a_{ij} - \hat{a}_{ij})^2\right) \tag{4}$$

$$(\hat{x}_{i1}, \cdots, \hat{x}_{i500}, \hat{y}_{i1}, \cdots, \hat{y}_{i500}) = g(\hat{a}_{i1}, \cdots, \hat{a}_{i6}) \tag{5}$$

$$\mathcal{L}_{physics} = \frac{1}{N}\sum_{i=1}^{N}\left(\sum_{k=1}^{500}(x_{ik} - \hat{x}_{ik})^2 + (y_{ik} - \hat{y}_{ik})^2\right) \tag{6}$$

Finally, the overall loss is:

$$\mathcal{L}(\theta) = (1-\lambda)\mathcal{L}_{data} + \lambda\mathcal{L}_{physics} \tag{7}$$

In equation 3, $h_\theta$ refers to the function applied by the neural network characterized by the vector of weights $\theta$. The network takes in input the trajectory (defined by the coordinates x and y for every timestep considered) and estimates the six parameters of the four-bar mechanism: $\hat{a}_{i1}, \cdots, \hat{a}_{i6}$. The subscript $i$ indicated the $i^{th}$ observation of the batch.
In equation 4 the data-driven loss is obtained as the mean square error of the estimated parameters $\hat{a}_{ij}$ with respect to the correct ones $a_{ij}$. $N$ is the number of elements in a batch.
In equation 5 the function $g$ refers to the kinematic laws applied by the multibody analysis: the values estimated by the neural networks $\hat{a}_{i1}, \cdots, \hat{a}_{i6}$ are used to run a simulation, so to obtained the correspondent estimated trajectories $\hat{x}_{i1}, \cdots, \hat{x}_{i500}, \hat{y}_{i1}, \cdots, \hat{y}_{i500}$.
Equation 6 indicates that the physics loss is calculated as the euclidean distance between the points of the estimated trajectory and the correct one.
In conclusion, equation 7 shows that the overall loss is computed as a weighted sum of the data-driven loss and the physics-based loss. $\lambda$ is a value that spans between 0 and 1, which establishes how much weight is attributed to the physics-based loss. The procedure for the computation of the loss for one batch of data is summarised in figure 7. In figure 8 the overall procedure to physics inform the neural network: the creation of the training set, computation of the data-driven and physics-based losses and the action of the optimizer in updating the weights.

Many variations to the formulations and definitions of PINN have been introduced in the last years,[17] but for consistency, we decided to use the name "physics informed" for the methodology proposed in this paper, although some details must be discussed:

- In the form originally proposed, NN were "physics informed" by implementing partial differential equation (PDE) without solving them.[17] In this paper instead, the physics-based loss is computed by solving the kinematic relationships for given four-bar parameters.
- The methodology proposed in this research is substantially mesh-free, which is a property peculiar to PINNs as proposed in the literature.
- Eventually, both approaches are partially data-driven and partially based on physical laws.

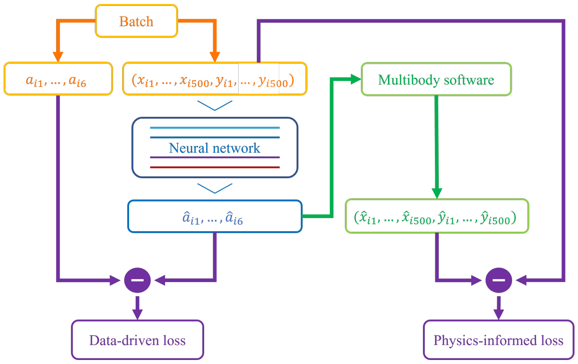


Figure 7. Summary on the computation of physics-based loss and data-driven loss for training. Focus on one element of a batch.

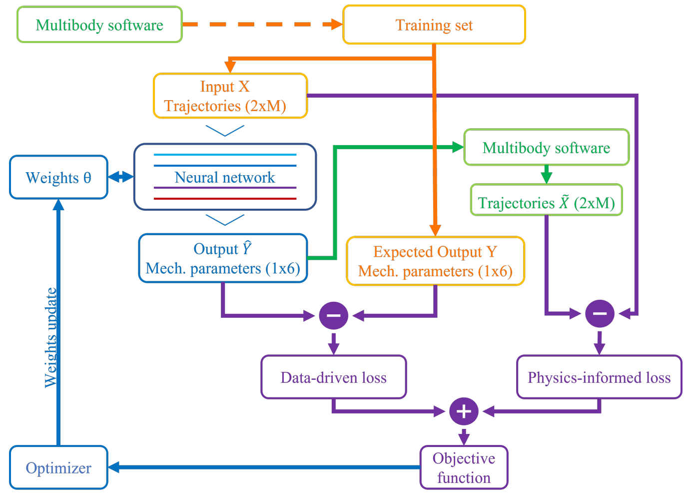


Figure 8. Training set generation, training procedure of the physics-informed neural network, computation of the overall loss and weights update.

For all these considerations, it looked suitable to use the terminology "physics informed" also for the neural network under study, although it is essential remarking that in literature the terminology mainly refers to partial differential equations.

## 5. PHYSICS INFORMED VS REGULAR NEURAL NETWORKS

There are a few *hyper-parameters* that regulate the training process: the number and type of layers in the network, the nature and learning rate of the optimizer chosen, etc. However, particular attention should be given to the *regularization term* $\lambda$, which assesses how much the physics-informed loss must be accounted for in equation 7. Therefore, the training procedure described in section 4 is applied for the training of three generative neural networks:

- The first neural network is trained with $\lambda = 0$. This forces the network to use only the data-driven loss so that the training follows the traditional scheme.
- A second neural network is trained with $\lambda = 0.3$. In this way, much of the gradient descent is *driven* by the data loss, whereas the effect of the physics injection is minor.

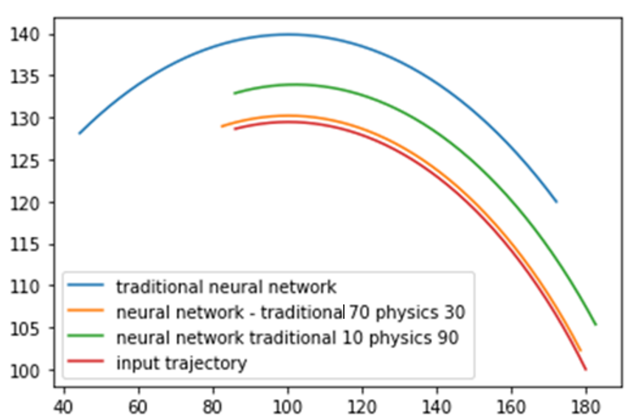


Figure 9. Example of trajectories generated by the three neural networks.

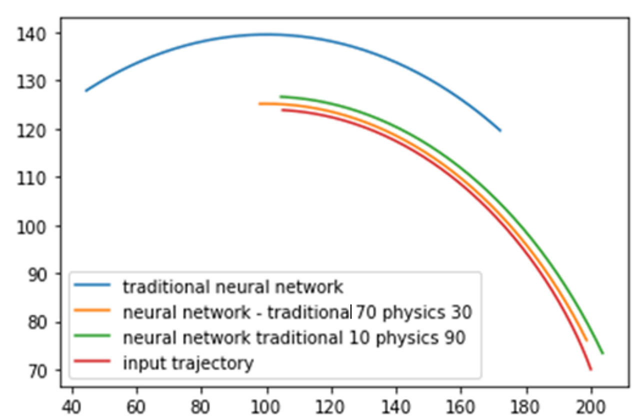


Figure 10. Another example of trajectories generated by the three neural networks.

- A third neural network is trained by accounting for $\lambda = 0.9$. The resulting network massively depends on the physics contribution.

To compare the three approaches, the following method is adopted.

1. A random combination of the four-bar parameters, among the ranges introduced in table 1 is selected, and the *true trajectory* plotted by the multibody software.
2. The *true trajectory* are fed into the three neural networks to obtain the six estimated parameters for each of them.
3. The estimated parameters are passed to the multibody software to obtain the three estimated trajectories.

This procedure is applied 100 times. In figure 9 and 10 two results that are representative of case history. The original (true) trajectory is depicted in red. In blue, the trajectory generated by the neural network fully data-driven ($\lambda = 0$). In orange, the trajectory generated by the network trained with $\lambda = 0.3$ Finally, the trajectory generated by the network mostly physics-informed ($\lambda = 0.9$) is coloured in green.
From the study, some considerations can already be drawn, even though it is not feasible to derive generally valid conclusions in this phase of the research. It seems evident that for this specific case study, the "traditional" neural network is incapable of estimating the mechanism's parameters. Then, it is not able to invert the function $g$ as defined in section 3.
Nevertheless, the two *physics informed* networks seem able to approximate the function $g^{-1}$. In detail, the network with a minor physics contribution ($\lambda = 0.3$) yields consistently better results, suggesting that the contribution of the data-driven loss is still crucial for the proper training of the network.
It is crucial to underline that this training procedure requires running simulations at every data batch evaluated in training, which means that, in this case, the training time increases by a factor of 5 circa.

## 6. CONCLUSION

This paper proposes a possible approach for the automatic synthesis of mechanisms, based on a generative physics-informed network applied to the *trajectory generation* problem. Although not strictly necessary for the scope, a multibody software has been exploited, so that the same methodology can be extended to other similar cases (which goes beyond the simple kinematic relationships among rigid bodies).
The results show how the implementation of the physics-informed loss greatly improves the final performance of the generative network. Moreover, it seems that a lower contribution of the *physics* loss with respect to the *data* loss yields better performances. Nevertheless, it is still not clear if there exists an optimal value for the weight attributed to the physical loss.
It is worth mentioning also that this improvement in performance comes at a higher training time.
The next steps on the topic of automatic mechanism synthesis must involve the generalization of this approach to include more four-bar parameters, so as to increase the plethora of possible trajectories that can be generated. This might be achieved by increasing the range of the parameters already considered, by adding $\overline{AE}$ and $\overline{BE}$ among the variables and by varying the span of the angle $\alpha$. The performance of a *generative adversarial network* (GAN) should be accounted for in future works as well.

## REFERENCES

[1] Cabrera, J., Simon, A., and Prado, M., "Optimal synthesis of mechanisms with genetic algorithms," *Mechanism and Machine Theory* **37**(10), 1165–1177 (2002).

[2] Sandor, G. N., [*A general complex-number method for plane kinematic synthesis with applications*], Columbia University (1959).

[3] Fang, W. E., "Simultaneous type and dimensional synthesis of mechanisms by genetic algorithms," in [*International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*], **12846**, 35–41, American Society of Mechanical Engineers (1994).

[4] Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., and Lee, H., "Generative adversarial text to image synthesis," in [*International conference on machine learning*], 1060–1069, PMLR (2016).

[5] Arık, S. Ö., Chrzanowski, M., Coates, A., Diamos, G., Gibiansky, A., Kang, Y., Li, X., Miller, J., Ng, A., Raiman, J., et al., "Deep voice: Real-time neural text-to-speech," in [*International conference on machine learning*], 195–204, PMLR (2017).

[6] Iqbal, T. and Qureshi, S., "The survey: Text generation models in deep learning," *Journal of King Saud University-Computer and Information Sciences* (2020).

[7] Hornik, K., Stinchcombe, M., and White, H., "Multilayer feedforward networks are universal approximators," *Neural networks* **2**(5), 359–366 (1989).

[8] Raissi, M., Perdikaris, P., and Karniadakis, G. E., "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational physics* **378**, 686–707 (2019).

[9] Magnani, P. L. and Ruggieri, G., [*Meccanismi per macchine automatiche*], Utet (1986).

[10] Erdman, A. G., "Three and four precision point kinematic synthesis of planar linkages," *Mechanism and Machine Theory* **16**(3), 227–245 (1981).

[11] Sandor, G. N. and Erdman, A. G., [*Advanced mechanism design v. 2: Analysis and synthesis*], Prentice-Hall (1984).

[12] Sohoni, V. and Haug, E., "A state space technique for optimal design of mechanisms," (1982).

[13] Scarselli, F. and Tsoi, A. C., "Universal approximation using feedforward neural networks: A survey of some existing methods, and some new results," *Neural networks* **11**(1), 15–37 (1998).

[14] Daum, F. and Huang, J., "Curse of dimensionality and particle filters," in [*2003 IEEE aerospace conference proceedings (Cat. No. 03TH8652)*], **4**, 4_1979–4_1993, IEEE (2003).

[15] Zhang, C., Song, D., Chen, Y., Feng, X., Lumezanu, C., Cheng, W., Ni, J., Zong, B., Chen, H., and Chawla, N. V., "A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data," in [*Proceedings of the AAAI conference on artificial intelligence*], **33**(01), 1409–1416 (2019).

[16] Ruder, S., "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747* (2016).

[17] Cuomo, S., Di Cola, V. S., Giampaolo, F., Rozza, G., Raissi, M., and Piccialli, F., "Scientific machine learning through physics–informed neural networks: where we are and what's next," *Journal of Scientific Computing* **92**(3), 88 (2022).