

Applying physics-based loss functions to neural networks for improved generalizability in mechanics problems

Samuel J. Raymond^{a,*}, David B. Camarillo^{a,b,c}

^a*Department of Bioengineering, Stanford University, Stanford, CA, 94305, USA*

^b*Department of Neurosurgery, Stanford University, Stanford, CA, 94305, USA.*

^c*Department of Mechanical Engineering, Stanford University, Stanford, CA, 94305, USA*

Abstract

Physics-Informed Machine Learning (PIML) has gained momentum in the last 5 years with scientists and researchers aiming to utilize the benefits afforded by advances in machine learning, particularly in deep learning. With large scientific data sets with rich spatio-temporal data and high-performance computing providing large amounts of data to be inferred and interpreted, the task of PIML is to ensure that these predictions, categorizations, and inferences are enforced by, and conform to the limits imposed by physical laws. In this work a new approach to utilizing PIML is discussed that deals with the use of physics-based loss functions. While typical usage of physical equations in the loss function requires complex layers of derivatives and other functions to ensure that the known governing equation is satisfied, here we show that a similar level of enforcement can be found by implementing more simpler loss functions on specific kinds of output data. The generalizability that this approach affords is shown using examples of simple mechanical models that can be thought of as sufficiently simplified surrogate models for a wide class of problems.

1. Introduction

Physics-Informed Machine Learning (PIML) is a cutting-edge new field that sits at the intersection of scientific computing and machine learning. The field is only a few years old now but has already begun producing some valuable insights into the combined approaches of these two domains, particularly in the intersection of computational mechanics, modeling real-world materials/fields, and deep learning, using advanced neural network architectures.

During the immense rise to power of Artificial Intelligence and Machine Learning in the last two decades, scientific computing was not a largely looked at field of application. Web traffic[1], customer habits [2], Geo-spatial information [3], medical imaging [4], and many others were far easier to apply techniques such as deep learning to, as the guiding models for these areas are often too complex to develop from first principles, and the amount of data available made the training and testing very attractive. Also, as a fundamentally probabilistic endeavor, deep learning may infer values that might violate or exceed reasonable bounds [5], which would violate the Laws used in numerical physics, for example, rendering it fundamentally flawed as an application. Scientific data and numerical models, however, are not perfect, and errors and systematic biases are present in these approaches as well. So, if the bounds of expected errors that were to come from a learned-model were similar to those present in a scientific model, an argument could be made to rely just as much on the data-driven approach. In numerical modeling, the starting point is almost always the discretization of the governing partial differential equation (PDE), into a form that can be converted into the syntax of a computer program. On its own, the PDE is in some sense useless as a tool for modeling, until specific boundary and initial values are used, and appropriate parameters set. This transforms the universally

*Corresponding author

Email address: sjray@stanford.edu (Samuel J. Raymond)

applicable PDE to a specific model of a specific use case. In a corollary manner, a deep neural network can be thought of as a universal function approximator [6], where, with enough connections and neurons, any equation or transfer function relating two variables is theoretically constructable. However, without data and training, the neural network, like the PDE is also completely useless and offers no context-relevant inference. Only once the data has been fed to the neural network and the weights updated through the training process is the general function approximator converted to something far more brittle, yet useful. Brittle here refers to the extent with which the network can make accurate inferences. Only within the bounds of the data used to train the model can a network be relied upon to provide any useful information. Even within these bounds though, there has already been a number of useful applications of deep learning in the physical sciences [7, 8, 9, 10, 11, 12, 13, 14] and this manner of application will continue to develop as more data is generated for scientific purposes. Relying on a data-driven model to provide mechanistic predictions though, creates a number of problems. Adversarial neural network research [15] has shown just how brittle these models can be even with human-imperceptible changes to an input. One advantage of numerical models is that they are far less opaque to their brittleness and generally do far better at coping with a large range of input parameter variations without needed to remake the model, though there is certainly a limit to all models. A dangerous prospect, therefore, presents itself when relying on traditional neural networks to provide scientifically relevant inferences.

The goals of PIML are to address these shortcomings of traditional neural networks, and to leverage the speed and fusion of data that using neural networks affords. Computational models are commonplace in essentially every area of human endeavor, with new methods and techniques constantly introduced to manage increasingly complex scenarios [16, 17, 18, 19, 20, 21, 22]. However, as these models become more sophisticated, they also become horrendously expensive on the scales of climate modeling[23] or large-scale molecular dynamics models [24], and with ever-larger supercomputers requiring even more energy to run[25], PIML also offers a more energy-efficient and sustainable approach to infuse these networks with the benefits of pre-existing domain knowledge. In the last few years, a number of distinct fields of PIML have started to emerge, each using a different approach to embed domain knowledge into deep learning frameworks. These include Physics-Informed Neural Networks (PINNs)[26], synthetic data[7], and data-driven equation learning [27].

Of these fields this work is primarily interested in the development and use of PINNs [28, 29, 30, 31, 32, 33, 34, 26, 35, 36, 37, 38, 39]. The approach in PINNs is to use as an input-output pair to the neural network a fundamental variable, or set of variables for the physical problem, and use these variables to construct the quantities that appear in the governing equation of motion, such as the velocity and pressure fields in a fluids problem. The equation of motion, for instance the Euler or Navier-Stokes equation, is then used as an additional term in the loss function of the deep learning architecture using an additional layer after the output neurons to convert the variables into the forms needed for the equation of motion. The seminal work on this was conducted by Raissi et al. [26] and the interested reader is encouraged to read their work. The new loss function of PINNs then acts in a similar manner to the residuals in a Finite Element scheme, and the training is performed using traditional backpropagation methods until the loss associated with both the L2-norm and the equation of motion residual is minimized. More work though, is required to understand the right trade-off between these two errors and if one should be weighted more than the other.

The loss function in PINNs, as in all deep learning, plays a crucial role in the ability of the network to train well [40]. The approach of PINNs is to leverage the full understanding of the governing physics at play to build a basis function, in the form of a neural network, that solves these equations. However, there are many situations where either the governing equation is not clear, or the inclusion of the full PDE introduces such a computational overhead that any efficiencies afforded by the use of a PINN are negated. It may also be that the entire PDE is not required for a sufficiently accurate solution to be produced, for the tolerated error. For example, in viscous flows or simple fluid problems, the full 3D Navier-Stokes is not required and training with such a complex PDE may be unnecessary. In this work we introduce a new approach to this problem by utilizing the Laws of physics in the loss functions, ones that tend to sit above governing PDEs but can be applied much more efficiently and liberally to a number of different problems. To present this, the paper is structured as follows: first the observation of an automatic satisfaction of the underlying Principle of Least Action is shown to arise naturally in backpropagation of particular models, then a number

of experiments to reproduce phase-space trajectories with minimized Action are conducted. Next a simpler loss-function approach is presented that uses the conservation of energy as the guiding principle and this is used to build a neural network to predict the motion of a pendulum. This is compared with the traditional, data-only approach, and the discussion of the comparison concludes the work.

2. Methods

2.1. Least-Action and The Conventional Loss Function

Suppose there exists some system that can be modeled with a Lagrangian, \mathcal{L} . The equations of motion that govern this system can be found by invoking the Principle of Least (or stationary) Action [41]. This principle states that the action, which is defined as the time integral of the Lagrangian from one state to another, is minimized by the trajectory of the system as it moves through phase-space. The equations that come from invoking this principle are known as the Euler-Lagrange equations [42], and are essentially a reformulation of Newton's Laws of motion. Mathematically, this is described as follows.

Some Lagrangian, \mathcal{L} is proposed for a system, incorporating the kinetic energy and any potential energies which are formed as functions of the generalized positions, q , and velocities, \dot{q} of the particles.

$$\mathcal{L} = f(q, \dot{q}) = T - V \quad (1)$$

For a system of particles, the kinetic energy, T_{sys} , and potential energy, V_{sys} , can be formed by combining the contributions from all the particles within the system.

$$T_{sys} = \sum_{i=0}^N \frac{1}{2} m_i \dot{q}_i \cdot \dot{q}_i \quad V_{sys} = \sum_{i=0}^N f(q_i, \dot{q}_i) \quad (2)$$

The action is then formed as an integral between two points in time.

$$A = \int_{t_1}^{t_2} \mathcal{L}(q, \dot{q}) \quad (3)$$

To find the trajectory that minimizes the action, the calculus of variations is invoked, leading to the Euler-Lagrange equations.

$$\delta A = \delta \int_{t_1}^{t_2} \mathcal{L}(q, \dot{q}) = 0 \quad (4)$$

Any space-time trajectory that is given by the Euler-Lagrange equations, therefore, satisfies the principle of least action for this system.

2.1.1. Minimizing Action via Back Propagation

Remarkably, an approximation to this procedure is also found in the training of a neural network if the inputs to the system are the initial conditions of the system and the outputs are the full, space-time, trajectories of the system to another point in time. When this is combined with the mean-squared error (a very commonly used measurement for error in deep learning) an approximation to the calculus of variations is produced.

$$\text{mean squared error (mse)} : \sum_{\forall k} \sqrt{z_k^2 - y_k} \quad (5)$$

$$z \equiv f(q_i, \dot{q}_i) \quad (6)$$

$$\text{mse} = \sum_{\forall k} \sqrt{z_k^2 - y_k} = \sum_{\forall k} \mathcal{L}' - \mathcal{L}_O = \sum \delta \mathcal{L} \cong \delta A \Rightarrow 0 \quad (7)$$

Because this summation is over the time step outputs of the system, this approximates the integral used in the analytical version of the principle, and as the optimizer converges, a space-time trajectory is found that

most closely resembles that which would satisfy the Euler-Lagrange equations. This result implies that it is in fact very straightforward to use a neural net to predict the motion and behavior of a system. For different systems with a different governing Lagrangian, it may be possible to find precise error estimates that produce the required conservation properties (energy, momentum, angular momentum) needed to simulate a system's evolution. The role of back propagation is to use the given error term as a means to direct the optimizer such that the total error of the predicted versus given result is as small as possible. It is this optimization process that forces the output trajectory to be as similar to the Least Action-obeying trajectory as possible that produces this alternative approach to learning the laws of physics within the network. This requires the variables to span the phase-space, but in many mechanics problems, the key variables of position and moment are used and are sufficient to span this space.

To test this, two different mechanical systems were simulated to generate data, and then a neural network was trained to learn the underlying structures so that for an initial starting position in phase space (position and momentum), the full space-time trajectory of the system would be output by the network. The first focused on the motion of a projectile under the influence of gravity, both with and without the effect of drag. The second investigated the modeling of a pendulum, both a single pendulum and a double pendulum.

2.1.2. Simple Projectile Motion

Projectile motion is one of the simplest forms of mechanics problems that is often covered in the first physics-stream classes in high schools. This problem is useful as it describes the motion of objects in a wide range of fields (many common phenomena incorporate projectile motion such as fountains, sports, etc.) and is the simplest form of motion, after constant velocity motion. The basis for projectile motion is that the motion is governed by a constant acceleration (notably gravity), that acts in one direction. This simplest form of projectile motion completely decouples the motion in the horizontal and vertical directions. The motion in the horizontal direction ($x(t)$) and the vertical direction ($y(t)$) can be described simply as:

$$x(t + dt) = x(t) + u(t) dt \quad (8)$$

$$y(t + dt) = y(t) + v(t) dt - \frac{1}{2} g dt^2 \quad (9)$$

Where $u(t)$ and $v(t)$ are the x and y velocities, respectively, and g is the acceleration due to gravity (assumed to be constant, and acting only in the y direction). To generate the data for the neural network, a simple, forward Euler integrator was used to generate the (x, y) positions of the projectile from a starting position with a given starting velocity. A range of values for the gravity and time of flight were used as well as a range of starting x and y velocities. The Deep Learning Toolbox in MATLAB R2020a was used to train a fully connected feedforward network. The neural net consisted of an input layer with 6 neurons, 2-hidden layers with 15 neurons per layer, each with with ReLU activation functions, and an output layer consisting of the 4x200 time step values of the resulting motion of position and velocities in the 2 directions. Training was performed using the mean-squared error performance criterion and the Levenberg-Marquardt algorithm was used for back propagation. The neural net was given the starting position, velocity, gravity strength, and time of flight, as input. The output was set as the (x, y) positions and velocities of the projectiles over the time of flight.

2.1.3. Nonlinear Projectile Motion

To incorporate more realistic effects, non-linear drag was also added to the equations of motion such that the new equations of motion coupled the x and y directions together.

$$x(t + dt) = x(t) + u(t) dt - C_D \frac{v(t)^2}{V(t)} dt \quad (10)$$

$$y(t + dt) = y(t) + v(t) dt - \left(\frac{1}{2} g dt^2\right) - C_D \frac{v(t)^2}{V(t)} dt \quad (11)$$

Here, C_D is the coefficient of drag and $V(t)$ is the absolute velocity of the projectile. The same neural network architecture and training procedure was used for this second case.

2.1.4. Single Pendulum Motion

Periodic motion is another class of commonly occurring motion that scientists and engineering encounter. To determine the efficacy of this workflow on the prediction of periodic motion, again a similar, feedforward network was created to learn the motion of a pendulum. While it may seem that a recurrent network may seem more appropriate for period motion learning but with the final desire to be able to model the chaotic motion of double pendulums, where periodicity is not assumed, the simple deep, fully connected layers were again used. For the single pendulum model, the generated data for the neural net came again from a simple, forward Euler integrator. This was used to produce the angle and angular velocity trajectory for a mass from an initial starting angle and starting angular velocity. The length and mass of the pendulum were set to equal 1.0 for all models in this work.

2.1.5. Double Pendulum Motion

Chaotic motion is described by a solution that is heavily dependent on the initial conditions of a dynamic system. The typical double pendulum is one such system that exhibits chaotic motion. Modeling this kind of system is challenging for even traditional methods when they follow the constraints they are required to follow. The two masses were given different initial starting angles and angular velocities and were allowed to oscillate for two full periods, in the same manner as the single pendulum. Here also is the reason for choosing not to use a recurrent network model. The results of these two pendulum models are shown in the next section as the neural network attempted to predict the full phase-space trajectory of the pendulum masses from the initial conditions alone.

2.2. A custom loss function for robust time step predictions

In the previous section, a full time-space trajectory was predicted from initial conditions due to the ability of the neural network to minimize the correct trajectories and satisfy the principle of least action. However, since this method relies on having a full phase-space solution, this is not a simple method to scale to more complex problems. Instead, now we present an approach that only learns the evolution of a system by one time step, but ensuring that energy is conserved from one point in time to the next. Here we show that the traditional loss function, when exposed to several systems of different energy levels, will tend to produce the lower energy solution as this better satisfies the minimization process. By adding the extra constraint of the conservation of energy, better generalizability will be introduced to the network. To prepare this workflow, a pendulum is simulated and the initial conditions are set at: $[\theta_0 = \frac{\pi}{4}, \dot{\theta}_0 = \frac{\pi}{6}, \theta_0 = \frac{\pi}{8}]$ with the angular velocity, $\dot{\theta}_0 = 0$ in each case. The neural network was designed with two input and output neurons, representing angular position and angular velocity at times t and $t + 1$, respectively. The pendulum was simulated for 6 full cycles and the pairs of t and $t + 1$ values were randomly mixed and used to train the network. Two loss functions were used to train the network. The standard loss function $L_{standard}$ was set as the mean-squared error, and the second physics-loss function, $L_{phys} = L_{standard} + \Delta E$, where ΔE , the change in energy was taken as:

$$\Delta E = [\frac{1}{2}m\dot{\theta}_T^2 + mg(1 - \cos(\theta_T))] - [\frac{1}{2}m\dot{\theta}_Y^2 + mg(1 - \cos(\theta_Y))] \quad (12)$$

where the T and Y subscripts refer to the true and network-predicted values, respectively. This custom loss function was added into the training processes via MATLAB's Deep Learning Toolbox. To determine the effect of different loss functions on the prediction of the network, starting positions of $[\theta_0 = \frac{\pi}{2}, \theta_0 = \frac{\pi}{3}, \theta_0 = \frac{\pi}{6}]$ were used and the output from each step was used as the input for the next until a solution over a number of periods was produced. The results of this approach are shown in the next section.

3. Results

3.1. Trajectories with least action

To compare the neural net trajectories with that solved from by the simulator, sample input values were given to both the simulation engine and the neural net. Comparisons are shown for both the projectile and pendulum motions with both linear and nonlinear features.

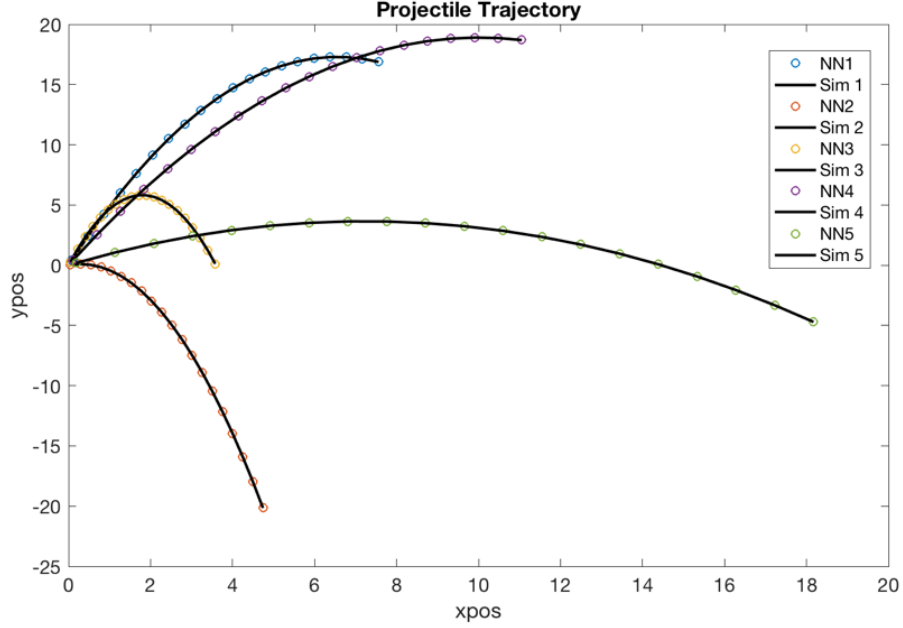


Figure 1: Simple projectile motion showing the spatial trajectory of the point through 2D space. Black lines indicate the numerical simulation results and the circles are the outputs space-time trajectories from the neural network.

3.1.1. Projectile Motion

The results, shown in Figure 1 indicate excellent agreement with the trajectories from both numerical procedures. The results are shown in Figure 2 with the difference being that the coefficient of drag is now an added input variable. The neural net is again able to successfully learn the relationships in the data provided between the input and output and the trajectories lie on top of the simulated values.

These results indicate that the network is able to take only a few input variables (initial velocity, position, drag terms) and produce a physically accurate trajectory of the motion of the projectile.

3.1.2. Pendulum Motion

The results are shown in Figure 3 and again we can see that the network is able to follow the motion through several periods of oscillation, again indicating that the underlying physics are being correctly inferred from the training. As Figure 4 shows, this neural network approach is also capable of following complex, multi body motion in chaotic systems to a surprising degree of accuracy. The path in phase space shows very little periodicity and the feedforward fully connected layers are able to learn the trajectories of the pendulum and predict the masses paths through phase space with very good accuracy.

3.2. Predicting Motion with a Physics-based loss function

To compare the efficacy of standard loss functions and PIML-motivated loss functions, the prediction of a simple pendulum's motion is shown for both the conventional and physics-based loss functions. Different starting positions and the resulting predicted motions are shown for three different starting angles.

3.2.1. Conventional Loss Function

Figures 5 - 7 show the results of the prediction of the trajectory of the pendulum from the different starting positions. While the initial predictions follow the phase-space curves for some time, they quickly decay to the smallest energy phase-space.

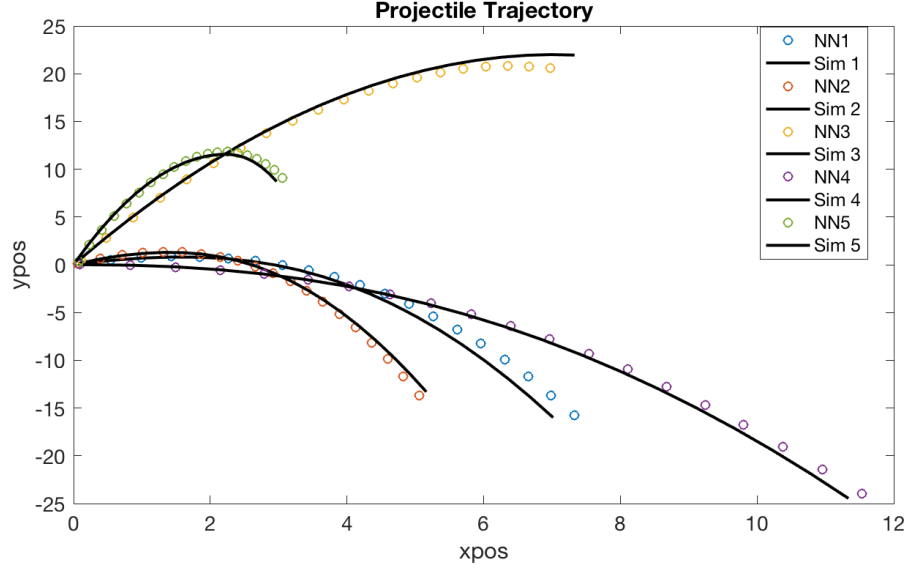


Figure 2: Nonlinear projectile motion showing the spatial trajectory of the point through 2D space. Black lines indicate the numerical simulation results and the circles are the outputs space-time trajectories from the neural network.

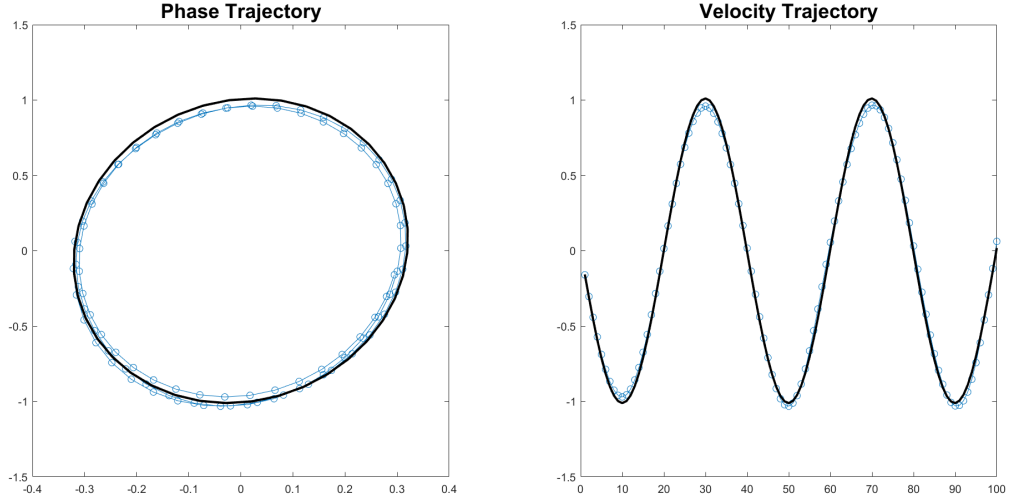


Figure 3: (Left) Phase space trajectory of the motion (Right) angular velocity. Black lines indicate the numerical simulation results and the circles are the outputs space-time trajectories from the neural network.

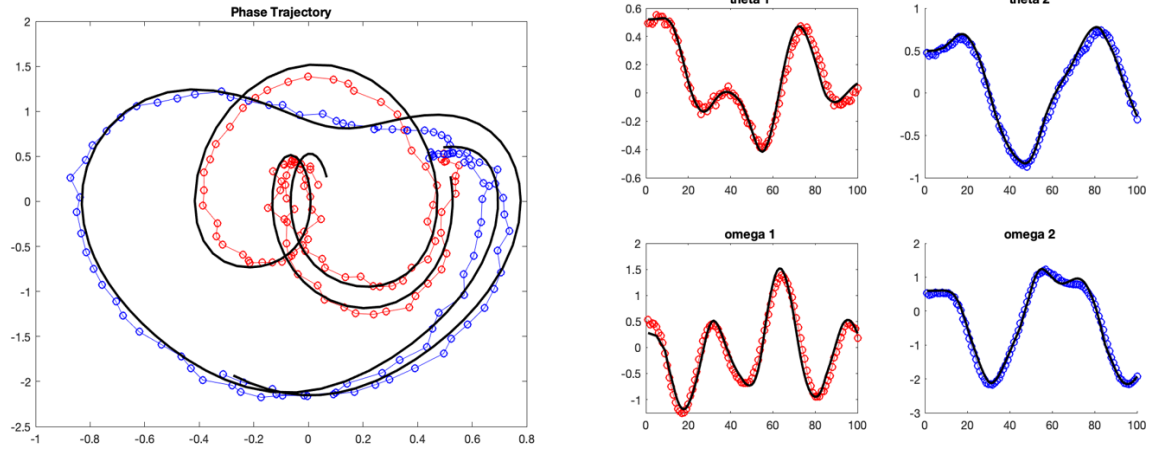


Figure 4: Motion of the double pendulum. (Left) Phase space trajectory of the motion of the top (blue) and lower (red) masses. (Right) angular velocities and angular positions of the upper (blue) and lower (red) masses. Black lines indicate the numerical simulation results and the circles are the outputs space-time trajectories from the neural network.

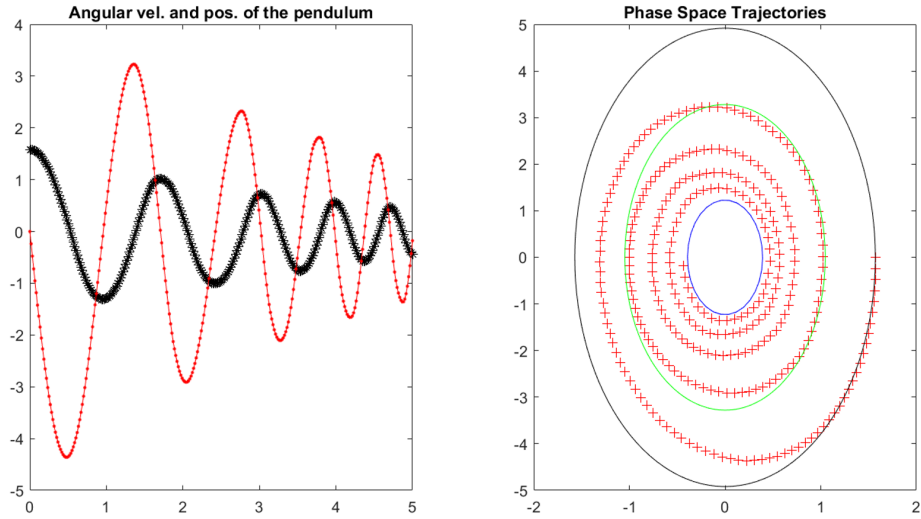


Figure 5: Prediction of the motion of a pendulum from the starting position of $\frac{\pi}{2}$ using the conventional loss function over 4 periods.

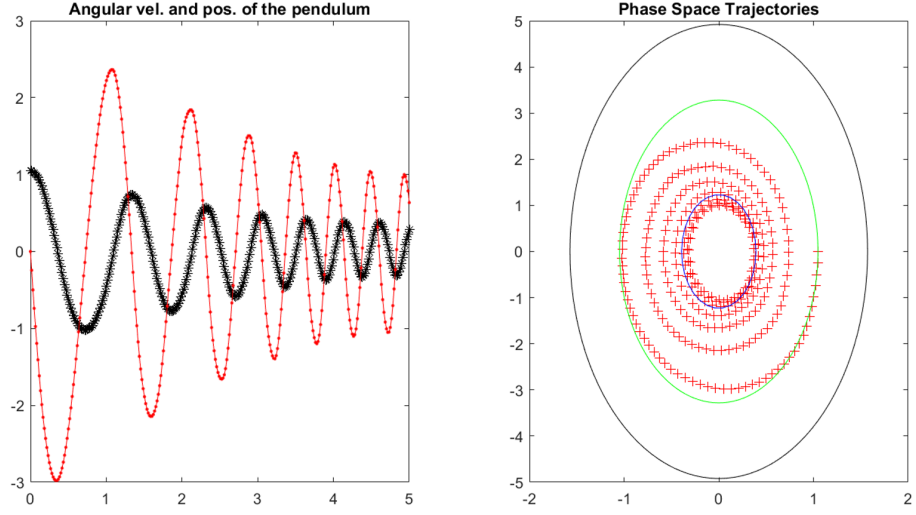


Figure 6: Prediction of the motion of a pendulum from the starting position of $\frac{\pi}{3}$ using the conventional loss function over 4 periods.

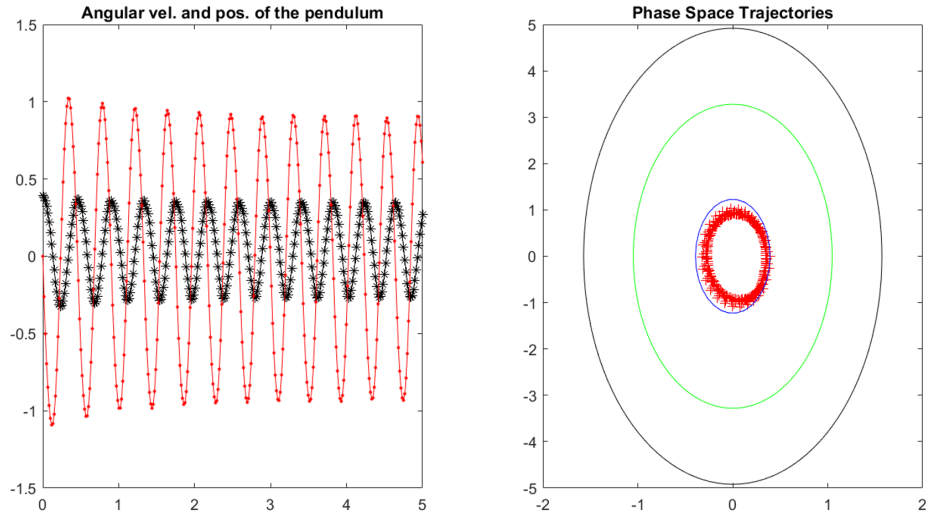


Figure 7: Prediction of the motion of a pendulum from the starting position of $\frac{\pi}{6}$ using the conventional loss function over 4 periods.

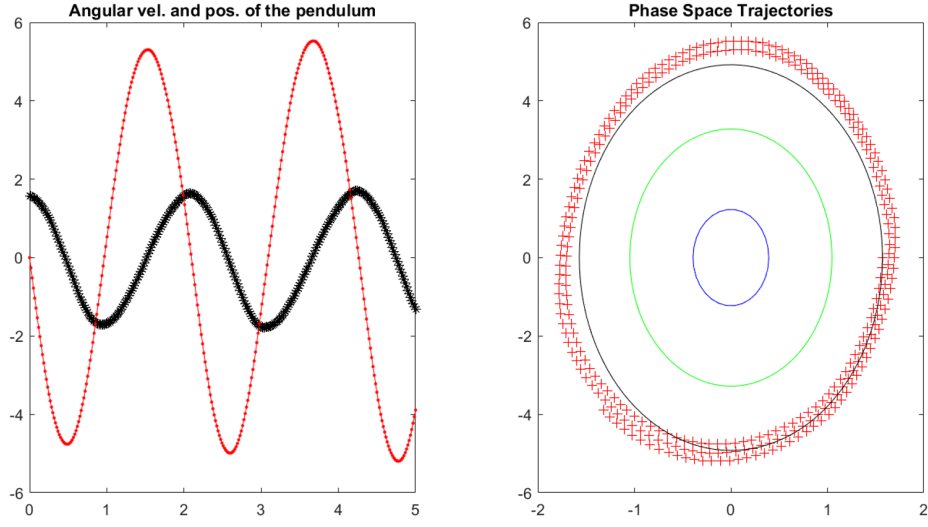


Figure 8: Prediction of the motion of a pendulum from the starting position of $\frac{\pi}{2}$ using the combined physics-based and conventional loss function over 4 periods.

3.2.2. Physics-based Loss Function

Figures 8 - 10 show the results of the prediction of the trajectory of the pendulum from the different starting positions using the neural network trained with the custom loss function adding the conservation of energy term. These results preserve the energy far better than the conventional approach and no decaying of the predictions is found for multiple periods.

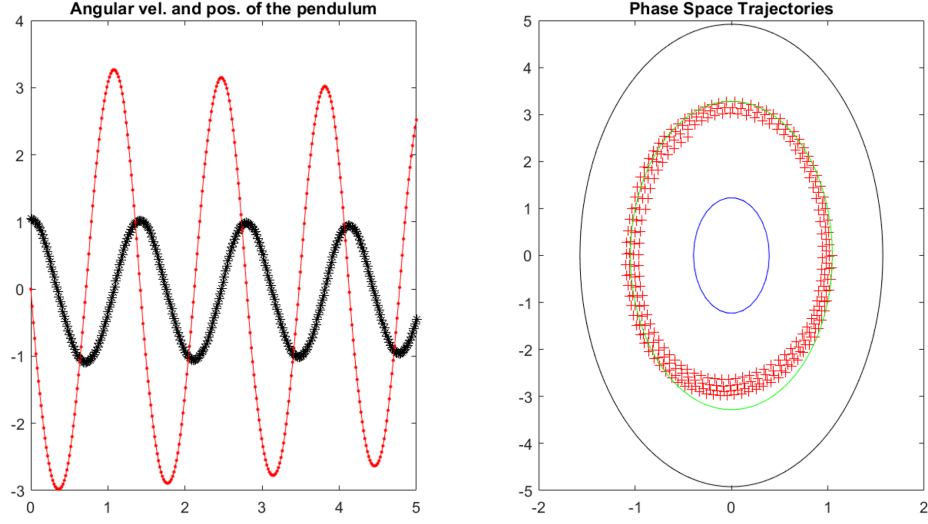


Figure 9: Prediction of the motion of a pendulum from the starting position of $\frac{\pi}{3}$ using the combined physics-based and conventional loss function over 4 periods.

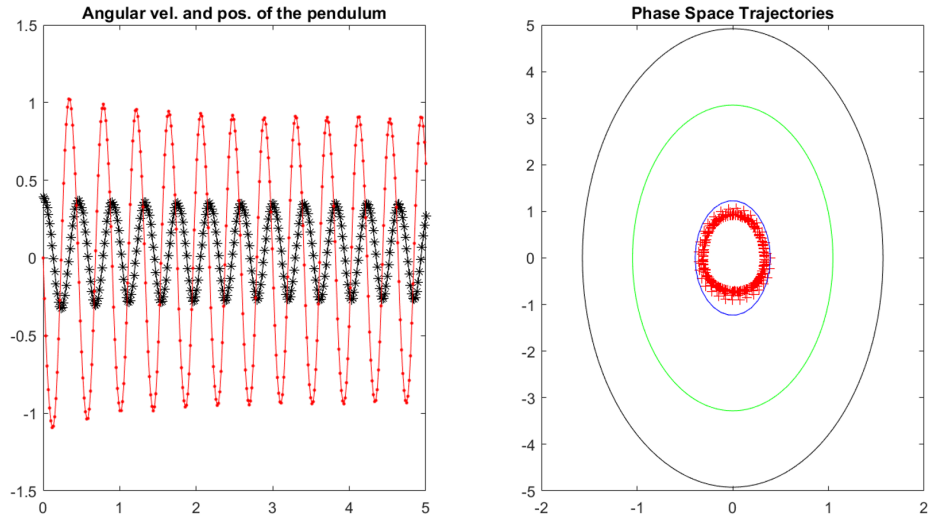


Figure 10: Prediction of the motion of a pendulum from the starting position of $\frac{\pi}{6}$ using the combined physics-based and conventional loss function over 4 periods.

4. Discussion

In this work a new approach to PIML-based neural networks was proposed that suggests adding only simple but universally applicable laws to the error function to enforce physically accurate predictions. Initially, a comparison was made between the process of training and in solving the fundamental principle in mechanics, the Principal of Least Action. Using the appropriate variables, it was shown that a standard loss function from deep learning, the mean squared error, when combined with output variables of position and momentum (or velocity) could solve the Action principle, albeit discretely and statistically. This was used to predict the full time-space trajectories of a number of representative mechanics problems, both linear and nonlinear in the form of projectile motion with and without drag, and a single and double pendulum system. By training the network on the initial conditions for a projectile’s flight, a network was built that could predict the trajectories of the masses, as shown in Figure 1 and with non-linear drag in Figure 2. While simple problems in mechanics, the benefit offered by this approach is computational speed and existing data and computational data. Projectile motions are commonplace in sports and military applications, and quickly predicting the path of a projectile is oftentimes needed. Additionally, two pendulum systems were modeled, and the trajectories of their masses were simulated, both for a single and double pendulum system. The trained neural network, while based on a simple fully connected set of neurons was very capable of reproducing the periodic behavior of the single pendulum (Figure 3) and also the more chaotic behavior of the double pendulum system (Figure 4). While it is well understood that chaotic systems are impossible to predict after a particular amount of elapsed time, this approach performed well to capture the nonlinear motions of the pendulum for as long as was shown. It would be interesting to explore this further to ascertain the limits of this approach for more detailed non-linear, chaotic, systems as they appear in nearly every aspect of human endeavor. The second focus of this work was to investigate the utility of adding a simple physics-based loss term into the loss function during training. For this work a neural network was constructed to predict the next time step of motion for a pendulum system, rather than the entire time-space history. This was intended to reflect a more useful use of this form of PIML loss function as only temporally-neighboring states of a system are needed for data in training, rather than the entire time history. A pendulum was modeled and, initially, a regular loss function using the mean-squared error was used to predict the time evolution of a pendulum when raised from a height of $\frac{\pi}{2}$ (Figure 5), $\frac{\pi}{3}$ (Figure 6), and $\frac{\pi}{6}$ (Figure 7). These results showed that the trained network was preferential to the lowest energy data and, after some initial success in prediction of the larger energy systems, decayed to the smallest energy state. This form of mode collapse is likely due to the prevalence in the training to minimize the magnitude of the error and the error itself, preferentiating the lowest valued data. When a new loss function that combined the standard mean-squared error with a measure of the difference in the energy between the input and output states was used, however, the results were drastically different. In each of the three cases (Figures 8 - 10) the neural network was able to predict the motion of the pendulum and conserve the energy far better than in the original case. It is worth noting that the values predicted by the second neural network in this case show a slightly less precise prediction as the values lie above or below the exact solution, but this is likely the effect of having to balance both this extra energy conservation constraint, and the minimization of the error in predicted values.

Despite the simplistic models used to test the proposed architecture, these results show a promising new avenue to explore when wanting to use simulated data and/or real-world data for physically relevant predictions and inferences. Adding new constraints such as the conservation of energy into the loss function is far more computationally efficient than the application of the traditional PINN approach and can be essentially universally applied to any system where the energy is conserved, or the loss to the system can be quantified appropriately. This also implies that other conservation law, linear and angular momenta, and even other forms of symmetries could be embedded to a loss function when the input and output neuron variables are physical quantities. However, as most models of interest do comprise of the same energy forms presented here, systems of multiple particles, and more complex dynamics may pose more difficult to predict as the number of phase-space dimensions increases. In this work only simulated data, which is often not a good measure of what can be expected of real-world data, was used meaning that the networks may not perform as well when managing real-world noisy data. There is also the need for a large amount of training

data that necessitates a large computational load to be performed before the training of the network can be done effectively. Also, while creating this data, the choice of the parameters and their ranges is a problem when considering how wide the spectrum of values would need to be for a neural network-based approach to be useful. However, as the networks are much faster to use once they are trained, and with the immense amount of existing simulated and real world data that already exists and is oftentimes just discarded anyway, the cost-benefit analysis looks promising for this form of PIML moving forward.

5. Conclusion

In this work we present a new form of physics-based loss function to train a neural network to prediction the evolution of a system. Comparisons are drawn between the process of backpropagation and the solving of the Principle of Least Action that governs all mechanical systems in physics. Predictions of the motion of projectiles and pendulums with both linear and nonlinear components are made with this approach to solving the Least Action principle, and a new loss function is constructed to predict the time step evolution of a mechanical system. Unlike conventional loss functions that only consider the error between the predicted and true value, here we show that in cases where systems of different energy are used as training data, conventional loss functions fail to properly predict different systems. Instead, we add an additional term to the error to enforce the conservation of energy between the input and the out of the network and the results show that this drastically improves the prediction of the evolution of different systems with differing levels of total energy. This simple but powerful alteration to the loss function in the field of Physics-Informed Machine Learning opens up a new set of opportunities to fuse simulation and real-world data for deep learning predictions of physical systems.

References

- [1] X. Bu, J. Rao, C.-Z. Xu, A reinforcement learning approach to online web systems auto-configuration, 2009, pp. 2–11, cited By 67. doi:10.1109/ICDCS.2009.76.
- [2] S. Vazquez, T. Muñoz-García, I. Campanella, M. Poch, B. Fisas, N. Bel, G. Andreu, A classification of user-generated content into consumer decision journey stages, *Neural Networks* 58 (2014) 68–81, cited By 24. doi:10.1016/j.neunet.2014.05.026.
- [3] G. Cheng, J. Han, A survey on object detection in optical remote sensing images, *ISPRS Journal of Photogrammetry and Remote Sensing* 117 (2016) 11–28, cited By 505. doi:10.1016/j.isprsjprs.2016.03.014.
- [4] G. Litjens, T. Kooi, B. Bejnordi, A. Setio, F. Ciompi, M. Ghafoorian, J. van der Laak, B. van Ginneken, C. Sánchez, A survey on deep learning in medical image analysis, *Medical Image Analysis* 42 (2017) 60–88, cited By 3406. doi:10.1016/j.media.2017.07.005.
- [5] J. Su, D. Vargas, K. Sakurai, One pixel attack for fooling deep neural networks, *IEEE Transactions on Evolutionary Computation* 23 (5) (2019) 828–841, cited By 280. doi:10.1109/TEVC.2019.2890858.
- [6] K. Hornik, Approximation capabilities of multilayer feedforward networks, *Neural Networks* 4 (2) (1991) 251–257, cited By 2585. doi:10.1016/0893-6080(91)90009-T.
- [7] S. J. Raymond, D. J. Collins, R. O’Rorke, M. Tayebi, Y. Ai, J. Williams, A deep learning approach for designed diffraction-based acoustic patterning in microchannels, *Scientific reports* 10 (1) (2020) 1–12.
- [8] J. Montgomery, S. Raymond, F. O’Sullivan, J. Williams, Shale gas production forecasting is an ill-posed inverse problem and requires regularization, *Upstream Oil and Gas Technology* 5 (2020) 100022.
- [9] X. Zhan, Y. Liu, S. J. Raymond, H. V. Alizadeh, A. G. Domel, O. Gevaert, M. Zeineh, G. Grant, D. B. Camarillo, Deep learning head model for real-time estimation of entire brain deformation in concussion, *arXiv preprint arXiv:2010.08527* (2020).
- [10] A. G. Domel, S. J. Raymond, C. Giordano, Y. Liu, S. A. Yousefsani, M. Fanton, N. J. Cecchi, O. Vovk, I. Pirozzi, A. Kight, B. Avery, A. Boumis, T. Fettes, S. Jandu, W. M. Mehring, S. Monga, N. Mouchawar, I. Rangel, E. Rice, P. Roy, S. Sami, H. Singh, L. Wu, C. Kuo, M. Zeineh, G. G. . D. B. Camarillo, A new open-access platform for measuring and sharing mtbi data 11 (7501). doi:https://doi.org/10.1038/s41598-021-87085-2.
- [11] S. J. Raymond, J. Maragh, A. Masic, J. R. Williams, Towards an understanding of the chemo-mechanical influences on kidney stone failure via the material point method, *Plos one* 15 (12) (2020) e0240133.
- [12] D. Collins, S. Raymond, Y. Ai, J. Williams, R. O’Rorke, M. Tayebi, Acoustic field design in microfluidic geometries via huygens-fresnel diffraction and deep neural networks, *The Journal of the Acoustical Society of America* 148 (4) (2020) 2707–2707.
- [13] X. Zhan, Y. Li, Y. Liu, A. G. Domel, H. V. Alidazeh, S. J. Raymond, J. Ruan, S. Barbat, S. Tiernan, O. Gevaert, et al., Prediction of brain strain across head impact subtypes using 18 brain injury criteria, *arXiv preprint arXiv:2012.10006* (2020).

- [14] Y. Liu, A. G. Domel, N. J. Cecchi, E. Rice, A. A. Callan, S. J. Raymond, Z. Zhou, X. Zhan, M. Zeineh, G. Grant, et al., Time window of head impact kinematics measurement for calculation of brain strain and strain rate in american football, arXiv preprint arXiv:2102.05728 (2021).
- [15] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, V. Lempitsky, Domain-adversarial training of neural networks, *Journal of Machine Learning Research* 17, cited By 1505 (2016).
- [16] S. Raymond, V. Lemiale, R. Ibrahim, R. Lau, A meshfree study of the kalthoff-winkler experiment in 3d at room and low temperatures under dynamic loading using viscoplastic modelling, *Engineering Analysis with Boundary Elements* 42 (2014) 20–25.
- [17] S. Raymond, Y. Aimene, J. Nairn, A. Ouenes, et al., Coupled fluid-solid geomechanical modeling of multiple hydraulic fractures interacting with natural fractures and the resulting proppant distribution, in: SPE/CSUR Unconventional Resources Conference, Society of Petroleum Engineers, 2015.
- [18] S. Raymond, E. Aimene, A. Ouenes, et al., Estimation of the propped volume through the geomechanical modeling of multiple hydraulic fractures interacting with natural fractures, in: SPE Asia Pacific Unconventional Resources Conference and Exhibition, Society of Petroleum Engineers, 2015.
- [19] S. J. Raymond, B. Jones, J. R. Williams, A strategy to couple the material point method (mpm) and smoothed particle hydrodynamics (sph) computational techniques, *Computational Particle Mechanics* (2016) 1–10.
- [20] S. J. Raymond, B. D. Jones, J. R. Williams, Modeling damage and plasticity in aggregates with the material point method (mpm), *Computational Particle Mechanics* 6 (3) (2019) 371–382.
- [21] S. Wiegold, Z. Liu, S. J. Raymond, L. T. Meyer, J. R. Williams, T. Buonassisi, E. M. Sachs, Detection of sub-500- μm cracks in multicrystalline silicon wafer using edge-illuminated dark-field imaging to enable thin solar cell manufacturing, *Solar Energy Materials and Solar Cells* 196 (2019) 70–77.
- [22] S. J. Raymond, B. D. Jones, J. R. Williams, Fracture shearing of polycrystalline material simulations using the material point method, *Computational Particle Mechanics* (2020) 1–14.
- [23] I. Harris, P. Jones, T. Osborn, D. Lister, Updated high-resolution grids of monthly climatic observations - the cru ts3.10 dataset, *International Journal of Climatology* 34 (3) (2014) 623–642, cited By 3789. doi:10.1002/joc.3711.
- [24] M. Meyers, A. Mishra, D. Benson, Mechanical properties of nanocrystalline materials, *Progress in Materials Science* 51 (4) (2006) 427–556, cited By 3265. doi:10.1016/j.pmatsci.2005.08.003.
- [25] R. Springer, D. Lowenthal, B. Rountree, V. Freeh, Minimizing execution time in mpi programs on an energy-constrained, power-scalable cluster, Vol. 2006, 2006, pp. 230–238, cited By 66. doi:10.1145/1122971.1123006.
- [26] M. Raissi, P. Perdikaris, G. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, *Journal of Computational Physics* 378 (2019) 686–707. doi:https://doi.org/10.1016/j.jcp.2018.10.045.
URL https://www.sciencedirect.com/science/article/pii/S0021999118307125
- [27] L. Zanna, T. Bolton, Data-driven equation discovery of ocean mesoscale closures, *Geophysical Research Letters* 47 (17) (2020) e2020GL088376, e2020GL088376 10.1029/2020GL088376. arXiv:https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2020GL088376, doi:https://doi.org/10.1029/2020GL088376.
- [28] S. Goswami, C. Anitescu, S. Chakraborty, T. Rabczuk, Transfer learning enhanced physics informed neural network for phase-field modeling of fracture, *Theoretical and Applied Fracture Mechanics* 106 (2020) 102447. doi:https://doi.org/10.1016/j.tafmec.2019.102447.
URL https://www.sciencedirect.com/science/article/pii/S016784421930357X
- [29] E. Haghighat, R. Juanes, Sciann: A keras/tensorflow wrapper for scientific computations and physics-informed deep learning using artificial neural networks, *Computer Methods in Applied Mechanics and Engineering* 373 (2021) 113552. doi:https://doi.org/10.1016/j.cma.2020.113552.
URL https://www.sciencedirect.com/science/article/pii/S0045782520307374
- [30] E. Haghighat, M. Raissi, A. Moure, H. Gomez, R. Juanes, A physics-informed deep learning framework for inversion and surrogate modeling in solid mechanics, *Computer Methods in Applied Mechanics and Engineering* 379 (2021) 113741. doi:https://doi.org/10.1016/j.cma.2021.113741.
URL https://www.sciencedirect.com/science/article/pii/S0045782521000773
- [31] E. J. Hall, S. Taverniers, M. A. Katsoulakis, D. M. Tartakovsky, Ginns: Graph-informed neural networks for multiscale physics, *Journal of Computational Physics* 433 (2021) 110192. doi:https://doi.org/10.1016/j.jcp.2021.110192.
URL https://www.sciencedirect.com/science/article/pii/S0021999121000875
- [32] Q. He, D. Barajas-Solano, G. Tartakovsky, A. M. Tartakovsky, Physics-informed neural networks for multiphysics data assimilation with application to subsurface transport, *Advances in Water Resources* 141 (2020) 103610. doi:https://doi.org/10.1016/j.advwatres.2020.103610.
URL https://www.sciencedirect.com/science/article/pii/S0309170819311649
- [33] X. Jin, S. Cai, H. Li, G. E. Karniadakis, Nsfnet (navier-stokes flow nets): Physics-informed neural networks for the incompressible navier-stokes equations, *Journal of Computational Physics* 426 (2021) 109951. doi:https://doi.org/10.1016/j.jcp.2020.109951.
URL https://www.sciencedirect.com/science/article/pii/S0021999120307257
- [34] M. Liu, L. Liang, W. Sun, A generic physics-informed neural network-based constitutive model for soft biological tissues, *Computer Methods in Applied Mechanics and Engineering* 372 (2020) 113402. doi:https://doi.org/10.1016/j.cma.2020.113402.
URL https://www.sciencedirect.com/science/article/pii/S0045782520305879
- [35] S. Wang, P. Perdikaris, Deep learning of free boundary and stefan problems, *Journal of Computational Physics* 428 (2021) 109914. doi:https://doi.org/10.1016/j.jcp.2020.109914.

- URL <https://www.sciencedirect.com/science/article/pii/S0021999120306884>
- [36] L. Yang, X. Meng, G. E. Karniadakis, B-pinns: Bayesian physics-informed neural networks for forward and inverse pde problems with noisy data, *Journal of Computational Physics* 425 (2021) 109913. doi:<https://doi.org/10.1016/j.jcp.2020.109913>.
URL <https://www.sciencedirect.com/science/article/pii/S0021999120306872>
 - [37] Z. Zhang, G. X. Gu, Physics-informed deep learning for digital materials, *Theoretical and Applied Mechanics Letters* (2021) 100220doi:<https://doi.org/10.1016/j.taml.2021.100220>.
URL <https://www.sciencedirect.com/science/article/pii/S2095034921000258>
 - [38] N. Zobeiry, K. D. Humfeld, A physics-informed machine learning approach for solving heat transfer equation in advanced manufacturing and engineering applications, *Engineering Applications of Artificial Intelligence* 101 (2021) 104232. doi:<https://doi.org/10.1016/j.engappai.2021.104232>.
URL <https://www.sciencedirect.com/science/article/pii/S0952197621000798>
 - [39] M. Mahmoudabadbozchelou, M. Caggioni, S. Shahsavari, W. H. Hartt, G. Em Karniadakis, S. Jamali, Data-driven physics-informed constitutive metamodeling of complex fluids: A multifidelity neural network (mfnn) framework, *Journal of Rheology* 65 (2) (2021) 179–198. arXiv:<https://doi.org/10.1122/8.0000138>, doi:10.1122/8.0000138.
URL <https://doi.org/10.1122/8.0000138>
 - [40] J. Johnson, A. Alahi, L. Fei-Fei, Perceptual losses for real-time style transfer and super-resolution, *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 9906 LNCS (2016) 694–711, cited By 2211. doi:10.1007/978-3-319-46475-6-43.
 - [41] J. Larmor, On the direct application of the principle of least action to the dynamics of solid and fluid systems, and analogous elastic problems, *Proceedings of the London Mathematical Society* s1-15 (1) (1883) 170–185, cited By 5. doi:10.1112/plms/s1-15.1.170.
 - [42] O. Agrawal, Formulation of euler-lagrange equations for fractional variational problems, *Journal of Mathematical Analysis and Applications* 272 (1) (2002) 368–379. doi:10.1016/S0022-247X(02)00180-4.