Physics-Informed Neural Networks for General Three-Body Dynamics

Angel Martinez [1]
Department of Computer Science
California State University, Chico, U.S.
angelmartinez20@csuchico.edu

Abstract

The three-body problem describes the gravitational motion of three interacting bodies, and is known for its chaotic behavior and sensitivity to initial conditions. The problem has no analytical solution, therefore traditional methods use numerical integration which is computationally expensive and prone to instability over time. Recent work has shown neural networks offer a faster, more efficient, and accurate alternative, but they have mainly been applied to a simplified three-body problem. As a result, this paper explores the application of Physics-Informed Neural Networks (PINNs) to general three-body dynamics. Our results demonstrate that PINNs are able to learn some of the geneal structure a three-body trajectory takes. But given the chaotic nature of the three-body problem, there is much variability in their different trajectories. As a result, much training data is needed to accurately capture general three-body dynamics. This makes numerical integrators the most reliable method for precise long-term simulations.

1. Introduction

For centuries, the three-body problem has posed a challenge in celestial mechanics. The problem was initially presented by Sir Isaac Newton in his *Principia Mathematica* (Newton, 1687), where he described how the motion of bodies gravitationally interacts with one another. Many physicists and mathematicians throughout history have found analytical solutions to specific instances of the three-body problem. However, it was not until 1892 that Poincaré demonstrated that the general three-body problem lacks a closed-form analytical solution in general (Poincaré, 1892). This is due to the bodies' complex interactions that produce chaotic behavior that are highly sensitive to initial conditions. Mathematician Sundman did develop a theoretically valid solution to the three-body problem, but it's impractical due to its infinite series of expansions causing very slow convergence (Sundman, 1913).

Given the difficulties of deriving an analytical solution, numerical integration methods have become the primary approach for approximating solutions to the three-body problem. Numerical integration breaks down the problem into short time steps that initially compute each body's acceleration, then numerically integrates along time to find their new positions and velocities at the next time step. Such calculations are nevertheless computationally expensive and suffer from precision errors over time. Additionally, each body's position and velocity are updated in discrete steps that result in a sequential updating process. This creates a slight mismatch with the actual continuous motion of the bodies over time (Valtonen et al., 2016).

Due to these limitations, current research is exploring whether neural networks can provide a better alternative than numerical methods. As later discussed in Section 3, neural networks have shown promising results. However, previous studies have typically simplified the three-body problem by assuming:

- All bodies have the same mass
- All bodies start with zero initial velocity
- Simulations are in two dimensions
- Only short-term predictions are made (usually across 10 seconds)

These constraints limit real-world applicability. Therefore, this paper investigates whether Physics-Informed Neural Networks (PINNs) can provide a better approximation alternative than numerical integration in general three-body dynamics.

2. Background

In this section, a lower-level background is provided for the main components used in this research paper. From the

equations explaining the three-body problem to how PINNs differ from traditional neural networks, this section highlights the fundamental principles behind each component.

2.1. Three-Body Problem

To determine the positions of the three bodies, various mathematical formulations have been developed to track their coordinates within a reference frame. One of them is the widely used Barycentric Coordinate System (Broucke & Lass, 1973), which is a differential system of equations defining the motion of bodies relative to their center of mass. The corresponding equations are as follows:

$$\begin{aligned}
\ddot{\mathbf{r}}_1 &= -m_2 \frac{\mathbf{r}_1 - \mathbf{r}_2}{|\mathbf{r}_1 - \mathbf{r}_2|^3} - m_3 \frac{\mathbf{r}_1 - \mathbf{r}_3}{|\mathbf{r}_1 - \mathbf{r}_3|^3}, \\
\ddot{\mathbf{r}}_2 &= -m_1 \frac{\mathbf{r}_2 - \mathbf{r}_1}{|\mathbf{r}_2 - \mathbf{r}_1|^3} - m_3 \frac{\mathbf{r}_2 - \mathbf{r}_3}{|\mathbf{r}_2 - \mathbf{r}_3|^3}, \\
\ddot{\mathbf{r}}_3 &= -m_1 \frac{\mathbf{r}_3 - \mathbf{r}_1}{|\mathbf{r}_3 - \mathbf{r}_1|^3} - m_2 \frac{\mathbf{r}_3 - \mathbf{r}_2}{|\mathbf{r}_3 - \mathbf{r}_2|^3}.
\end{aligned} \tag{1}$$

m_n represents the mass of the n-th body, while $\mathbf{r}_n$ is the relative position vector. The Euclidean norm $\|\mathbf{v}\|$ is defined as the square root of the sum of the squares of its components. This gives the straight-line distance between the two bodies in question within its dimension space. These equations are then reduced to the second derivative of $\mathbf{r}$ with respect to time, which represent the acceleration of the current body. Given acceleration, velocity can be updated, allowing the new positions to be found.

Equations (1) describe the motion of bodies due to gravitational forces between them. If an analytical solution were available, exact formulas for the positions $\mathbf{r}_1(t), \mathbf{r}_2(t), \mathbf{r}_3(t)$, could be derived. Another formulation for the dynamics of a system can be given in terms of the Lagrangian formalism (Broucke & Lass, 1973) which is expressed as:

$$\begin{aligned}
\mathcal{L} = &\frac{1}{2}\left[m_1\dot{\mathbf{r}}_1^2 + m_2\dot{\mathbf{r}}_2^2 + m_3\dot{\mathbf{r}}_3^2\right] \\
&+ \left[\frac{m_2 m_3}{|\mathbf{r}_3 - \mathbf{r}_2|} + \frac{m_1 m_3}{|\mathbf{r}_1 - \mathbf{r}_3|} + \frac{m_1 m_2}{|\mathbf{r}_2 - \mathbf{r}_1|}\right].
\end{aligned} \tag{2}$$

The first term describes the system's kinetic energy, as it accumulates the energy associated with each body's velocity. Then the second term (potential energy), is calculated based on the distances between each pair of bodies and their mutual gravitational interactions. Adding these two terms together results in the total energy of the three-body system, which according to the principle of energy conservation, the total energy of interacting bodies in a closed system remains constant (Pitts, 2021). Therefore, the energy at any time t should match the initial energy of the system. This theorem is what allows us to use Lagrangian equation as a method to asses accuracy of approximate solutions to the three-body problem.

2.2. Artificial Neural Networks

The underlying main framework of PINNs relies on Artificial Neural Networks (ANNs), a Machine Learning method that mimics the biological structure of neurons in the human brain. Dating from the 1940s, researchers first developed a learning rule to explain the behavior of interconnected neurons. This later led to Minsky and Papert developing a perceptron (single-layer neural network), and in 1986, multilayer perceptrons and the backpropagation algorithm were developed (Walczak & Cerpa, 2003).

Given a set of inputs, data is processed in an ANN by a sequence of computation fields where every connection carries a weight value. This output predicted by the network is then compared with the actual value which defines the loss function. The provided error is fed through a gradient descent optimizer with the purpose of decreasing the Mean Squared Error (MSE). This process improves the network's accuracy by adjusting the weights during the training phase until the network's output meets the expected results.

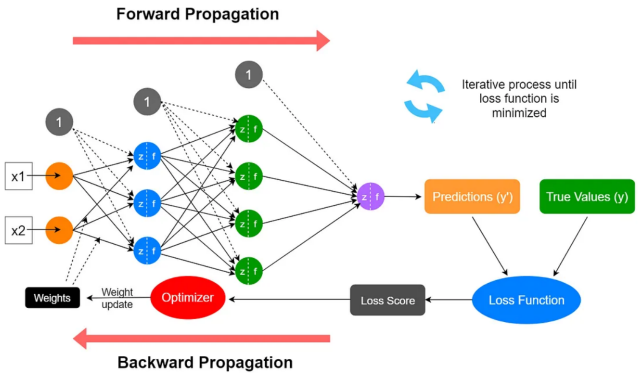

Figure 1. ANN architecture with both forward and backward prorogation training process (Soak, 2024).

2.3. Physics-informed Neural Networks

PINNs are an extension of ANNs where its term was first coined in 2017 by (Raissi et al., 2017). PINNs are directly utilized to address forward and inverse problems of nonlinear partial differential equations (PDEs). Taking the same structure as Figure 1, PINNs simply include PDEs constraints directly into the loss function which allows the network to better correct its predicted values. This integration enables the neural network to learn the intrinsic physical laws that govern the problem, rather than simply curve-fit the existing points of the problem. The updated

loss function is composed of two parts:

$$MSE = MSE_u + MSE_f. \tag{3}$$

MSE_u is the discrepancy between the neural network output and actual values of the known training data. This section checks whether or not he network correctly maps given points to the given values. As shown in equation (4), $u(t_u^i, x_u^i)$ is the predicted value of the network and, and u^i is the data value itself.

$$MSE_u = \frac{1}{N_u} \sum_{i=1}^{N_u} \left| u(t_u^i, x_u^i) - u^i \right|^2. \tag{4}$$

The second term, MSE_f is the discrepancy between the network's prediction and the governing equation of the PDE. This term ensures that predictions made by the network are adherent to physical laws embodied in the PDE. As depicted in equation (5), $f(t_f^i, x_f^i)$ is the residual of the PDE at the collocation points, which are sample points from the system.

$$MSE_f = \frac{1}{N_f} \sum_{i=1}^{N_f} \left| f(t_f^i, x_f^i) \right|^2. \tag{5}$$

3. Literature review

Given that there are no analytical solutions to the three-body problem, a number of approximation techniques have been investigated to determine the orbits of the bodies. Recent research has looked at various numerical integration techniques as well as machine learning techniques, particularly using various types of neural networks.

3.1. Numerical Integration Methods

There are many types of numerical integrators where each of them calculates the weighting schemes and interpolation techniques differently. As a result, these numerical algorithms could produce extremely distinct computer-generated trajectories of chaotic systems for a very long time spans (Liao et al., 2021). A widely used numerical integrator is Runge-Kutta, which is found to be stable even using a fixed time step in integration and for long time intervals (Křižek, 1995). But, due to its limitation in double-precision arithmetic, methods such as Clean Numerical Simulation (CNS) have been introduced to minimize not only truncation errors but also round-off errors (Li & Liao, 2018). Some integrators focus solely on achieving extreme high precision such as Brutus (Boekholt & Zwart, 2014), which calculates the average of the results of multiple integrators. Yet, this approach is extremely computational resulting in making long-term predictions impractical.

3.2. Neural Networks Methods

A case study examined the use of a Deep Neural Network (DNNs) in approximating the three-body problem (Breen et al., 2020). The problem was simplified by setting a fixed time interval of $t \leq 10$ time unit broken into 256 steps, and assigning all bodies to have the same mass and zero initial velocity. A DNN is a type of ANN which contains additional layers to handle more complex problems. The DNN used here consisted of 10 hidden layers with 128 interconnected nodes each. it was trained using the ADAM optimizer over 10,000 batches, each containing 5,000 data points. The results demonstrated that the DNN could achieve similar accuracy to the Brutus integrator while being up to 100 million times faster.

Under the same training parameters, a second case study built upon the DNN work by implementing a PINN to approximate the three-body problem (Pereira et al., 2025). This case study added a ResNet architecture, which enhances gradient flow via residual connections between layers, decreasing the "exploding gradients problem" and stabilizing training. The PINN with the ResNet architecture was more accurate and had lower computational requirements compared to conventional DNNs

There has also been work on Hamiltonian Neural Networks (HNNs) that is quite similar to PINNs. However, HNNs are explicitly designed to include Hamiltonian mechanics, which represent the total energy of the system (Greydanus et al., 2019). HNNs were found to be good at preserving scalar quantities, i.e., the total energy for various systems such as springs and pendulums. It outperformed traditional neural networks in doing this, however they could not accurately model the complex dynamics of the three-body problem.

4. Methods

This section describes the methods used to conduct our experiments. The code used for the experiments is available on GitHub using the following link: `https://github.com/Angelmartinez-20/PINN_3BodyProblem`

4.1. Training Data

Since the three-body problem lacks an analytical solution, the PINN must be trained on data generated by a numerical integrator. Runge-Kutta method of order 5(4) was used due to its stability, as CNS integration remains theoretical with limited practical applications. A total of 7,500 simulations were generated, each calculating the trajectories of the three

bodies over $t = 50$ with a time step of $\Delta t = 0.01$ (5,000 steps).

To improve on prior research and simulate general three-body dynamics, the initial masses, positions, and velocities were randomized within specified ranges. The masses of the bodies were set between $[1.0, 5.0]$, and the x,y,z velocities ranged from $[-0.5, 0.5]$. The position of $\mathbf{p}_1$ was set at the origin $[0.0, 0.0, 0.0]$, while the x,y,z coordinates of $\mathbf{p}_2$ and $\mathbf{p}_3$ were set within $[-10, 10]$. A minimum distance of 2.5 units between the bodies was also ensured.

4.2. PINN Architecture

The PINN design follows the neural network architecture from the prior study that applied a PINN to a simplified three-body problem (Pereira et al., 2025). Its architecture consists of 10 hidden layers with 128 interconnected nodes, using the ReLU activation function and an Adam optimizer. Additionally, gradient clipping was implemented to mitigate the exploding gradient problem (Philipp et al., 2018).

The input layer has 22 nodes which captures the bodies masses, x,y,z position values, x,y,z velocity values, and lastly the timestamp $(3 + 9 + 9 + 1)$. Then the output of the PINN will result in the final position and velocities $(9 + 9)$ of the bodies at its input timestamp t.

4.3. Validating Results

Validating and comparing the accuracy of the numerical integrator and the PINN is challenging, as both are approximators without an exact solution to compare against. Thus, the Lagrangian formalism equation (2) is used to assess the extent to which the system's energy changes over time. The principle of energy conservation explains how the total energy of interacting bodies in a closed system remains constant. If an analytical solution were known, its final positions and velocities would hold an energy value identical to its initial position and velocity values. As a result, the greater the difference in the system's energy, the less accurate the numerical integrator or PINN is in predicting the true trajectories of the three bodies.

The conducted experiments include comparing the PINN against a set of testing data and evaluating its performance against the numerical integrator for trajectories beyond the training range. Of the 7,500 simulations built, 6,000 are used for training, and 1,500 are used for testing. This experiment highlights why a PINN can be more useful than a numerical integrator by determining the trajectories of the three bodies for a new problem without having to integrate over the entire series of time steps again. Finally, the PINN is evaluated for its ability to predict the trajectories of bodies exceeding the training time range of $t = 50$.

In addition to accuracy comparisons, a runtime experiment is conducted to compare the trained PINN's performance against simulations with larger timesteps. This comparison quantifies the efficiency gains of trained PINNs for trajectory prediction compared toward numerical integrators for each simulation.

5. Experiments

Before attempting to train a PINN model on the general three-body problem. We recreate the simplified three-body problem from subsection 3.2. We also included the code for the simplified three-body problem experiment in our GitHub as the previous authors did not open source their experiment. The PINN architecture remains the same as explained in subsection 4.2, but there are only 7 input nodes and 12 outputs given that the problem is in 2-dimensions and there is no initial velocity. They initially trained there model with 30,000 trajectories, but as a proof of concept, we trained ours over 8,000 trajectories and used 2,000 for testing which was compared to the RK45 integrator instead of BRUTUS.

5.1. Simplified Three-Body Problem

An important hyperparameter when training a PINN is a λ_f value which determines how strong do you want the weight of the physics residual (MSE_f) to impact the loss function. Given this option, we trained several different PINN models with different λ_f weights. As proposed in the previous PINN Simple Three-Body case study (Pereira et al., 2025), we also implemented "warm up" models which applied the physics residual after the model had the chance to train after 80 epoch as a regular DNN model. This showed positive results for the previous case study in building a more accurate PINN. Table 1 shows the average MSE value across all the training trajectories during its final epoch.

Table 1. PINNs Simple 3-Body Training Final Epoch MSE

λ_f	No Warmup	Warmup
10.0	1.146	1.103
1.0	0.931	0.920
0.1	0.776	0.809
0.01	0.684	0.640
0.001	0.629	0.612
0.001 - 0.75	0.880	0.835
0.0	**0.589**	**0.589**

Given our training performance results, we observed that applying a physics residual of any weights resulted in a less accurate model than without any physics (regular DNN model). This was an interesting observation; however, the models still needed to be tested to confirm. The "warm up" models did learn more accurately than without which does

align with the previous case study observation.

When testing each model, we took into account the MSE value over the entire trajectory, the final MSE value for the final prediction, and the Energy drift given the conditions of the bodies at the last time step compared to the first. These values where calculated for all 2,000 testing trajectories and then averaged. Table 2 includes the models without any "warm ups" before apply MSE_f to the training while Table 3 does include the "warm up"

Table 2. PINNs Simple 3-Body Testing Results (No Warmup)

λ_f	MSE	MSE ($t = 10$)	E. Drift
10.0	2.823	6.231	1.714
1.0	2.332	5.599	1.239
0.1	2.053	5.509	**0.538**
0.01	1.841	5.216	0.658
0.001	1.760	4.962	2.073
0.001 - 0.75	2.147	5.553	1.195
0.0	**1.702**	**4.929**	11.256

Table 3. PINNs Simple 3-Body Testing Results (Warmup)

λ_f	MSE	MSE ($t = 10$)	E. Drift
10.0	3.090	6.796	1.798
1.0	2.194	5.569	1.440
0.1	2.048	5.474	**0.569**
0.01	1.701	4.818	0.754
0.001	**1.675**	**4.764**	2.072
0.001 - 0.75	2.083	5.411	1.137
0.0	1.702	4.929	11.256

After testing our models, the "warm up" PINN with $\lambda_f = 0.001$ ended up being the best performer with respect to MSE while "no warm up" PINN with $\lambda_f = 0.1$ had the least energy difference. Given that our PINNs preformed better during testing than training, we concluded that the physics residual made the model learn the geneal dynamics better, rather than over-fitting on the training trajectories. Figure 2 is a visual comparison of how the PINN trajectories compare to the numeral integrator trajectories.

As a result, the different models capture some general structure but lacks the sufficient detail to be able to be use reliably. An interesting observation was despite "no warm up" PINN with $\lambda_f = 0.1$ being the better model with respect to energy drift, its trajectories are the most different compared to the numeric integrator. This suggests that energy drift alone may not fully explain how far the final positions of the three bodies are from the ground truth. Instead, it may reflect whether the predicted final positions and velocities are physically consistent with the initial conditions. In this view, the model appears to prioritize generating a trajectory that is physically possible to exists. Thus, the energy difference at the final time step may indicate the plausibility of the predicted state at any point in time, rather than its accuracy at the final step specifically.

5.2. General Three-Body Problem

Moving onto our contribution with applying a PINN model to a more general three-body problem, we followed a similar training and testing experiment to that of the simple three-body problem with different λ_f values. Table 4 shows are average MSE values at the last epoch step for our different models.

Table 4. PINNs General 3-Body Training Final MSE

λ_f	No Warmup	Warmup
10.0	7.174	8.648
0.1	6.637	6.658
0.001	5.896	5.335
0.0	**4.022**	**4.022**

Just like with the simple three-body problem, our PINN models preform less accurate than without a physics component (DNN). We can infer that this due to the PINN models learning more of the physical properties rather than the training data points. Also another immediate observation is that the MSE values are higher than before which indicated the models are strangling to learn the complexity of a more general three-body dynamic. For testing, Table 5 is are "no warm up" models while Table 6 includes a "warm up".

Table 5. PINNs General 3-Body Testing Results (No Warmup)

λ_f	MSE	MSE ($t = 50$)	E. Drift
10.0	9.397	24.360	**1.060**
0.1	**9.351**	**24.488**	2.330
0.001	9.367	24.757	2.534
0.0	10.418	27.705	1.392

Table 6. PINNs General 3-Body Testing Results (Warmup)

λ_f	MSE	MSE ($t = 50$)	E. Drift
10.0	10.683	26.624	1.812
0.1	10.114	26.323	2.382
0.001	**9.508**	**25.478**	2.479
0.0	10.418	27.705	**1.392**

Unlike the simple three body problem, most of my PINN models outperform the regular DNN model without any physics. This goes to show as the problem becomes more apparent to its true physical nature, the physics residual in a PINN allows the model to better capture its properties. A

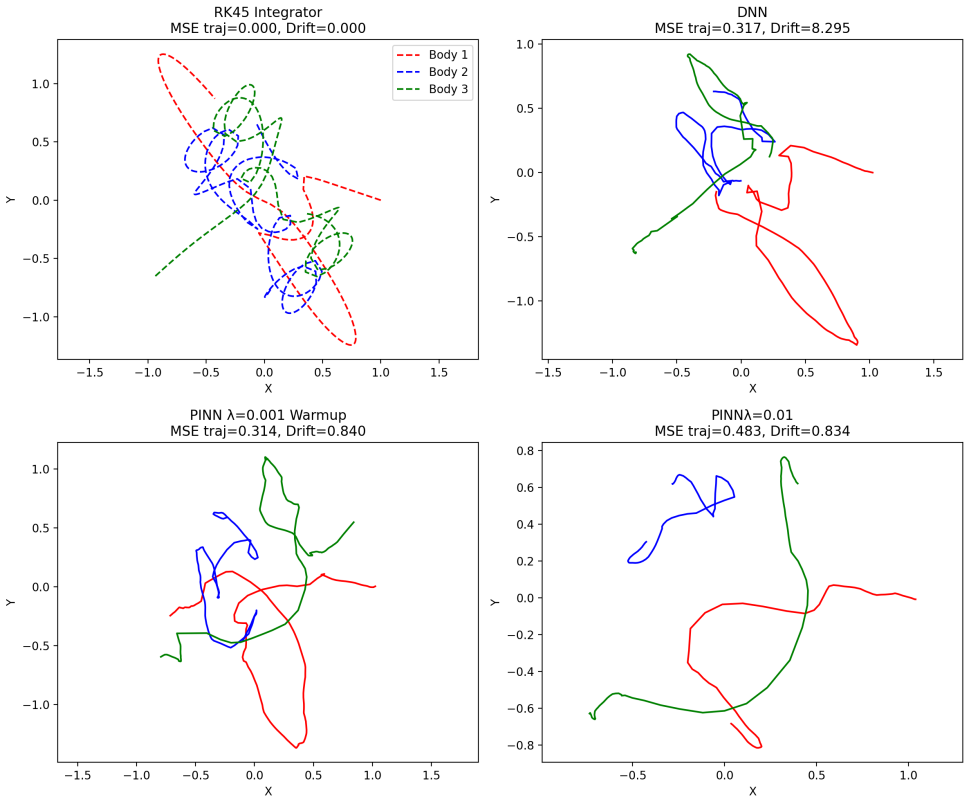

Figure 2. Comparison of the Simple 3-body trajectories: top-left is the RK45 numerical solution; top-right is the PINN without physics loss (DNN); bottom-left is $\lambda_f = 0.001$ warmup (best MSE model); bottom-right is $\lambda_f = 0.1$ (best energy drift model). Each subplot reports the mean-trajectory MSE and energy drift.

noticeable change in results was that our "warm up" models preformed less accurately. We infer that as the model is now trying to capture a more general dynamic, it needs the physics residual to steer it in the right direction in the beginning rather than just relying on data points until epoch 80.

With the best performing MSE model being $\lambda_f = 0.1$, and best energy drift model being $\lambda_f = 10$ with both "no warm up", Figure 3 is a visual comparison of their trajectory differences. In this example, $\lambda_f = 0.1$ model shows how its better predicting the loop the 3rd body exhibits compared to the other models.

Although all PINN models where trained to predict the trajectories out to $t = 50$, we also conducted an overflow analysis to test how our best preforming PINN model will compare against capturing the trajectory over $t = 250$. Figure 4 display the resulting figure comparing the numerical integrator trajectory with $\lambda_f = 0.1$ model. This corresponds to Figure 5 and 6 as these are the MSE and Energy drift over each time step.

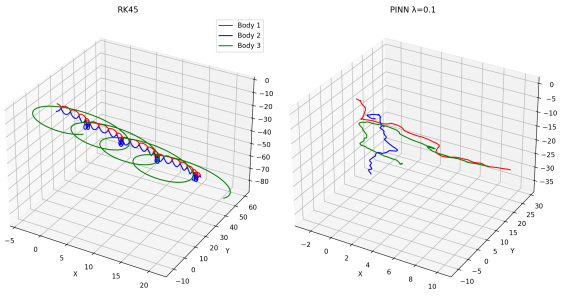

Figure 4. Trajectory comparison $t = 250$

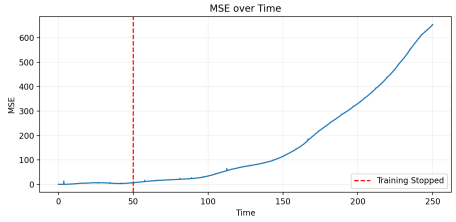

Figure 5. MSE over time

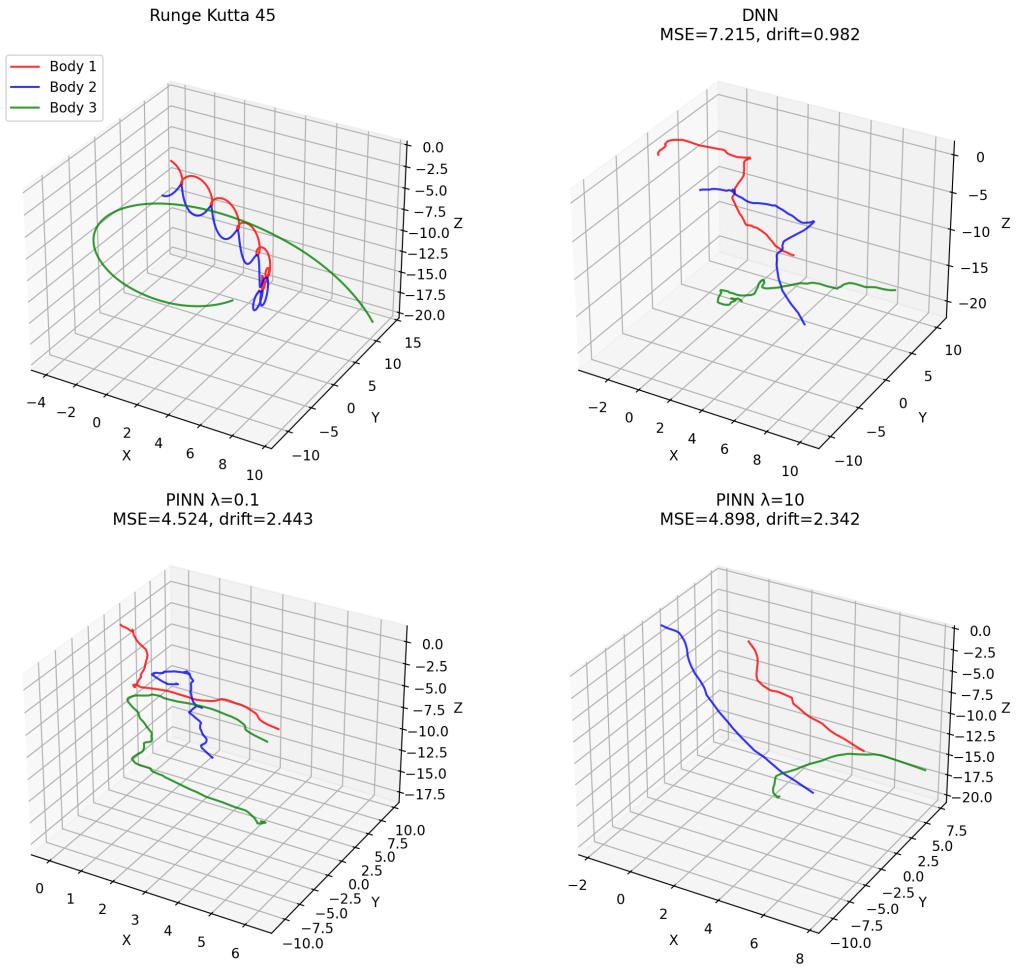

Figure 3. Comparison of the general 3-body trajectories: top-left is the RK45 numerical solution; top-right is the PINN without physics loss (DNN); bottom-left is $\lambda_f = 0.1$; bottom-right is $\lambda_f = 10$. Each subplot reports the mean-trajectory MSE and energy drift.

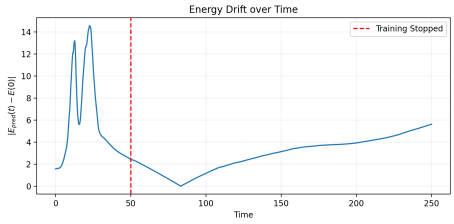

Figure 6. Energy Drift over time

6. Conclusion

As a result, a PINN was able to capture certain general structures of the general three body trajectories. Despite training over 6,000 different trajectories, it does require more extensive training in order to become more accurate and usable. It's MSE and Energy drift do stay stable after a small time outside the scope of its training, but then it increases rapidly as it looses its ability to understand how the trajectories will continue to evolve. The physics component inside a PINN does make it a more applicable machine learning method than a standard DNN model which can be used for other physical systems that involve PDEs. Given the results, we conclude that numerical integrators are still the more reliable method to use for chaotic systems as training a well defined model to learn how trajectories can evolved is outside our computation resources. Especially if the model will be used to predict outwards of days or months compared to just 50 seconds.

7. Future Work

Other than creating more training data for our models to learn from, future work could include using different hyper parameters for our PINN models other than λ_f. The previous case study had different performance results with their differently tuned PINN models and their ResNet architecture

showed more accurate results (Pereira et al., 2025). Also despite our general model being able to be used for real-world scenarios, its not that practical to model the trajectories of 3 freely close and similar massed bodies. Instead, we can train a PINN model to determine the trajectories of a space craft given the two other bodies being the earth and moon. This creates a much different problem than our general model as the bodies distances are thousands of miles away and the masses of 2 bodies are drastically much more than that of a spacecraft. However, if a PINN is able to learn a more practical three body dynamic, then this can be a step forward in machine learning models being used for at least quickly sketching the dynamics of a chaotic system.

References

Boekholt, T. and Zwart, S. P. On the reliability of n-body simulations. 2014.

Breen, P. G., Foley, C. N., Boekholt, T., and Zwart, S. P. Newton versus the machine: solving the chaotic three-body problem using deep neural networks. *Monthly Notices of the Royal Astronomical Society*, 2020. doi: 10.1093/mnras/staa713.

Broucke, R. and Lass, H. A note on relative motion in the general three-body problem. *Celestial Mechanics*, 8(1): 5–10, 1973.

Greydanus, S., Dzamba, M., and Yosinski, J. Hamiltonian neural networks. 2019.

Křižek, M. Numerical experience with the three-body problem. *Journal of Computational and Applied Mathematics*, 63, 1995.

Li, X. and Liao, S. Clean numerical simulation: A new strategy to obtain reliable solutions of chaotic dynamic systems. 2018.

Liao, S., Li, X., and Yang, Y. Three-body problem - from newton to supercomputer plus machine learning. 2021. doi: 10.21203/rs.3.rs-395522/v1.

Newton, I. *Newton's Philosophiae Naturalis Principia Mathematica*. 1687.

Pereira, M. S., Tripa, L., Lima, N., Caldas, F., and Soares, C. Advancing solutions for the three-body problem through physics-informed neural networks. 2025.

Philipp, G., Song, D., and Carbonell, J. G. The exploding gradient problem demystified - definition, prevalence, impact, origin, tradeoffs, and solutions. 2018.

Pitts, J. Conservation of energy: Missing features in its nature and justification and why they matter. *Foundations of Science*, 26(3), 2021. doi: 10.1007/s10699-020-09657-1.

Poincaré, H. *Les Méthodes Nouvelles De La Mécanique Céleste*. Gauthier-Villars etfils, imprimeurs-libraires, 1892.

Raissi, M., Perdikaris, P., and Karniadakis, G. E. Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations, 2017.

Soak, O. Training anns: A deep dive into backpropagation and gradient descent, 2024.

Sundman, K. F. *Mémoire Sur Le Probléme Des Trois Corps*. 1913.

Valtonen, M., Anosova, J., Kholshevnikov, K., Mylläri, A., Orlov, V., and Tanikawa, K. *The Three-body Problem from Pythagoras to Hawking*. Springer International Publishing, 2016.

Walczak, S. and Cerpa, N. Artificial neural networks. *Encyclopedia of Physical Science and Technology*, 2003. doi: 10.1016/B0-12-227410-5/00837-1.