nature portfolio

Peer Review File

Training all-mechanical neural networks for task learning through in situ backpropagation

Corresponding Author: Professor Xiaoming Mao

This file contains all reviewer reports in order by version, followed by all author rebuttals in order by version.

Attachments originally included by the reviewers as part of their assessment can be found at the end of this file.

Version 0:

Reviewer comments:

Reviewer #1

(Remarks to the Author)
See my report in the attached pdf.

(Remarks on code availability)

Reviewer #2

(Remarks to the Author)
The authors present a proof-of-concept study on a limited form of backpropagation in linear mechanical systems. Overall, the core scientific content is great and the authors' beautiful experimental and theoretical results stand as great contributions on their own. Conversely, I found the paper to be written in a way that felt misleading or imbalanced - positive aspects felt oversold, and the many limitations of the work felt like they were not mentioned, or were mentioned only in the slightest detail. I can easily imagine the paper in its current form being frustrating for many readers - this is a shame because the core content is brilliant!

Here are my comments, many also in the attached PDF
- In the introduction the case for what the authors describe as MNNs compared to optical or wave-based PNNs is vague or unconvincing. The case for MNNs being useful beyond computation is made reasonably well. I recommend the authors consolidate these arguments somewhere, and consider which ones are convincingly justified and which ones are simply too vague. The impact and weight of the paper will be vastly improved by cutting the "fluff".
- There are several steps of the learning algorithm that require a digital computer, but this was never adequately explained, and I felt at times like the authors might be attempting to divert my attention from a fundamental issue. Overall while it is likely unintentional, I really did not like how the paper addressed this aspect. In a revised version the authors should clearly sketch out - in the main paper and likely the first figure - what is done physically and what must still be done with a conventional computer. For the parts done physically and with a conventional computer, the computational cost (e.g., scaling with number of parameters) should be written down. It should be noted not only that a digital computer is required to compute part of the total gradient, but also that it is necessary to measure the elongation, and to actually perform the update of the spring constant (assuming a device that has programmable springs)
- The MNNs used by the authors are introduced in a general way, but they make a number of assumptions implicitly that may or may not be obvious to the broad audience the paper is attempting to reach. Unless there is some paper the authors can cite which introduces a canonical definition of this type of mechanical neural network, they should take more care in introducing what the physical system is (i.e., what class of systems, with some concrete examples, the theoretical model describes) and what it neglects (e.g., nonlinearity, dissipation).
- I felt like the authors failed to directly address most of the relevant literature. The work from Murugan's group or the Penn groups covers very similar topics and it is important that the authors clearly explain how their work goes beyond this (which includes but is not limited to the experimental demonstrations in 3D-printed mechanics). Ref. 19 is also very relevant, but is hidden in a big "lump citation". I know how easily this can happen during the writing process, but to a skeptical reader this feels a little "off".

- With respect to this paper's relation to prior work, it deserves to be mentioned the relationship with coupled learning and equilibrium propagation. Is this technique actually materially different from those techniques? This may not be as obvious as it seems. I can certainly see that it has been derived in a different way but even after reading the paper in full, I admit I am not that sure how much these algorithms actually differ in terms of what they entail in practical implementation.
- One practical issue is that the experiments are performed with a platform that is, as far as I understood, not exactly reprogrammable, or at least not in the rapid, spring-by-spring way that would be required to do a full training process. The authors must note this! The way the paper is written now, it is not actually obvious and, given the dearth of experimental details outside the SM, convincing oneself of this with certainty takes a longer time than should be necessary. The fact this works despite that is impressive, but it is notable as a deficiency of the proof-of-concept
- Relatedly, it would be beneficial for the authors to discuss routes to such a programmable device implementation, as well as generalizations to the case of nonlinearity

Overall, I would like to recommend publication but in its current form there are many things that need to be changed (primarily just in the presentation/description) before I could do that. To some extent whether I recommend publication in Nature Communications or a different journal is contingent on how some of the above issues are resolved, though I think it is more likely than not that I will recommend publication if the authors can realize a good-faith revision along these lines.

(Remarks on code availability)

Reviewer #3

(Remarks to the Author)
Li and Mao conduct a hybrid simulation/experimental study in which they train simulated linear spring networks to perform a variety of tasks ranging from simple single-input tasks to the iris dataset (4 input 3 class classification task). They do so using an implementation of the adjoint method, which they derive and experimentally verify. This method is local, meaning that only direct measurements of a given piece of the network are required to evolve it in the proper direction to lower the given loss function. The final simulated networks are physically constructed and experimentally tested for several tasks. The authors show in simulation that such a mechanical system trained with this method can be retrained for multiple tasks, and additionally recover from physical damage if training is performed again.

Overall, the study was interesting, and it was excellent to see experimental realizations of the simulations. However, the study lacked sufficient novelty for Nature Communications:

The proposed method is – I believe - identical to Equilibrium Propagation (Scellier and Bengio 2017), and this work was not cited or mentioned. If the methods are not the same, then a difference should be explicitly stated. If so, how does the work compare to implementations of EP (e.g. Laydevant et. al. Nature Communications 2024)?

The experiments in this work were replicas of the simulation (other works use local rules in mechanical networks in simulation, e.g. Stern et. al. PRX 2021), and as far as I could tell there was no feedback between the experiment and simulation. Further, the tasks in this work were extremely similar to those in Dillavou et. al. PR Applied 2022, which was an entirely experimental work.

Finally, much of the authors' claims of value rest on their method's superiority to other methods, yet show no comparison to other local methods like Coupled Learning. Further, systems trained in silico using any method (regular backpropagation, equilibrium propagation, coupled learning, the authors' scheme) would have the retrainable properties shown in Fig 5.

As such I do not think this study meets the high bar for novelty and broad interest required for Nature Communications.

Below I include minor points that I hope will help improve the manuscript.

In several places the authors could make it easier to evaluate their experimental results. Specifically, in figures 3 and 4, the loss and accuracy of the experiment are not shown, but are arguably the most important quantities in the manuscript.

The authors imply on page 2 that mechanical neural networks may improve the speed of traditional neural networks, which I do not believe is supported by their references. Neural networks are extremely fast! The (I believe incorrect) speed advantage is mentioned again at the start of section 2.3.

MNN should be defined at its first use.
I am not familiar with the phrase "behaviors learning" (in contrast to machine learning tasks) if it is not a standard term I suggest the authors define it explicitly.

Backpropagation should be defined when it is first used in the text.

The authors state that they retrieve the exact gradient of the loss "essentially without a computer." I understand the spirit of this claim and agree with its importance, but I do not think this phrasing is convincing or appropriate, as a computer was

certainly used. “using local rules” or “without taking derivatives” or another more precise claim would work better.

I found derivation of eqs 3-6 difficult to follow. There are several small but key details left out that would greatly help readability. For example, pointing out that dF/dk = 0 helps get to eq (3), D being symmetric allows D = D^T, necessary for multiple steps, etc. Further, it is unclear why u*, then u*^T, then u_{adj} are all defined in quick succession – are all needed? Could the authors just start with u_{adj}^T ?

Perhaps point out earlier (before eq 5 or 4?) that the adjoint displacement field can be thought of (and indeed will be) the response to an imposed force at the output dependent on the output and the loss function? Or perhaps define “adjoint problem” explicitly.

I am unsure what the “unexpected spikes in the decreasing loss when using coupled learning rules” refers to in references 31,32. In the former, the authors physically damage the system, which produces spikes in error, but that is similar to figure 5 in this work.

“our experimental gradient achieves over 90% accuracy” is confusing – do the authors mean that the average error is less than 10% of the true value? How is the averaging done?

The authors’ method is an approximation, and a key piece of information required to understand this (“in the linear regime” after eq 5) should be stated more clearly. I also think the authors should remove “In sharp contrast, the numerical implementation of our method does not produce error” for this reason. Further, the numerical implementation of the authors’ method certainly produces roundoff error, as every computational calculation does – perhaps showing numerical error as a function of adjoint force, the relevant small parameter in this situation. This should be added to Fig1e for a fair comparison, or most of the x range should be removed, leaving only the experimentally feasible region, and experimental error.

In figure 1b, perhaps the authors could use the empty space in the top right block to plot experimental vs simulated gradient against each other – it is very hard to get a quantitative sense of the quality of gradient from the heatmaps. Additionally it would help readability of the heatmaps to make the network edges thicker.

“another advantage of our MNNs is the potential utilization of the same node as both input and output node” – is this not feasible in other physical networks? E.g. in electrical networks, imposing current as an input and retrieving voltage?

I enjoyed the supplementary videos very much, however it is hard to tell what is going on in the network, since all that is shown is the structure changing. Additionally showing the state when force is applied would help greatly!

“It is noteworthy that using the cross-entropy function as the loss encourages the maximization of the displacement difference between two nodes, rather than a specific value when using mean-squared error (MSE) as the loss.” Why did the authors choose not to use MSE of the difference between the node heights? Would that not have worked as well?

2.3.1 – it would be helpful if the authors could cite a regression benchmark for machine learning. The most famous ones are all classification (e.g. MNIST, CIFAR-10, etc).

Why does the data in Fig 3a include upward forces if they cannot be used in the experiment?
The training and test error in Fig 4 look to be about 95%, which is excellent for a linear system, but I think the authors should remove “nearly 100%” and be more quantitative about these results. Additionally, it is hard to see the final accuracy with such a large y scale (Fig4b).

It is difficult to tell how well the system is doing from the plots in Fig4c at a non-approximate level. Perhaps the authors could highlight the incorrectly identified flowers?

Are there differences between the networks in the images Fig 4d? If so they are too subtle to tell. What is the experimental classification accuracy?

In Fig 4, was a different input/output configuration attempted? If so did it succeed? The success at this task may depend on the specific node placements, since there are only so many connections in the network.

The task switching figure and supplementary video are great.

The authors point out retainability several times as an advantage, but their experimental apparatus is manufactured with specific weights and must be entirely remanufactured if any changes are made. I think these claims are premature for this work.

“In contrast to other physical neural networks, such as optical neural networks, our MNNs offer greater ease of operation in experiments and real-world applications across diverse environments.” I do not think this claim is justified and should be removed.

(Remarks on code availability)
I could not access the code through the above DOI.

Version 1:

Reviewer comments:

Reviewer #1

(Remarks to the Author)
The authors have sufficiently addressed many of my comments. However, one important comment remains, having to do with the relations of in-situ backpropagation and the equilibrium propagation algorithm. I still maintain that both are the same for linear systems. They certainly suggest the same learning rule. The rebuttal offered by the authors does not contradict this observation. Specifically, the comment that both algorithms are the same in the limit \beta -> 0 is indicative, as equilibrium propagation is after all defined in that limit.
I do appreciate that in situ backpropagation is derived in a different way then the original EqProp, and that this derivation technique may give unique learning rules for other physical systems, yet in linear networks both are in fact the same. Still, I believe that the current work is sufficiently novel, both in its derivation of the in situ backprop rule, and the way that it was used in theory and experiments of elastic structures. I'd be happy to recommend the paper for publication once the authors addressed this issue, dropping the claim that in situ backpropagation gives a different result (and certainly not superior) result than equilibrium propagation.

(Remarks on code availability)
I did not review the code as I do not have access to Matlab at the moment.

Reviewer #2

(Remarks to the Author)
I am more or less satisfied with the author's revisions and for my part am ready to see the work published. I expect there to be some further debate and discussion over exactly the nature of the novelty in the work, but I think this is resolved sufficiently to justify publication (and any further debate or refinement would, I think, benefit the community anyway)

Thanks to the authors for the care in revising the manuscript and addressing my concerns.

(Remarks on code availability)

Reviewer #3

(Remarks to the Author)
The authors have answered the bulk of my comments thoroughly, and I thank them for their hard work.

The text is much improved and the claims have been appropriately toned down, with needed additional clarifications about the linearity of their method and scope of their experimental work.

I now agree that their method and Equilibrium Propagation are, strictly speaking, distinct, for which their derivation was very helpful. The lack of a small parameter in their method is indeed a plus. However, as the authors state, their method is ‘exact’ only in the linear regime, therefore requiring small deformations of the system, which is in effect enforcing a small parameter. I believe extending this method to nonlinear systems will be difficult (and is the reason EP, Coupled Learning, and Contrastive Learning compare two like-states instead of applying the adjoint directly), however as long as the authors are clear about the scope of validity of their method, this seems like reasonable future work.

Notably, the abstract has remained unchanged from the previous version, which I believe is still a bit misleading. Specifically the mention of “exact” gradient without qualification and ambiguity about what was done in experiment and what was done in simulation.

Further, in the second to last paragraph of the introduction the statement “we showcase the successful training of a mechanical network” comes directly after mentioning experimental verification, and seems to imply the training was in the mechanical network. There are several instances of this implication in the paper, which confused and irked me as a reader. Making this aspect much clearer would help readers like me (and from the review, like Reviewer 2 as well) to not feel like there are details being hidden.

I agree with Reviewer 2’s assessment of much of this manuscript – there is excellent work here, but it was confusing as to what was actually done and in some places oversold. In its improved form, I will not stand in the way of this work’s publication, provided appropriate changes are made to the abstract and introduction.

A few additional comments:

The authors should clarify that [33] and [35] are simulations of Equilibrium Propagation, and not experimental demonstrations.

Coupled learning cannot precisely resolve an “arbitrary loss function” (though strictly speaking neither can any of the other discussed methods), but it can be used to solve arbitrary tasks. Could the authors rephrase this sentence?

The authors denote “100%” accuracy in reference to Fig 4d, by which I believe they mean for the three shown examples only? This is confusing as the simulation and experiment do not have 100% accuracy overall, as shown by 4b and 4c.

(Remarks on code availability)

Version 2:

Reviewer comments:

Reviewer #1

(Remarks to the Author)
I am pleased with the further revisions provided by the authors and now recommend the paper for publication at Nature Communications.

(Remarks on code availability)

Reviewer #3

(Remarks to the Author)
The authors have adequately addressed my concerns.

(Remarks on code availability)

Reviewer #1 (Remarks to the Author):

Training all-mechanical neural networks for task learning through in situ backpropagation – Review

This work introduces a training approach for mechanical neural networks (MNNs), spring networks whose spring constants are physically adapted such that the network exhibits desired responses (outputs) to applied forces (inputs). The learning rule used, dubbed in situ backpropagation, is based on the adjoint variable method, and recovers the gradient of a user-defined cost function. It is a local learning rule that allows the efficient realization of gradient descent in simulated, and possibly also physical, spring networks. The authors use this method to simulate training spring networks with finite-element models, yielding models with desired mechanical behaviors or machine-learning-like tasks, including linear regression and classification (on the Iris dataset). Such computational models can also recover from damage, simulated as removal of springs. Remarkably, the authors are also able to realize these simulated models in the lab and verify they perform the trained tasks.

I believe this is a good paper, describing a useful approach for implementing learning in spring networks. Such networks, being intrinsically non-linear, may possibly be used to exhibit a large variety of interesting and useful behaviors, specifically material properties that are not easily found in natural materials. As such, this paper contributes an interesting and timely perspective to the growing literature on self-learning in physical systems. The derivations and analyses are correct and clearly stated, and the experimental realization in these 3d-printed networks is novel and beautiful. Furthermore, the paper is well-written and reads fluently. I'd be happy to recommend the paper for publication once my questions and concerns, listed below, mostly related to the framing of certain concepts and clarifications of some statements, are addressed.

1. Certain claims made in the paper about MNNs are too strong, or at least not really substantiated. In particular, while I can see how MNNs would be energy efficient, I do not see much reason to believe they can perform computations faster than digital computers. Furthermore, the authors mention that MNNs "show interpretability", possibly hinting that the solutions found by in situ backpropagation are easier to understand in terms of how they give rise to the desired behavior, but this claim is not substantiated in the paper. These statements should be qualified or toned down.
2. The choice of the name in-situ backpropagation is quite interesting, but perhaps also confusing. Backpropagation usually refers to using the chain rule to compute the gradient of a deep neural network model. It is also usually invoked for feed-forward networks, though the term recurrent backpropagation is also used (e.g. the Almeida-Pineda algorithm). In this work, there are no neural networks, and the gradient is not computed in a way reminiscent of backpropagation in machine learning. Instead, the contrast between two states is used (as discussed in the top of pg. 5). Functionally speaking, this is a contrastive learning algorithm, specifically, Equilibrium Propagation (Scellier & Bengio, Frontiers in Comp. Neuro, 2017). The particular learning rule in Eq. (6) here was

derived recently in fluidic networks (Anisetti, Scellier, Schwarz, Phys, Rev Res, 2022) and in spring networks (Stern, Liu, Balasubramanian, Phys Rev E, 2024) using the Equilibrium Propagation algorithm. I'd suggest that the authors clarify this point, and would like to see the result related more clearly to the contrastive learning framework.

3. The name in-situ backpropagation is also a bit confusing, because while I believe a spring network realizing the learning rule of Eq. (6) could be built in principle, the authors simulate this learning rule on a computer running a finite element model of a spring network. In fact, I think the statement in the discussion that "the experimental feasibility… had been demonstrated in Fig 1" is misleading, because the learning rule was definitely not demonstrated in the experimental substrate itself. The authors mention in the discussion "numerous experimental avenues" for implementing the learning rule, listing systems with variable spring constants. I would appreciate if the authors discussed this a bit more, specifically if they see any particularly promising way of realizing such a learning network. This is quite important, because I do not think the computational implementation of such learning rules give much, if any, benefit in producing designed networks compared to standard gradient descent (see more in point 4 below). For example, while the authors obtain nice matching between their computational models and the experimental ones, there is still remaining discrepancy (aptly named by Wright, Onodera, et. al., Nature 2022, the "simulation-reality gap"). This discrepancy will likely get worse for larger networks or small components, where fabrication at scale becomes error prone. In this context, it would be good to acknowledge prior work of Pashine, Phys. Rev. Mat. 2021, which experimentally implemented training using a local contrastive learning rule to prune an elastic network, in which the evaluation and execution of the update was done "manually" in the experiment, overcoming the simulation-reality gap.
4. The authors compare their computational evaluation of the gradient to that obtained by finite differences, Eq. (8), and show their computational implementation of in situ backpropagation is superior in terms of the error produced by the calculation, related to an additional free parameter δk associated with finite differences. Further, the authors correctly claim that in situ backpropagation requires two computations of the system configurations, while the finite differences requires a number of computations that scales with system size. This comparison may give a false impression that in situ backpropagation is computationally superior to direct computation of the gradient. In fact, one can (at least in principle) use automatic differentiation to evaluate these gradients due to a change in some parameters (as routinely done in machine learning, and increasingly in condensed matter physics simulations). Evaluating the gradient with automatic differentiation requires essentially one evaluation of the system configuration, so it will not scale any worse than Eq. (6), and will not introduce the errors described by the authors for finite difference evaluations.
5. Some clarifications on the training process would be useful. In particular, the spring constants are presumably non-negative numbers. What happens in the algorithm when negative spring constants are requested? Is it important that the actual gradient of the cost function is no longer feasible experimentally?

Reviewer #1 (Remarks to the Author):

6. It is not clear what is meant by “the noncontroversial nature of input force and output displacement”. Is it that they are easily distinguishable? Could you also train the input and output on the same node by, say, having the input be horizontal displacement and the output vertical displacement?
7. In Figure 2, I think it is sufficient to show one scenario with a-symmetric displacement. The other is trivially symmetric to the first one, and displaying it confused me as to whether the same system has two different behaviors (which is unfortunately not the case).

Reviewer #1

This work introduces a training approach for mechanical neural networks (MNNs), spring networks whose spring constants are physically adapted such that the network exhibits desired responses (outputs) to applied forces (inputs). The learning rule used, dubbed in situ backpropagation, is based on the adjoint variable method, and recovers the gradient of a user-defined cost function. It is a local learning rule that allows the efficient realization of gradient descent in simulated, and possibly also physical, spring networks. The authors use this method to simulate training spring networks with finite-element models, yielding models with desired mechanical behaviors or machine-learning-like tasks, including linear regression and classification (on the Iris dataset). Such computational models can also recover from damage, simulated as removal of springs. Remarkably, the authors are also able to realize these simulated models in the lab and verify they perform the trained tasks.

I believe this is a good paper, describing a useful approach for implementing learning in spring networks. Such networks, being intrinsically non-linear, may possibly be used to exhibit a large variety of interesting and useful behaviors, specifically material properties that are not easily found in natural materials. As such, this paper contributes an interesting and timely perspective to the growing literature on self-learning in physical systems. The derivations and analyses are correct and clearly stated, and the experimental realization in these 3d-printed networks is novel and beautiful. Furthermore, the paper is well-written and reads fluently. I'd be happy to recommend the paper for publication once my questions and concerns, listed below, mostly related to the framing of certain concepts and clarifications of some statements, are addressed.

We thank the reviewer for the time spent in reviewing our manuscript and for the positive comments to our work! In the following, we address the reviewer's comments point by point, to the best of our ability, and we have revised our manuscript accordingly.

---

1. Certain claims made in the paper about MNNs are too strong, or at least not really substantiated. In particular, while I can see how MNNs would be energy efficient, I do not see much reason to believe they can perform computations faster than digital computers. Furthermore, the authors mention that MNNs "show interpretability", possibly hinting that the solutions found by in situ backpropagation are easier to understand in terms of how they give rise to the desired behavior, but this claim is not substantiated in the paper. These statements should be qualified or toned down.

   We thank the reviewer for the comments. We have adjusted these claims in the manuscript and added substantial discussions to more accurately reflect the comparison.

   Regarding the claim about computational speed, we state in the Conclusion section in our original manuscript "Also, MNNs provide fairly fast information processing at the speed of sound of materials in both in situ backpropagation at the training stage and decision-making at the prediction stage." While theoretically, the speed of information propagation in MNNs could be rapid due to the speed of sound in the materials, current implementations indeed

operate slower than digital computers. To avoid such confusion, we clarified this point in our revised manuscript.

Regarding the interpretability of MNNs, this stems from the physically-manufactured structures where the "weights" are visible and it is more straightforward to understand how these configurations lead to the desired behaviors, especially when contrasted with the black-box nature of traditional neural networks. For example, in the behavior learning example we presented, if we desire a larger displacement at the left node, it intuitively follows that the stiffness of the left side of the MNN should be less than that of the right side. This intuitive understanding is reflected in the trained MNN configurations shown in Fig. 2c. However, such intuitive interpretations may not directly apply to machine learning tasks involving datasets like regression and classification. Also considering the comment from Reviewer #2, we revised our manuscript by deleting such statements in page 8.

The revised manuscript in page 2 is attached as below:

"*One proposed solution lies in physical machine learning hardware platforms relying on physical processes, such as optical and mechanical neural networks, which hold promise for better energy efficiency compared to their digital counterparts [12-21]*"

The revised manuscript in page 11 is attached as below:

"*MNNs offer an avenue for implementing machine learning tasks with improved energy efficiency.*"

---

2. The choice of the name in-situ backpropagation is quite interesting, but perhaps also confusing. Backpropagation usually refers to using the chain rule to compute the gradient of a deep neural network model. It is also usually invoked for feed-forward networks, though the term recurrent backpropagation is also used (e.g. the Almeida-Pineda algorithm). In this work, there are no neural networks, and the gradient is not computed in a way reminiscent of backpropagation in machine learning. Instead, the contrast between two states is used (as discussed in the top of pg. 5). Functionally speaking, this is a contrastive learning algorithm, specifically, Equilibrium Propagation (Scellier & Bengio, Frontiers in Comp. Neuro, 2017). The particular learning rule in Eq. (6) here was derived recently in fluidic networks (Anisetti, Scellier, Schwarz, Phys, Rev Res, 2022) and in spring networks (Stern, Liu, Balasubramanian, Phys Rev E, 2024) using the Equilibrium Propagation algorithm. I'd suggest that the authors clarify this point, and would like to see the result related more clearly to the contrastive learning framework.

We thank the reviewer for the comment and bringing to us important relevant literature, some of which we were not aware of. We have added to the manuscript references to these papers and discussions on their relations to our work.

Relation to backpropagation. We understand that in traditional computer-based neural networks, backpropagation involves propagating error signals backward through the network

to determine gradients using the chain rule. This process features a forward pass where the input signal is fed forward and a backward pass where the error signal is propagated back. Similarly, in our work, we employ the adjoint method to derive the gradient of the loss function, which also consists of a forward pass and a backward pass. The backward pass, indicated by the adjoint force $-\left(\frac{\partial \mathcal{L}}{\partial u}\right)^T$, represents the error between prediction and target. For instance, when $\mathcal{L} = (u - u_T)^2$, the non-zero entry in $-\left(\frac{\partial \mathcal{L}}{\partial u}\right)^T$ corresponds to $2(u_T - u)$ for the output node. Moreover, previous works (Hughes et al., Optica, 2018; Pai et al., Science, 2023) on photonic neural networks have used the adjoint method and usually adjoint-method-based backpropagation or in situ backpropagation (if the backpropagation happens in the same system) is used to describe this training method (Momeni et al., arXiv, 2024).

Relation to contrastive learning frameworks. Contrastive learning, such as Equilibrium Propagation (EP) and coupled learning, uses two states (a free state at equilibrium and a nudged state at a new equilibrium slightly away from the original equilibrium) to approximate the gradient via their difference. Our approach also has two steps and two equilibrium states. However, our method introduces *two completely different equilibrium states induced by forward forces and adjoint (backward) forces, respectively.* These two equilibrium states can be significantly different from each other, and the forward forces are absent in the backward pass, distinct from the case of EP. Besides, algorithmically, in our method, the element-wise multiplication of elongations in two states yields the exact gradient, while in contrastive learning, the difference between two states approximates the gradient. We further derive a link between our approach and EP in the Supplementary Information that $\Delta k_i = \Delta k_{\mathrm{EP},i} + \frac{1}{2}\alpha\beta e^2_{\mathrm{adj},i}$. One can expect that when the nudge strength $\beta \to 0$, EP approaches our method.

It is encouraging to notice that the learning rule in this paper (Anisetti, Scellier, Schwarz, Phys, Rev Res, 2022) is similar to ours, where the pressure drop and concentration drop produces the gradient. If we regard the elongation in MNNs as "displacement drop", our learning rule has the same characteristics with theirs. However, we introduce a physical system distinct from theirs. Fluid networks are essentially scalar networks where each node has one degree of freedom, and bond direction is irrelevant. In contrast, mechanical networks are vector networks where each node has d degrees of freedom (with d denoting the dimension), and bond directions matter. Furthermore, our derivation of the learning rule using the compatibility matrices is more systematic and ready to generalize to other types of networks.

The paper (Stern et al. Phys. Rev. E, 2024) uses EP. The difference and link between EP and our method can be seen in the previous paragraph ("Relation to contrastive learning frameworks.").

Though MNNs lack the conventional layer-by-layer structure of neural networks, where the information flows in all directions, making our backpropagation distinct, we believe that our learning rule bears a *closer resemblance to backpropagation than to contrastive learning.* Therefore, we prefer to describe our method as the mechanical analogue of in situ back-

propagation. We also revised our manuscript by comparing our method with the contrastive learning framework. Note that to have a fair comparison, we compare our method with the case of EP where an explicit loss function can be defined.

The revised manuscript in page 3 is attached as below:

"*In light of this, the concept of* physical learning *offers a promising avenue to train MNNs only based on local information of the networks, inspired by neuroscience [27]. This methodology is promising for experimental implementations where the learning occurs physically in the system. One method, termed coupled learning, uses the contrast of two equilibrium states (free and nudged states) to update the system but cannot define the arbitrary loss function [28-31]. This method has enabled MNNs to perform well in supervised machine learning tasks. Each time a training example of a force pattern is applied to the MNN, it updates itself according to the learning rule. Over time, the MNNs learn to accurately respond to unseen forces that have spatial correlation patterns resembling those in the training examples.* "

"*Another well-known method for* physical learning *is Equilibrium Propagation (EP), sharing similar procedure with coupled learning but being able to define the arbitrary loss function [32]. This method has been demonstrated in various physical systems such as nonlinear resistor networks [33], Ising machines [34] and coupled phase oscillators [35].*"

The revised manuscript in page 6 is attached as below:

"*Similar to the energy-based learning methods, such as EP and coupled learning described in the Introduction, our method also introduces two equilibrium states induced by the forward force* F *and the adjoint force* $-\left(\frac{\partial \mathcal{L}}{\partial u}\right)^T$*. However, in energy-based learning methods, the nudged states are slightly away from the free states controlled by the nudging strength, whereas the two equilibrium states in our method are* independent*, where the input forces are absent in the second state. Besides, algorithmically, the difference between two equilibrium states approximates the gradient in energy-based learning methods but our method yields gradient through the multiplication, performing* backpropagation *in the MNN. In the Supplementary Information, we show the link between EP and our method.*"

"*When we further think about the form of the adjoint forces* $-\left(\frac{\partial \mathcal{L}}{\partial u}\right)^T$*, where the nonzero entry is* $2(u_T - u_p)$*, it can be regarded as the error between the output and the desired output. Therefore, essentially our method provides two signal passes, a forward pass sending the input signal and a backward pass sending the error signal, reminiscent of the adjoint-method-based backpropagation in optical neural networks [14,22,41]. Furthermore, recent work on training chemical signaling networks shows the similar procedures and conclusions, where the concentration drop and the pressure drop along the bond produces the gradient, derived by the special form of the system matrix [42]. Likewise, in our linear MNNs, the elongation can be regarded as the displacement drop along the bond, and thus the learning rules share the common characteristics.*"

The revised manuscript in page 16 is attached as below:

"*Besides, unlike fluid networks [29], chemical signaling networks [42] and electric networks [49], where both input and output are scalar quantities, our mechanical networks feature the vector input and output. This characteristic allows for more opportunities in higher dimensions.*"

The revised manuscript in page 9 is attached as below:

"*We also conduct the EP to estimate the gradient of* $\mathcal{L} = (u_{Ly} + 0.025)^2$ *and compare the results with ours, as described in Supplementary Information and Supplementary Figure 3. When the nudging strength is small, the results from our method and EP are almost identical. But with the increase of the nudging strength, the gradient error becomes large.*"

The revised Supplementary Information in page 4 is attached as below:

"***Comparison between our method of in situ backpropagation and equilibrium propagation.*** *Equilibrium propagation (EP) is a kind of energy-based learning framework [1]. Compared with other energy-based learning algorithm, such as contrastive learning and coupled learning, EP explicitly introduces the loss function* $\mathcal{L}$ *to represent the difference between the output and the desired output [2]. The implementation of EP for the MNNs has two phases. In the first phase, similar to our method of in situ backpropagation, the input forces are applied to the MNNs to reach the equilibrium state. In the second phase, EP proceeds by nudging the output nodes using forces* $-\beta\frac{\partial \mathcal{L}}{\partial u}$*, where* β *is called the nudging strength. In what follows, we apply EP to our MNNs to approximate the gradient of the loss function. Given the elongation of MNNs in two phases,* e *and* e_{nudged}*, the update of spring constant is given by* $\Delta k = \frac{\alpha}{2\beta}(e^2 - e_{\text{nudged}}^2)$*, where* α *is the learning rate. Algorithmically, the element-wise multiplication between forward elongation and adjoint elongation produces the exact gradient, while the difference between two states approximates the gradient.*
Now we show the link between our method and equilibrium propagation. The first phase is the same as that in our method, described by the governing equation:

$$Du = F.$$

The second phase requires a nudge on the output node, which can be expressed as:

$$Du_{\text{nudged}} = F - \beta \left(\frac{\partial \mathcal{L}}{\partial u}\right)^T.$$

Recall that the second phase for our method is:

$$Du_{\text{adj}} = -\left(\frac{\partial \mathcal{L}}{\partial u}\right)^T.$$

Therefore, in the linear regime, $u_{\text{nudged}} = u + \beta u_{\text{adj}}$ *and* $e_{\text{nudged}} = e + \beta e_{\text{adj}}$*. The update rule*

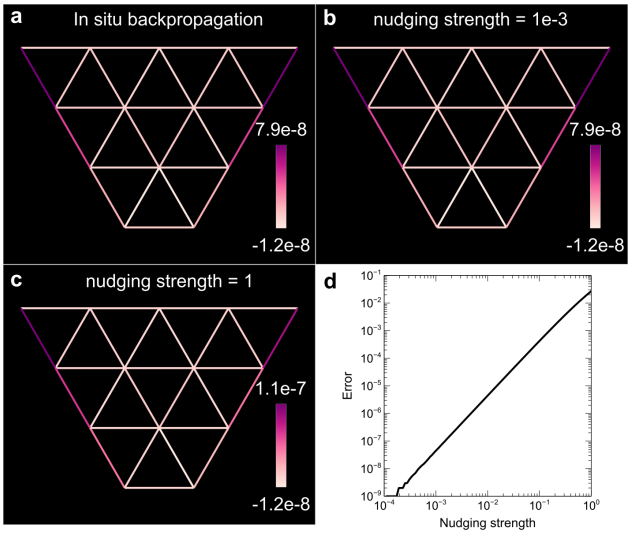
for our method is:

$$\begin{aligned}\Delta k_i &= -\alpha e_i e_{\mathrm{adj},i} \\ &= \frac{\alpha}{\beta}\left[\frac{1}{2}e_i^2 + \frac{1}{2}\beta^2 e_{\mathrm{adj},i}^2 - \frac{1}{2}(e_i + \beta e_{\mathrm{adj},i})^2\right] \\ &= \frac{\alpha}{\beta}\left[\frac{1}{2}(e_i^2 - e_{\mathrm{nudged},i}^2) + \frac{1}{2}\beta^2 e_{\mathrm{adj},i}^2\right] \\ &= \Delta k_{\mathrm{EP},i} + \frac{1}{2}\alpha\beta e_{\mathrm{adj},i}^2.\end{aligned}$$

The above equation shows the link between our method and the equilibrium propagation. One can expect that when $\beta \to 0$, $\Delta k_{\mathrm{EP},i} \to \Delta k_i$.

In the Supplementary Figure 3a, we show the gradient of the loss function $\mathcal{L} = (u_{Ly} + 0.025)^2$ *using the in situ backpropagation, which is the same as the third panel of Fig. 1c. In the Supplementary Figures 3b and c, we present the approximate gradient by EP when the nudging strength are* 10^{-3} *and* 1*, respectively. From the pattern of the gradient obtained from our method and EP, they are similar, especially when the nudging strength is small. However, when the nudging strength becomes large, the value of the gradient has significant change (see the limits of the colorbar in Supplementary Figure 3c). We also quantify the gradient error from EP as a function of the nudging strength, where the linear growth of the gradient error can be observed in the log-log graph.*"

The revised Supplementary Figure 3 is attached as below:



3. The name in-situ backpropagation is also a bit confusing, because while I believe a spring network realizing the learning rule of Eq. (6) could be built in principle, the authors simulate this learning rule on a computer running a finite element model of a spring network. In fact, I think the statement in the discussion that "the experimental feasibility... had been demonstrated in Fig

1" is misleading, because the learning rule was definitely not demonstrated in the experimental substrate itself. The authors mention in the discussion "numerous experimental avenues" for implementing the learning rule, listing systems with variable spring constants. I would appreciate if the authors discussed this a bit more, specifically if they see any particularly promising way of realizing such a learning network. This is quite important, because I do not think the computational implementation of such learning rules give much, if any, benefit in producing designed networks compared to standard gradient descent (see more in point 4 below). For example, while the authors obtain nice matching between their computational models and the experimental ones, there is still remaining discrepancy (aptly named by Wright, Onodera, et. al., Nature 2022, the "simulation-reality gap"). This discrepancy will likely get worse for larger networks or small components, where fabrication at scale becomes error prone. In this context, it would be good to acknowledge prior work of Pashine, Phys. Rev. Mat. 2021, which experimentally implemented training using a local contrastive learning rule to prune an elastic network, in which the evaluation and execution of the update was done "manually" in the experiment, overcoming the simulation-reality gap.

We thank the reviewer for the comments and suggestions. To our understanding, the term "in situ backpropagation" refers only to the algorithm for computing the gradient in the identical mechanical network represented by stiffness matrix D (Eq. (6) in our paper), not how the gradient is used to update the learning degree of freedom (Eq. (7) in our paper). In our work, we demonstrated experimentally how the gradient can be derived directly from local information of the mechanical neural networks (MNNs), as illustrated in Fig. 1.

Agreeing with the reviewer's comment, we acknowledge that, due to current limitations in our experimental platform—specifically, the inability to automatically adjust the spring constants—the step of updating the bond widths could not be implemented experimentally yet. Thus, we list some possible experimental platforms and following the suggestions of the Reviewers #1 and #2, we elaborate more details about the implementation on the programmable devices. In addition, the work of Pashine et al. (Phys. Rev. Mat. 2021), which experimentally implemented training using a local contrastive learning rule to prune an elastic network, is particularly relevant. In their work, the evaluation and execution of the update were conducted manually, effectively addressing the "simulation-reality gap" that Wright, Onodera, et al. identified. We acknowledge this important contribution and have referenced it in our revised manuscript. We also follow the Review #2's suggestion to propose a possible implementation based on the programmable device.

The revised manuscript in page 16 is attached as below:

"*At present, the learning process (i.e., the step of changing the spring constants according to the obtained gradients) does not involve the real MNNs because the MNNs have not been physically realized to update themselves. Instead, the update of spring constants is conducted numerically by the bar model (Eq. (7)), but the capabilities of the trained MNNs are validated through experiments.*"

"*...where material properties can be programmable in situ by external fields, hold promise for*

facilitating further experimentation with in situ backpropagation."

The revised mansucript in page 17 is attached as below:

"*Another important issue that affects the physical implementation of MNNs is the gap between the simulated model and the real materials system, including the manufacturing tolerances and the bias in parameter updates during training caused by the discrepancy on the gradient [20]. One proposed scheme is via local pruning rules that allow the manipulation of material response by pruning bonds of disordered networks in situ and apply to more complex networks than the linear spring networks [57]. This approach opens up ways to build MNNs with desired behaviors and machine learning functions, without relying on simulation models on computers, overcoming the simulation-reality gap.*"

"*Our method opens the possibility to build a pure experimental setup of trainable MNNs without a central computer. This can be done using a general programmable device that tunes the spring constant using a microcontroller for experimental realizations without computer simulations. According to our learning rule, input force is applied to the device, and node displacements and elongation (e) are read by sensors. The microcontroller calculates the Jacobian of the loss from the output node displacements and applies an equivalent adjoint force to the device, with (e_{adj}) read by sensors. The microcontroller then performs element-wise multiplication of e and e_{adj}, conducting the update of the spring constant based on the calculated gradient. As discussed above, these calculations are of very low time complexity and feasible for microcontrollers without a central computer. This pure experiment setup can also avoid simulation-reality gap and realize retrainability.*"

---

4. The authors compare their computational evaluation of the gradient to that obtained by finite differences, Eq. (8), and show their computational implementation of in situ backpropagation is superior in terms of the error produced by the calculation, related to an additional free parameter δk associated with finite differences. Further, the authors correctly claim that in situ backpropagation requires two computations of the system configurations, while the finite differences requires a number of computations that scales with system size. This comparison may give a false impression that in situ backpropagation is computationally superior to direct computation of the gradient. In fact, one can (at least in principle) use automatic differentiation to evaluate these gradients due to a change in some parameters (as routinely done in machine learning, and increasingly in condensed matter physics simulations). Evaluating the gradient with automatic differentiation requires essentially one evaluation of the system configuration, so it will not scale any worse than Eq. (6), and will not introduce the errors described by the authors for finite difference evaluations.

   We thank the reviewer for this timely reminder of automatic differentiation. Indeed, we agree with the reviewer that automatic differentiation is another efficient method to compute gradient and widely used in machine learning. We thereby add a brief mention of automatic differentiation in the manuscript to clarify this point.

The revised manuscript in page 8 is attached as below:

"*To provide a comparison with conventional numerical method, we calculate the approximate gradient using the forward difference as follows:*"

The revised manuscript in page 9 is attached as below:

"*It is worthwhile to mention here that automatic differentiation featured by forward differentiation is also proven to be efficient in training computer-based and optical neural networks [41,43].*"

---

5. Some clarifications on the training process would be useful. In particular, the spring constants are presumably non-negative numbers. What happens in the algorithm when negative spring constants are requested? Is it important that the actual gradient of the cost function is no longer feasible experimentally?

We thank the reviewer for the comment and suggestion. As described in the Methods and Supplementary Information, to ensure non-negative spring constants, we employ a projection scheme that sets the corresponding bounds of the spring constant using a modified Sigmoid function, which ensures the non-negative values. It is expressed as:

$$k = k_L + \frac{k_U - k_L}{1 + e^{-\eta v}}$$

Here, k_L and k_U represent the lower bound and upper bound of the spring constant. v serves as the actual training parameter during the learning process and η controls the strength of projection varying from 1 to 10 depending on different cases. Actually, for the purpose of better fabrication and experiments, we set the range of width of the bonds to be from 1.5 mm to 2.5 mm, and thus the corresponding $k_L = 103.125$ N/m, $k_U = 171.875$ N/m. We also revised the manuscript by adding this information in the main text.

The revised manuscript in page 10 is attached as below:

"*During the training process, to keep the bond width in a certain range from* 1.5 mm *to* 2.5 mm *for the purpose of fabrication and following experiments, we use a projection scheme by a modified Sigmoid function, as detailed in Methods and Supplementary Information. This strategy also applies to the machine learning tasks in the following sections.*"

The revised manuscript in page 19 is attached as below:

"*For experimental purposes, the width of each bond is restricted to range from* 1.5 mm *to* 2.5 mm *through the projection scheme enabled by a modified Sigmoid function in the simulation.*"

---

6. It is not clear what is meant by "the noncontroversial nature of input force and output displacement". Is it that they are easily distinguishable? Could you also train the input and output

on the same node by, say, having the input be horizontal displacement and the output vertical displacement?

We thank the reviewer for the comment. Based on the governing equation of statics $Du = F$, the input is the force (F) and the output is the displacement (u). Because the input force and the output displacement are different physical quantities, they can be defined on the same node and do not affect each other during the training. Specifically, the input force applied to a node does not directly conflict with the displacement observed at the same node. Regarding the question about having horizontal displacement as input and vertical displacement as output, the current framework of our method strictly considers the input as a force and the output as a displacement. Therefore, while the output can be a target vertical displacement, the input cannot be a prescribed horizontal displacement (but it can be a prescribed horizontal force). Also considering a similar comment from Reviewer #3, we revised our manuscript by clarifying these points.

The revised manuscript in page 9 is attached as below:

"*In addition, another advantage of our MNNs is the potential utilization of the same node as both input and output node, owing to the force and displacement being different physical quantities that can be defined on the same node.*"

---

7. In Figure 2, I think it is sufficient to show one scenario with a-symmetric displacement. The other is trivially symmetric to the first one, and displaying it confused me as to whether the same system has two different behaviors (which is unfortunately not the case).

We thank the reviewer for the suggestion. We follow the reviewer's suggestion to display only one scenario in the Fig. 2. The other symmetric scenario, which is trivially related to the first one, has been moved to the Supplementary Figure 4.

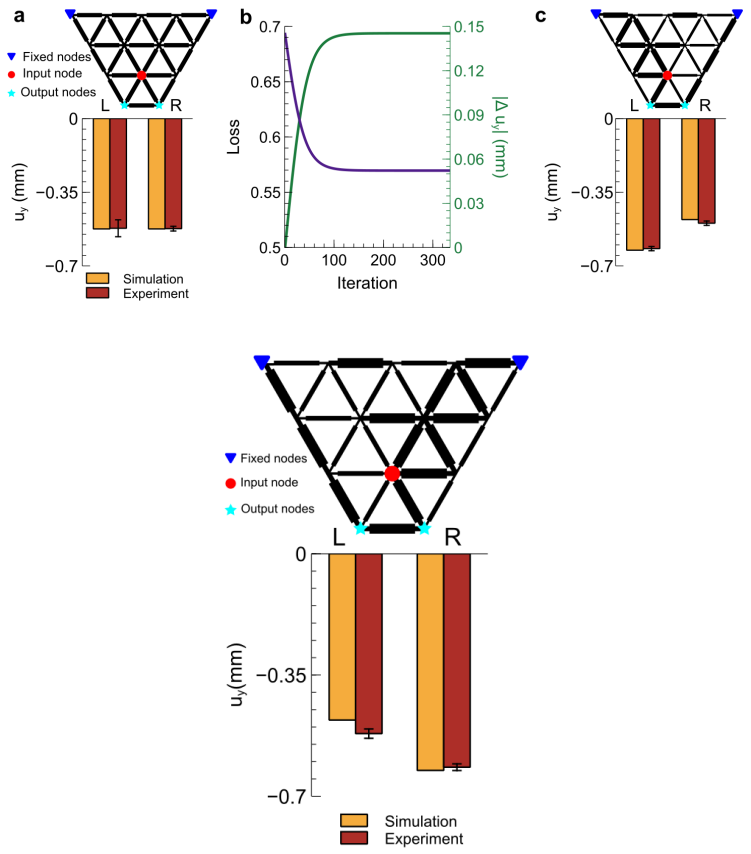
The revised manuscript in page 10 is attached as below:

"*On the contrary, in Supplementary Figure 4, another trained MNN presents an alternate scenario where the node on the right displays a greater displacement under the same input force.*"

The revised Supplementary Information in page 5 is attached as below:

"*In Supplementary Figure 4, we show the case where the right node develops larger displacement than the left node does. The trained MNN is mirror symmetric to the trained MNN in Fig. 2c, which has opposite behaviors under the same force.*"

The revised Figure 2 and Supplementray Figure 4 are attached as below:



Reviewer #2

The authors present a proof-of-concept study on a limited form of backpropagation in linear mechanical systems. Overall, the core scientific content is great and the authors' beautiful experimental and theoretical results stand as great contributions on their own. Conversely, I found the paper to be written in a way that felt misleading or imbalanced - positive aspects felt oversold, and the many limitations of the work felt like they were not mentioned, or were mentioned only in the slightest detail. I can easily imagine the paper in its current form being frustrating for many readers - this is a shame because the core content is brilliant!

We thank the reviewer for the time spent in reviewing our manuscript and for the positive comments to our work. We also appreciate the reviewer's suggestions about discussions on the limitations of our work. In the following, we address the reviewer's comments point by point, to the best of our ability, and we have revised our manuscript accordingly. Regarding the annotations on the PDF file, some are covered in the following comments as they are related to points in the numbered list from the reviewer, and others are listed separately in the end.

Here are my comments, many also in the attached PDF

1. In the introduction the case for what the authors describe as MNNs compared to optical or wave-

based PNNs is vague or unconvincing. The case for MNNs being useful beyond computation is made reasonably well. I recommend the authors consolidate these arguments somewhere, and consider which ones are convincingly justified and which ones are simply too vague. The impact and weight of the paper will be vastly improved by cutting the "fluff".

We thank the reviewer for the comment and suggestions. We also notice that there are some related annotations in the Introduction in the PDF file in page 2, "This is not compelling because it is too vague", "Explain this more". We have revised the manuscript to discuss in more specific ways the advantages and limitations of our proposed method.

There are two major advantages as MNNs are compared to optical or wave-based PNNs. One is that MNNs show better resistance to complex electromagnetic environment compared with optical counterparts. The other is that the implementation of MNNs in the statics regime can avoid multiple vibration modes in the dynamics regime (a challenge in modeling of the real materials system). Also, we think that MNNs in the statics regime offer the ease of operation in experiments and real-world applications, but the Reviewer #3 suggests to remove it. Therefore, considering both comments, we revise the manuscript and hope that it is now clearer and more convincing.

The revised manuscript in page 2 is attached as below:

"*Wave-matter interactions are commonly utilized in optical neural networks to implement machine learning, employing mechanisms including diffraction and the equivalence between recurrent neural networks and wave physics [14,23]. Similar ideas are used to establish the learning framework in MNNs [21,24]. Another proposed method uses a vibrating plate, activated by acoustic waves, to function as both input and output. Instead of training the neural network by adjusting weights directly, this approach employs electrically generated masking signals from interfering acoustic waves for the training process [19]. This idea has been expanded by adding multiple layers of vibrating plates, thus forming a deep physical neural network [20]. MNNs are expected to show advantages in challenging environments with complex electromagnetic conditions where optical counterparts may underperform.*"

The revised manuscript in page 3 is attached as below:

"*Despite the effective wave-based implementations, wave dynamics is complex involving multiple resonant modes in real materials systems, leading to challenges in the dynamical modeling of the physical systems, which may cause large simulation-reality gap. In contrast, the physical change induced in MNNs under static forces offer a promising solution to address these challenges, which has been demonstrated to learn behaviors (design desired response of materials under load) [25,26]. However, the approach using optimization algorithms (e.g., genetic algorithms) operated on computers ultimately relies on conventional digital processors. Therefore, challenges exist in terms of efficient training methods that rely on physical processes and corresponding experimental demonstrations.*"

2. There are several steps of the learning algorithm that require a digital computer, but this was never adequately explained, and I felt at times like the authors might be attempting to divert my attention from a fundamental issue. Overall while it is likely unintentional, I really did not like how the paper addressed this aspect. In a revised version the authors should clearly sketch out - in the main paper and likely the first figure - what is done physically and what must still be done with a conventional computer. For the parts done physically and with a conventional computer, the computational cost (e.g., scaling with number of parameters) should be written down. It should be noted not only that a digital computer is required to compute part of the total gradient, but also that it is necessary to measure the elongation, and to actually perform the update of the spring constant (assuming a device that has programmable springs)

   We thank the reviewer for the comment. We also notice that there are some related annotations in the Introduction in the PDF file in pages 3 and 5, "misleading on essentially without a computer", "we need to be able to measure e_{adj}", "Done digitally. This is ok but need to be clear".

   We have tuned down the claim as detailed below, but we would like to also provide some clarifications on our original statement of "essentially without a computer." Our learning rule indicates that the gradient of the loss is the element-wise multiplication between adjoint elongation and forward elongation, where two types of elongations can be obtained directly from the physical system. There are two parts that need computation in this process. One is the Jacobian of the loss function, whose values serve as the value of the adjoint force to induce adjoint elongation. The other is the element-wise multiplication between adjoint elongation and forward elongation, and are now clearly marked in Fig. 1. Given the simple form of the commonly-used loss function such as mean squared error and cross-entropy loss, and a limited number of output nodes, the sparse Jacobian can be easily calculated with the computational cost of $\mathcal{O}(dn_{out})$ (n_{out} is the number of the output nodes). Besides, the element-wise multiplication is an elementary arithmetic operation with the computational cost $\mathcal{O}(m)$ (m is the number of the bonds). Given that the analytic form of the Jacobian can be calculated at the beginning of training, and later steps of computing the values of the Jacobian and the multiplication could be done using simple circuits at each bond, we refereed to this process as "essentially without a computer" in our manuscript. Of course, we agree that using a computer is more practical for these calculations at the current stage. Also considering the suggestion made by Reviewer #3, we revised it to "using local rules".

   The measurement of the elongations is another step where we have used conventional computers in the experiment, which is not "essential" but a practical step in our experiment. Here, different experimental systems may have different computational cost. For example, in our work, we utilize the cross-correlation to track the nodes using the pictures of MNNs, where the computational cost depends on the size of the subset and search area, typically $\mathcal{O}\left((M-P+1)(N-Q+1)PQ\right)$ with the size of the image subset $P \times Q$ and the size of the search area $M \times N$. But if other experimental setups, say, sensors measuring the elongation

are attached on the MNNs and can read out corresponding data directly, an external computer will not be needed, and the computational cost can be $\mathcal{O}(m)$ (m is the number of the bonds).

We revised our manuscript by making this clear and marking these procedures in Fig. 1.

The revised manuscript in page 4 is attached as below:

"*By using 3D-printed MNNs, we demonstrate the feasibility of obtaining the exact gradient of the loss function experimentally solely from the bond elongation of MNNs in only two steps, using local rules.*"

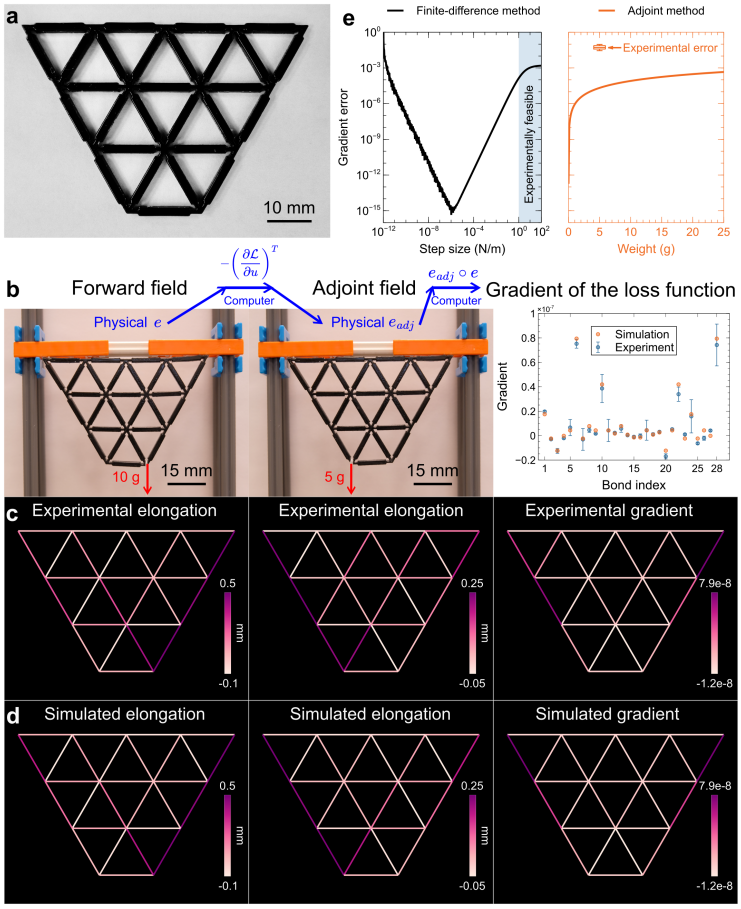
The revised manuscript in page 6 is attached as below:

"*(2) Calculate* $-\left(\frac{\partial \mathcal{L}}{\partial u}\right)$ *given the form of the loss function* $\mathcal{L}(u)$ *using the displacement in step (1) with computational cost* $\mathcal{O}(dn_{\text{out}})$ *in digital computers. Note that only nonzero entries in* $-\left(\frac{\partial \mathcal{L}}{\partial u}\right)$ *are calculated corresponding to the output nodes, e.g.,* $\mathcal{L} = (u_p - u_T)^2$ *for letting displacement of node* p, u_p, *close to desired displacement* u_T, *the only nonzero entry for the adjoint forces will be* $2(u_T - u_p)$. *Apply the force* $-\left(\frac{\partial \mathcal{L}}{\partial u}\right)^T$ *to the same system to obtain the adjoint elongation of the bonds* e_{adj}. *The gradient is the element-wise multiplication of the forward elongation and the adjoint elongation with computational cost* $\mathcal{O}(m)$ *where* m *is the number of bonds.*"

The revised manuscript in page 18 is attached as below:

"*To obtain the elongation of the bonds in mechanical networks, we employ correlation-based algorithms[57-59], to track the centers of the joints with the computational cost* $\mathcal{O}\left((M-P+1)(N-Q+1)PQ\right)$, *where the size of the image subset is* $P \times Q$ *and the size of the search area is* $M \times N$."

The revised Figure 1 is attached as below:



3. The MNNs used by the authors are introduced in a general way, but they make a number of assumptions implicitly that may or may not be obvious to the broad audience the paper is attempting to reach. Unless there is some paper the authors can cite which introduces a canonical definition of this type of mechanical neural network, they should take more care in introducing what the physical system is (i.e., what class of systems, with some concrete examples, the theoretical model describes) and what it neglects (e.g., nonlinearity, dissipation).

   We thank the reviewer for the useful suggestion. We also notice that there are some related annotations in the Introduction in the PDF file in page 3, "What does this mean?", "I feel like we need to define MNN in terms of spring networks first", "How are D and K connected?" and "Were you not already there on the linear regime?".

   The MNNs in our work refer to as a network with n nodes and m linear springs in d dimension. We also ensure that MNNs do not have zero mode under the applied boundary condition, meaning the compatibility matrix has full rank, in order to avoid the zero elongation under the applied force. Besides, the stiffness matrix D related to the diagonal matrix K in a way of $D = C^T K C$.

   The revised manuscript in page 4 is attached as below:

"*Here we have n nodes embedded in d-dimensional space, located at position* $\{x_j\}$. *The numbers of input and output nodes are* n_{in} *and* n_{out}, *respectively. The nodes are connected by m linear springs, each of which has a spring constant* k_i. *Also, zero modes are prohibited by properly designing the network connectivity such that the compatibility matrix C is fully ranked, which will be introduced later. Compared with the typical computer-based neural networks with layer-by-layer structures, MNNs are regarded as entire networks.*"

The revised manuscript in page 5 is attached as below:

"$D \in \mathbb{R}^{dn \times dn}$ *is the stiffness matrix which is symmetric (also know as the Hessian matrix or the dynamical matrix in different contexts)*"

"*Then, we utilize the compatibility matrix [39,40],* $C \in \mathbb{R}^{m \times dn}$ *that maps the node displacement u to the bond elongation e in the linear regime, such that,* $e_i = \hat{I}_{j_1 j_2} \cdot (u_{j_1} - u_{j_2})$, *where* $\hat{I}_{j_1 j_2}$ *is a unit vector pointing from node* j_1 *to node* j_2, *which determines each entry in C. Besides, the stiffness matrix D can be decomposed into* $C^T K C$, *where K is the diagonal matrix with k as the diagonal entries.*"

---

4. I felt like the authors failed to directly address most of the relevant literature. The work from Murugan's group or the Penn groups covers very similar topics and it is important that the authors clearly explain how their work goes beyond this (which includes but is not limited to the experimental demonstrations in 3D-printed mechanics). Ref. 19 is also very relevant, but is hidden in a big "lump citation". I know how easily this can happen during the writing process, but to a skeptical reader this feels a little "off".

We appreciate this suggestion and the significant contributions from other groups. The major highlight of our work is to propose a new method, in situ backpropagation, to train MNNs and conduct the experiment using 3D printed MNNs. We revised the manuscript by citing more relevant works, discussing them in details and highlighting the novelty of our works.

The revised manuscript in page 2 is attached as below:

"*Another proposed method uses a vibrating plate, activated by acoustic waves, to function as both input and output. Instead of training the neural network by adjusting weights directly, this approach employs electrically generated masking signals from interfering acoustic waves for the training process [19]. This idea has been expanded by adding multiple layers of vibrating plates, thus forming a deep physical neural network [20].*"

The revised manuscript in page 3 is attached as below:

"*Therefore, challenges exist in terms of efficient training methods that rely on physical processes and corresponding experimental demonstrations.*"

"*So far, the MNNs based on the* physical learning *have been developed using the platform of origami structures [28,36] and disordered networks[29,37] to demonstrate machine learning*

through simulations. The experimental proposals involve using directed springs with variable stiffness[38] and manually adjusting the rest length of springs[31]."

"*Here, we present a highly-efficient training protocol for MNNs through mechanical analogue of in situ backpropagation, derived from the adjoint variable method, in which the learning only relies on the local information. In contrast to coupled learning, backpropagation is a well tested method in machine learning, and has the advantage of producing* the exact gradient of arbitrarily chosen loss function[1]. *We introduce backpropagation to MNNs and experimentally demonstrate this method.*"

The revised manuscript in page 17 is attached as below:

"*One proposed scheme is via local pruning rules that allow the manipulation of material response by pruning bonds of disordered networks in situ and apply to more complex networks than the linear spring networks[57]. This approach opens up ways to build MNNs with desired behaviors and machine learning functions, without relying on simulation models on computers, overcoming the simulation-reality gap.*"

---

5. With respect to this paper's relation to prior work, it deserves to be mentioned the relationship with coupled learning and equilibrium propagation. Is this technique actually materially different from those techniques? This may not be as obvious as it seems. I can certainly see that it has been derived in a different way but even after reading the paper in full, I admit I am not that sure how much these algorithms actually differ in terms of what they entail in practical implementation.

We thank the reviewer for raising this great question. First of all, the contrastive learning framework, such as EP and coupled learning, utilizes two states (a free state in equilibrium and a perturbed state with a new equilibrium slightly away from the original equilibrium) to approximate the gradient via the difference between them. In comparison, our method introduces two completely different equilibrium states, which are induced by forward forces and adjoint forces, respectively. Two equilibrium states can be far away from each other, as the input forces are absent in the adjoint step. These two equilibrium states can be understood by a forward pass sending input signal (forward force) and a backward pass sending error (adjoint force). This is the fundamental difference between our method and contrastive learning framework.

Second, algorithmically, in contrastive learning the difference between two states produces the approximated gradient affected by the nudging parameter. Whereas in our method, the element-wise multiplication of elongations in two states produces the exact gradient. We derive a relation between our approach and EP in the Supplementary Information that $\Delta k_i = \Delta k_{\mathrm{EP},i} + \frac{1}{2}\alpha\beta e^2_{\mathrm{adj},i}$. One can expect that when $\beta \to 0$, EP approaches our method.

Third, although the coupled learning is demonstrated in some works, the loss function cannot be defined as an arbitrary function. Besides, a very recent paper (Scellier et. al., NIPS, 2024) points out and proves that coupled learning rules neither perform gradient descent

on $\mathcal{L} = \frac{1}{\beta}(E_{nudge} - E)$ nor optimize the mean squared error $\mathcal{L}_{MSE}$. Whereas our method unambiguously performs the gradient descent on the loss function.

The revised manuscript in page 3 is attached as below:

"*In light of this, the concept of* physical learning *offers a promising avenue to train MNNs only based on local information of the networks, inspired by neuroscience[27]. This methodology is promising for experimental implementations where the learning occurs physically in the system. One method, termed coupled learning, uses the contrast of two equilibrium states (free and nudged states) to update the system but cannot define the arbitrary loss function[28-31]. This method has enabled MNNs to perform well in supervised machine learning tasks. Each time a training example of a force pattern is applied to the MNN, it updates itself according to the learning rule. Over time, the MNNs learn to accurately respond to unseen forces that have spatial correlation patterns resembling those in the training examples.*"

"*Another well-known method for* physical learning *is Equilibrium Propagation (EP), sharing similar procedure with coupled learning but being able to define the arbitrary loss function[32]. This method has been demonstrated in various physical systems such as nonlinear resistor networks[33], Ising machines[34] and coupled phase oscillators[35].*"

The revised manuscript in page 6 is attached as below:

"*Similar to the energy-based learning methods, such as EP and coupled learning described in the Introduction, our method also introduces two equilibrium states induced by the forward force F and the adjoint force $-\left(\frac{\partial \mathcal{L}}{\partial u}\right)^T$. However, in energy-based learning methods, the nudged states are slightly away from the free states controlled by the nudging strength, whereas the two equilibrium states in our method are* independent*, where the input forces are absent in the second state. Besides, algorithmically, the difference between two equilibrium states approximates the gradient in energy-based learning methods but our method yields gradient through the multiplication, performing* backpropagation *in the MNN. In the Supplementary Information, we show the link between EP and our method.*"

The revised manuscript in page 9 is attached as below:

"*We also conduct the EP to estimate the gradient of $\mathcal{L} = (u_{Ly} + 0.025)^2$ and compare the results with ours, as described in Supplementary Information and Supplementary Figure 3. When the nudging strength is small, the results from our method and EP are almost identical. But with the increase of the nudging strength, the gradient error becomes large.*"

The revised Supplementary Information in page 4 is attached as below:

"***Comparison between our method of in situ backpropagation and equilibrium propagation.*** *Equilibrium propagation (EP) is a kind of energy-based learning framework [1]. Compared with other energy-based learning algorithm, such as contrastive learning and coupled learning, EP explicitly introduces the loss function $\mathcal{L}$ to represent the difference between the output and the desired output [2]. The implementation of EP for the MNNs has two phases. In the first phase, similar to our method of in situ backpropagation, the input forces*

are applied to the MNNs to reach the equilibrium state. In the second phase, EP proceeds by nudging the output nodes using forces $-\beta\frac{\partial \mathcal{L}}{\partial u}$, where β is called the nudging strength. In what follows, we apply EP to our MNNs to approximate the gradient of the loss function. Given the elongation of MNNs in two phases, e and e_{nudged}, the update of spring constant is given by $\Delta k = \frac{\alpha}{2\beta}(e^2 - e_{\text{nudged}}^2)$, where α is the learning rate. Algorithmically, the element-wise multiplication between forward elongation and adjoint elongation produces the exact gradient, while the difference between two states approximates the gradient.

Now we show the link between our method and equilibrium propagation. The first phase is the same as that in our method, described by the governing equation:

$$Du = F.$$

The second phase requires a nudge on the output node, which can be expressed as:

$$Du_{\text{nudged}} = F - \beta \left(\frac{\partial \mathcal{L}}{\partial u}\right)^T.$$

Recall that the second phase for our method is:

$$Du_{\text{adj}} = -\left(\frac{\partial \mathcal{L}}{\partial u}\right)^T.$$

Therefore, in the linear regime, $u_{\text{nudged}} = u + \beta u_{\text{adj}}$ and $e_{\text{nudged}} = e + \beta e_{\text{adj}}$. The update rule for our method is:

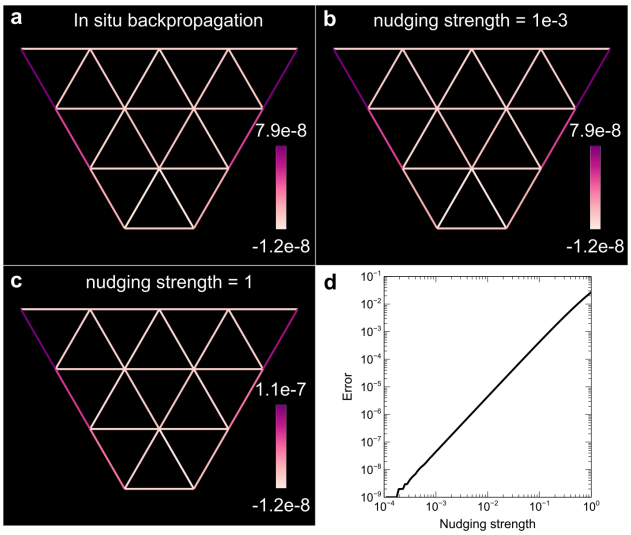
$$\begin{aligned}
\Delta k_i &= -\alpha e_i e_{\text{adj},i} \\
&= \frac{\alpha}{\beta}\left[\frac{1}{2}e_i^2 + \frac{1}{2}\beta^2 e_{\text{adj},i}^2 - \frac{1}{2}(e_i + \beta e_{\text{adj},i})^2\right] \\
&= \frac{\alpha}{\beta}\left[\frac{1}{2}(e_i^2 - e_{\text{nudged},i}^2) + \frac{1}{2}\beta^2 e_{\text{adj},i}^2\right] \\
&= \Delta k_{\text{EP},i} + \frac{1}{2}\alpha\beta e_{\text{adj},i}^2.
\end{aligned}$$

The above equation shows the link between our method and the equilibrium propagation. One can expect that when $\beta \to 0$, $\Delta k_{\text{EP},i} \to \Delta k_i$.

In the Supplementary Figure 3a, we show the gradient of the loss function $\mathcal{L} = (u_{Ly} + 0.025)^2$ using the in situ backpropagation, which is the same as the third panel of Fig. 1c. In the Supplementary Figures 3b and c, we present the approximate gradient by EP when the nudging strength are 10^{-3} and 1, respectively. From the pattern of the gradient obtained from our method and EP, they are similar, especially when the nudging strength is small. However, when the nudging strength becomes large, the value of the gradient has significant change (see the limits of the colorbar in Supplementary Figure 3c). We also quantify the gradient error from EP as a function of the nudging strength, where the linear growth of the gradient error

can be observed in the log-log graph."

The revised Supplementary Figure 3 is attached as below:



---

6. One practical issue is that the experiments are performed with a platform that is, as far as I understood, not exactly reprogrammable, or at least not in the rapid, spring-by-spring way that would be required to do a full training process. The authors must note this! The way the paper is written now, it is not actually obvious and, given the dearth of experimental details outside the SM, convincing oneself of this with certainty takes a longer time than should be necessary. The fact this works despite that is impressive, but it is notable as a deficiency of the proof-of-concept.

   We thank the reviewer for the comment, which is very helpful for us to understand readers' perspective and improve our presentation.

   In the original version, we wrote in the Conclusion, "It is important to note that the experimental feasibility of in situ backpropagation has been demonstrated as shown in Fig.1, where the error signal is backpropagated to each bond to obtain the gradient. Although the update of spring constants is conducted numerically in the current model, the capabilities of the trained MNNs are validated through experiments.". We can only update the spring constants in simulations, instead of in experiments. To make it clearer, we add more details about experiments and limitations in the main text.

   The revised manuscript in page 16 is attached as below:

   "*At present, the learning process (i.e., the step of changing the spring constants according to the obtained gradients) does not involve the real MNNs because the MNNs have not been physically realized to update themselves. Instead, the update of spring constants is conducted numerically by the bar model (Eq.(7)), but the capabilities of the trained MNNs are validated through experiments.*"

7. Relatedly, it would be beneficial for the authors to discuss routes to such a programmable device implementation, as well as generalizations to the case of nonlinearity.

We thank the reviewer for the suggestion. Since the major limitation of our experimental setup is that the spring constants in MNNs cannot be updated automatically according to the learning rule, a programmable device such as a bar system with a micro-controller would be a good idea. The detailed learning procedure is listed as below:

(a) The input force is applied to the programmable device as the input.

(b) The displacements of the nodes can be read out and elongation can be calculated by the micro-controller.

(c) The Jacobian of the loss is calculated with the displacements of the node by the micro-controller.

(d) The adjoint force equivalent to the Jacobian of the loss is applied to the programmable device.

(e) The displacements of the nodes can be read out and elongation can be calculated by the micro-controller.

(f) The element-wise multiplication is done by the micro-controller and according to the calculated gradient, the spring constant can be updated.

We revised our manuscript by adding the possible programmable device implementation in the discussion.

As for nonlinearity, the current governing equation cannot be used. We must start from the general governing equation of statics, where the net force of each node is zero. This can be solved via nonlinear optimization in the forward problem. If we follow the similar derivation in the main text, the mathematical tricks such as decomposition of D will fail, which leads to a much more complicated adjoint problem and learning rules. It means this method cannot be directly generalized to the nonlinear regime. Due to its distinction and complexity, it is beyond the scope of our current work and it guarantees another work to fully explore nonlinear scenarios. But, indeed, nonlinear case is important to deal with the nonlinear dataset, so we revised our manuscript by adding relevant discussions and outlook about nonlinearity in the main text. Besides, suggested by Reviewer #3, we discussed the nonlinear effect to the gradient error.

The revised manuscript in page 7 is attached as below:

"*So far, our learning rules apply to the linear regime strictly, indicating the utilization of small deformation of MNNs to function.*"

The revised manuscript in page 16 is attached as below:

"*..., where material properties can be programmable in situ by external fields*"

The revised manuscript in page 17 is attached as below:

"*Our method opens the possibility to build a pure experimental setup of trainable MNNs without a central computer. This can be done using a general programmable device that tunes the spring constant using a microcontroller for experimental realizations without computer simulations. According to our learning rule, input force is applied to the device, and node displacements and elongation (e) are read by sensors. The microcontroller calculates the Jacobian of the loss from the output node displacements and applies an equivalent adjoint force to the device, with (e_{adj}) read by sensors. The microcontroller then performs element-wise multiplication of e and e_{adj}, conducting the update of the spring constant based on the calculated gradient. As discussed above, these calculations are of very low time complexity and feasible for microcontrollers without a central computer. This pure experiment setup can also avoid simulation-reality gap and realize retrainability.*"

The revised manuscript in page 17 is attached as below:

"*Given the derivation of our learning rules based on the matrix decomposition and linear operation, the nonlinear case cannot be naturally generalized. Therefore, exploring the nonlinear regime of MNNs by using nonlinear materials and geometric nonlinearity presents an opportunity not only to theoretical interest, but also to tackle nonlinear datasets and tasks, which is worth exploration in the future study.*"

The revised manuscript in page 8 is attached as below:

"*Although numerically the gradient in machine precision can be obtained, the assumption in the linear regime requires infinitesimal deformation, indicating that our method used in the experiment is always an approximation. To explore the validity of our method in experiments, we perform the nonlinear analysis numerically and show the gradient error as a function of the adjoint force represented by weights in Fig.1e. When the weight is small, the adjoint field is in the linear regime, where $e_i = \hat{I}_{j_1 j_2} \cdot (u_{j_1} - u_{j_2})$, leading to the small gradient error. However, when the weight becomes large, the adjoint field deviates from the linear regime, where $e_i = \|x_{j_1} + u_{j_1} - x_{j_2} - u_{j_2}\| - \|x_{j_1} - x_{j_2}\|$, resulting in the gradient error from 10^{-4} to 10^{-3}. This low gradient error in the large adjoint force implies that our method can produce gradient in high precision and efficiency. Experimentally, we choose the weight to be small enough for linearity and large enough to produce measurable displacements, leading to an error estimate shown in Figs.1b and e.*"

Annotations in the PDF file that are not covered by above comments are listed below:

---

8. Page 4: One sentence to justify this would help. Give typical computational complexity, e.g., on one example.

We thank the reviewer for the suggestion. We take the loss to be the mean square error $\mathcal{L} = (u_p - u_T)^2$, meaning the minimization of the displacement of the node p and u_T. The Jacobian

$\frac{\partial \mathcal{L}}{\partial u}$ is sparse and only the entries corresponding to the node p needs to be calculated. The computational complexity is $\mathcal{O}(d)$ (d is the spatial dimension, 2 in our case). In comparison, $\frac{du}{dk}$ is a $dn \times m$ matrix, whose computational complexity is $\mathcal{O}(dnm)$. We revised the manuscript by adding this example to justify our claim.

The revised manuscript in page 5 is attached as below:

"*For example, typically, the computational cost for $\frac{\partial \mathcal{L}}{\partial u}$ is $\mathcal{O}(dn_{\text{out}})$, but that for $\frac{du}{dk}$ is $\mathcal{O}(dmn)$.*"

The revised mnauscript in page 6 is attached as below:

"*Calculate $-\left(\frac{\partial \mathcal{L}}{\partial u}\right)$ given the form of the loss function $\mathcal{L}(u)$ using the displacement in step (1) with computational cost $\mathcal{O}(dn_{\text{out}})$ in digital computers. Note that only nonzero entries in $-\left(\frac{\partial \mathcal{L}}{\partial u}\right)$ are calculated corresponding to the output nodes, e.g., $\mathcal{L} = (u_p - u_T)^2$ for letting displacement of node p, u_p, close to desired displacement u_T, the only nonzero entry for the adjoint forces will be $2(u_T - u_p)$.*"

---

9. Page 4: No Italic for adj

We change u_{adj} to u_{adj} throughout our manuscript.

---

10. Page 4: Define e with math or define C. Hard to follow without this.

e is the elongation of the bond and C is the compatibility matrix that maps the displacement of the node to the elongation of the bond such that $Cu = e$. In the linear regime, the change in length of a bond, elongation, to the first order on the displacement is defined as $e_{ij} = \hat{I}_{ij} \cdot (u_i - u_j)$, where $\hat{I}_{ij}$ is a unit vector pointing from node j to node i. Each entry in the matrix C is determined by this equation. We revised our manuscript by defining e and C.

The revised manuscript in page 5 is attached as below:

"*Then, we utilize the compatibility matrix[39,40], $C \in \mathbb{R}^{m \times dn}$ that maps the node displacement u to the bond elongation e in the linear regime, such that, $e_i = \hat{I}_{j_1 j_2} \cdot (u_{j_1} - u_{j_2})$, where $\hat{I}_{j_1 j_2}$ is a unit vector pointing from node j_1 to node j_2, which determines each entry in C.*"

---

11. Page 4: Is there a simple intuition for this neat result?

From the perspective of the dimension, the dimension of gradient is the same as the dimension of the elongation of the bonds, so the intuitive guess is that the gradient is related to the elongation. More importantly, the form of the adjoint force $\left(-\frac{\partial \mathcal{L}}{\partial u}\right)^T$ represents the error signal. For example, $\mathcal{L} = (u - u_T)^2$ results in the nonzero entry of Jacobian being $2(u - u_T)$, which describes the difference between the prediction and the target. Therefore, the form of the adjoint problem represents the process of sending the error signal backwards in the networks, which has the similar spirit to the backpropagation.

The revised manuscript in page 6 is attached as below:

"*When we further think about the form of the adjoint forces* $-\left(\frac{\partial \mathcal{L}}{\partial u}\right)^T$, *where the nonzero entry is* $2(u_T - u_p)$, *it can be regarded as the error between the output and the desired output. Therefore, essentially our method provides two signal passes, a forward pass sending the input signal and a backward pass sending the error signal, reminiscent of the adjoint-method-based backpropagation in optical neural networks[14,22,41].*"

---

12. Page 5: Revise "two problems only have the difference on the input force" to "the two problems (forward and adjoint) differ in their forces"

    In the revised manuscript, we change it to "*the two problems (forward and adjoint) differ in their forces*" in page 5.

---

13. Page 5: Nice point but does this apply beyond 1-layer/linear?

    We consider the entire structure of the MNNs, denoted as stiffness matrix D in the derivation, so layers in MNNs do not matter and our learning rules still work. Our current learning rule is derived with the prerequisite of the linear regime, thus can only be applied in the linear regime. We respond in the previous response about the issue of nonlinearity, which is expected to be completely different from our current work. We revised the manuscript by making it clear.

    The revised manuscript in page 4 is attached as below:

    "*Compared with the typical computer-based neural networks with layer-by-layer structures, MNNs are regarded as entire networks.*"

    The revised manuscript in page 7 is attached as below:

    "*So far, our learning rules apply to the linear regime strictly, indicating the utilization of small deformation of MNNs to function.*"

    The revised manuscript in page 17 is attached as below:

    "*Given the derivation of our learning rules based on the matrix decomposition and linear operation, the nonlinear case cannot be naturally generalized. Therefore, exploring the nonlinear regime of MNNs by using nonlinear materials and geometric nonlinearity presents an opportunity not only to theoretical interest, but also to tackle nonlinear datasets and tasks, which is worth exploration in the future study.*"

---

14. Page 6: This is a bit too strong or needs more justifications. I think you mean the $\nabla \mathcal{L}_i$ is exact, not that there is no error overall. This is ok in context but want to be clear.

We thank the reviewer for the comment. Theoretically, the gradient is exact. However, like Reviewer #3 points out, every computation has a roundoff error, which we agree to. Once the mechanical system is simulated in the computer, roundoff error will always happen due to the machine precision. Also for the experimental implementation, the deformation is not infinitesimal. Combining the suggestions and comments made by Reviewer #3, we revised the manuscript by making it clear.

The revised manuscript in page 8 is attached as below:

"*Although numerically the gradient in machine precision can be obtained, the assumption in the linear regime requires infinitesimal deformation, indicating that our method used in the experiment is always an approximation. To explore the validity of our method in experiments, we perform the nonlinear analysis numerically and show the gradient error as a function of the adjoint force represented by weights in Fig.1e. When the weight is small, the adjoint field is in the linear regime, where* $e_i = \hat{I}_{j_1 j_2} \cdot (u_{j_1} - u_{j_2})$*, leading to the small gradient error. However, when the weight becomes large, the adjoint field deviates from the linear regime, where* $e_i = \|x_{j_1} + u_{j_1} - x_{j_2} - u_{j_2}\| - \|x_{j_1} - x_{j_2}\|$*, resulting in the gradient error from* 10^{-4} *to* 10^{-3}*. This low gradient error in the large adjoint force implies that our method can produce gradient in high precision and efficiency. Experimentally, we choose the weight to be small enough for linearity and large enough to produce measurable displacements, leading to an error estimate shown in Figs. 1b and e.*"

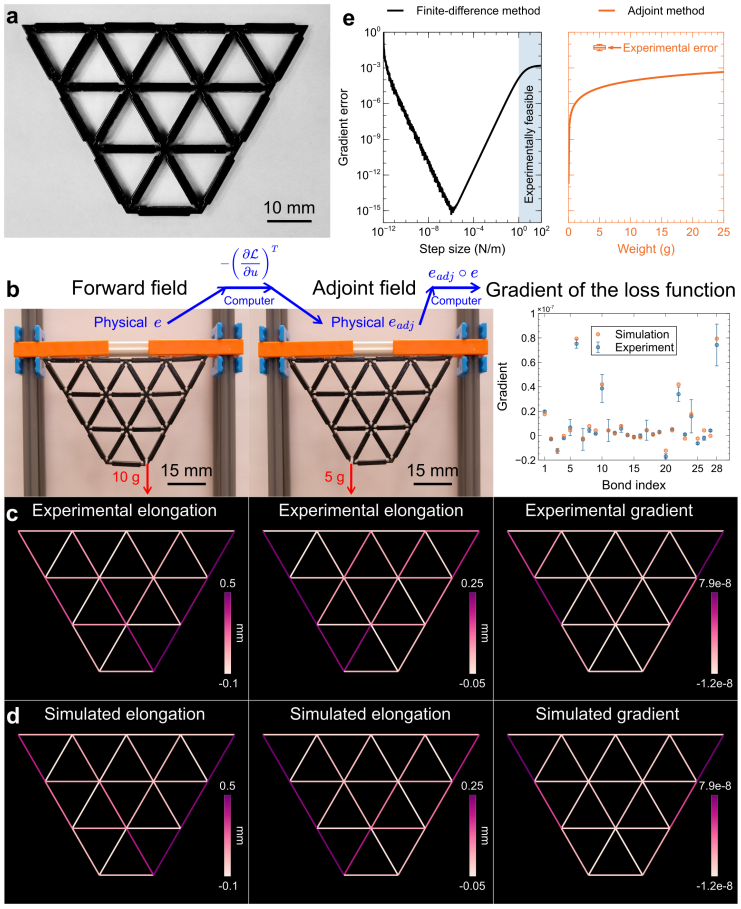
The revised manuscript in page 9 is attached as below:

"*Note that the roundoff error will also occur in our method, but thanks to the user-defined convergence condition, the minute adjoint force can be avoided.*"

The revised caption of Fig.1 is attached as below:

"*The numerical error of the adjoint method depends on the machine precision in the linear regime. The orange curve in the right panel represents the gradient error when the deformation is not infinitesimal and the response of the MNN is nonlinear.*"

The revised Figure 1 is attached as below:



15. Page 7: What does this mean? "owing to the noncontroversial nature of input force and output displacement"

Based on our learning rule and the governing equation of statics, the input is the force and the output is the displacement. Because the input force and the output displacement are two different physical quantities, the input and output can be defined on the same node and do not affect each other during the training. Combining the comments from Reviewer #1 and Reviewer #3, we revised the manuscript by making it clear.

The revised manuscript in page 9 is attached as below:

"*In addition, another advantage of our MNNs is the potential utilization of the same node as both input and output node, owing to the force and displacement being different physical quantities that can be defined on the same node.*"

16. Page 8: This is too vague to be effective.

The behaviors learning can also be regarded as the inverse design, where the materials and

structures are expected to have the desired behaviors under the applied force. Therefore, using our method of in situ backpropagation, we offer a straightforward methodology to train materials to have desired behaviors, thus impacting many fields in mechanical systems. We follow the reviewer's suggestion to make it clear and concrete.

The revised manuscript in page 10 is attached as below:

"*The example demonstrated above shows the MNNs can learn different behaviors under the applied force, where the utilization of in situ backpropagation offers a straightforward methodology. This technique simplifies the design process by providing the designated input and desired output. It can be used in creating advanced mechanical systems with desired functionalities. For example, automotive engineers could design cars with improved loading capability, while roboticists could develop robots with grasp and release behaviors under the control force.*"

---

17. Page 8: This is a sold claim I disagree with. Justify further or remove.

We agree with the reviewer to remove this sentence.

---

18. Page 11: This is true but sounds weird in the current context. Your results are amazing! You can get sway with "sale pitch" since they stand by themselves!

We thank the reviewer for the comment. We revised the manuscript by removing "remarkable".

---

19. Page 14: Equation would be appropriate. It is helpful to see a very general model still reduced to your spring model (not every physicist takes this for granted!)

We thank the reviewer for the suggestion. We use the general FEM bar model to derive the learning rule. The results show the elongation in FEM bar model is equivalent to the elongation in the reduced bar model. Therefore, the general FEM bar model can be easily reduced to our simplified model. We added this to the Supplementary Information.

The revised Supplementary Information in page 3 is attached as below:

"*Our reduced model by regarding each bond as a central-force bar improves the efficiency compared with FEM bar model with refined nodes. When the axial deformation is considered, the elongation in the linear regime is defined as:* $e_i = \hat{I}_{j_1 j_2} \cdot (u_{j_1} - u_{j_2})$*, where* $\hat{I}_{j_1 j_2}$ *is a unit vector pointing from node* j_1 *to node* j_2*. If* n *nodes are assigned on the bar, the elongation of*

$n-1$ segments can be expressed using $n-1$ equations:

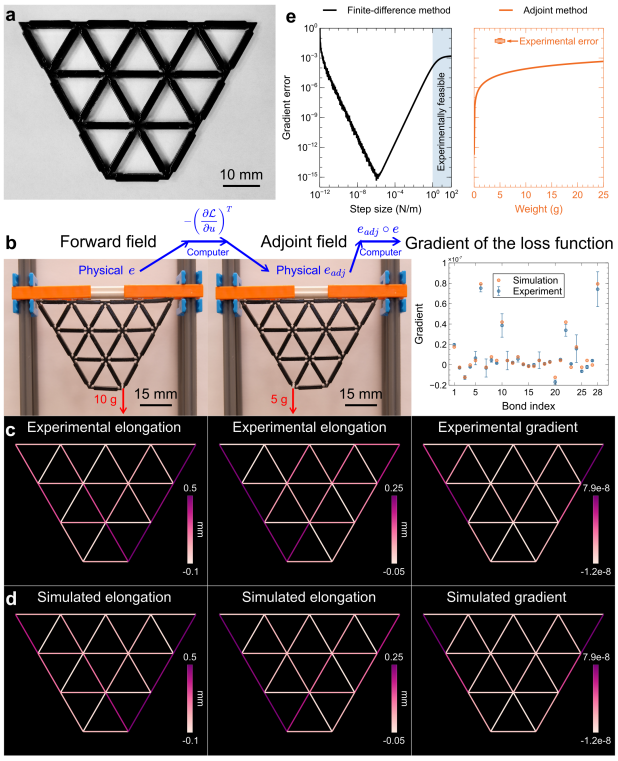
$$
\begin{aligned}
e_1 &= \hat{I}_{12} \cdot (u_1 - u_2) \\
e_2 &= \hat{I}_{23} \cdot (u_2 - u_3) \\
&\vdots \\
e_{n-1} &= \hat{I}_{n-1,n} \cdot (u_{n-1} - u_n).
\end{aligned}
$$

Note that $\hat{I}_{12} = \hat{I}_{23} = \ldots = \hat{I}_{n-1,n} = \hat{I}_{1n}$. Therefore, the summation of the equations above results in the $e_1+e_2+\ldots+e_{n-1} = \hat{I}_{1n} \cdot (u_1 - u_n)$. It indicates that the summation of elongations of all segments equals to the elongation of the entire bar, demonstrating the effectiveness of our reduced bar model to obtain the gradient through the bar elongation. "

---

20. Page 16: Beautiful plot but not easy to see agreement. Can you add one plot like this? This will emphasize amazing quantitative agreement.

    Good suggestion! We notice that Reviewer #3 also has similar suggestion. We revised the Figure 1 by plotting elongation and gradient of each bond using dots so that we can clearly see the agreement between experiments and our model.



---

21. Page 17: Annotate the nodes

We follow the reviewer's suggestion to label the nodes in all Figures.

---

22. Page 17: No noise? Is this experiment? No error bars?

This training in behaviors learning is not based on the dataset, so there is no different training dataset. Therefore, there is no variation of the loss or prediction (no noise and error bars), unlike the rest of the tasks. Besides, Fig. 2b is the simulation result but we test it upon the convergence of the loss, which are shown in Fig. 2c and Fig. 2d.

---

23. Page 17: Thickness $\propto k_{ij}$?

The thickness of the bond i is fixed to be 2 mm. We use the width of the bond to represent the spring constant, which is expressed as $k_i = \frac{Ytw_i}{L}$, where Y is Young's modulus, t is the thickness, w_i is the width and L is the length, as detailed in the Supplementary Information. From the equation, we can see $k_i \propto w_i$.

The revised manuscript in page 10 is attached as below:

"*The training process including loss decrease and bonds width change ($k_i \propto w_i$) is also shown in Supplementary Video* 1."

---

24. Page 19: For all figures, it would be helpful to have a small diagram showing how the test works, similar to Fig. 2a with annotations and math symbols.

We follow the reviewer's suggestion to add the annotations and math symbols to all relevant figures to clarify the input nodes, output nodes and fixed nodes.

---

25. Page 19: What is these points? Confirm from caption.

The triangle, square and circle symbols correspond to the representations of Iris flower in Figures 4a and 4c. We place the corresponding symbols on the top of the predicted results to show the successful classification. We follow the reviewer's suggestion to add the explanation in the figure caption.

The revised caption of Fig. 4 is attached as below:

"*The symbols on the top of the largest displacement correspond to the symbols in* $\boldsymbol{a}$*, representing the corresponding flower as the input.*"

Overall, I would like to recommend publication but in its current form there are many things that need to be changed (primarily just in the presentation/description) before I could do that. To some extent whether I recommend publication in Nature Communications or a different journal is contingent on how

some of the above issues are resolved, though I think it is more likely than not that I will recommend publication if the authors can realize a good-faith revision along these lines.

We deeply appreciate the reviewer for the positive comments and constructive suggestions. Given the amendments to the work induced by these comments and suggestions, we hope that the reviewer will now find this manuscript to be suitable for publication in Nature Communications.

Reviewer #3

Li and Mao conduct a hybrid simulation/experimental study in which they train simulated linear spring networks to perform a variety of tasks ranging from simple single-input tasks to the iris dataset (4 input 3 class classification task). They do so using an implementation of the adjoint method, which they derive and experimentally verify. This method is local, meaning that only direct measurements of a given piece of the network are required to evolve it in the proper direction to lower the given loss function. The final simulated networks are physically constructed and experimentally tested for several tasks. The authors show in simulation that such a mechanical system trained with this method can be retrained for multiple tasks, and additionally recover from physical damage if training is performed again. Overall, the study was interesting, and it was excellent to see experimental realizations of the simulations. However, the study lacked sufficient novelty for Nature Communications:

We thank the reviewer for the interest to our work and for the comment. The reviewer's criticism on the novelty of work appears to be based on the relation between our work and Equilibrium Propagation (EP). As detailed below, we thoroughly compared the two methods following the reviewers' suggestions, and show that our method is fundamentally different. We have also extensively revised our manuscript to clarify this important point. It is our hope that the revised manuscript now directly addresses the relations between our work and other existing methods, and thus the novelty is clear. In the following, we address the reviewer's comments point by point, to the best of our ability.

The proposed method is – I believe - identical to Equilibrium Propagation (Scellier and Bengio 2017), and this work was not cited or mentioned. If the methods are not the same, then a difference should be explicitly stated. If so, how does the work compare to implementations of EP (e.g. Laydevant et. al. Nature Communications 2024)?

We thank the reviewer for the comment. There are fundamental differences between EP and our method, despite some apparent similarities in the procedures. EP is a energy-based learning algorithm which explicitly introduces a loss function compared with other energy-based learning algorithms. In the first phase, input are clamped and the system reaches the equilibrium. In the second phase, EP proceeds by perturbing (or nudging) the output with a tunable nudging parameter. In this way, the system will reach a new equilibrium slightly away from the original equilibrium. By using the contrast between two equilibrium states, the gradient can be approximated and is affected by the nudging parameter. However, in our case, the system has two completely different

equilibrium states, which are induced by the forward forces and the adjoint forces, respectively. Two equilibrium states can be far away from each other, and the input forces are absent in the second state. The element-wise multiplication of elongations of two equilibrium states produces the exact gradient. Therefore, our method is new and has fundamental differences from EP. Furthermore, we derive a relation between our approach and EP in the Supplementary Information that $\Delta k_i = \Delta k_{\mathrm{EP},i} + \frac{1}{2}\alpha\beta e^2_{\mathrm{adj},i}$. One can expect that when $\beta \to 0$, EP approaches our method.

For the implementation of EP in the Laydevant et al., Nature Communications 2024, one needs write the total energy function in the form of $E_{tol} = E + \beta\mathcal{L}$, where E is the energy function of the physical system (Ising system, $E_{Ising} = \sum_{i>j} J_{ij}\sigma_i\sigma_j + \sum_i h_i\sigma_i$), β is the nudging parameter and $\mathcal{L}$ is the defined loss function. In the first phase, input are clamped and the system reaches the equilibrium. In the second phase, the nudge -$\beta\frac{\partial\mathcal{L}}{\partial\sigma}$ is added to the output. The update of the synaptic weights Δw is proportionate to $\frac{1}{\beta}\left(\frac{\partial E}{\partial w} - \frac{\partial E^{nudge}}{\partial w}\right)$. This is how the work mentioned by the reviewer implements EP in the Ising system. Our method is relatively simple. In the first phase, the input force is applied to the input, which is the same as EP. In the second phase, the adjoint force is applied to the output which has the same value as $-\frac{\partial\mathcal{L}}{\partial u}$. Note that we have another new equilibrium state induced by the adjoint force rather than a perturbation from the original state. And we do not need a nudging parameter to control the nudging strength. Instead, we have a exact value for the second phase. The update of the learning degree of freedom Δk is the element-wise multiplication between forward elongation and adjoint elongation $e_{adj} \circ e$, which is exact.

The revised manuscript in page 3 is attached as below:

"*Another well-known method for* physical learning *is Equilibrium Propagation (EP), sharing similar procedure with coupled learning but being able to define the arbitrary loss function[32]. This method has been demonstrated in various physical systems such as nonlinear resistor networks[33], Ising machines[34] and coupled phase oscillators[35].*"

The revised manuscript in page 6 is attached as below:

"*Similar to the energy-based learning methods, such as EP and coupled learning described in the Introduction, our method also introduces two equilibrium states induced by the forward force F and the adjoint force $-\left(\frac{\partial\mathcal{L}}{\partial u}\right)^T$. However, in energy-based learning methods, the nudged states are slightly away from the free states controlled by the nudging strength, whereas the two equilibrium states in our method are* independent*, where the input forces are absent in the second state. Besides, algorithmically, the difference between two equilibrium states approximates the gradient in energy-based learning methods but our method yields gradient through the multiplication, performing* backpropagation *in the MNN. In the Supplementary Information, we show the link between EP and our method.*"

The revised manuscript in page 9 is attached as below:

"*We also conduct the EP to estimate the gradient of $\mathcal{L} = (u_{Ly}+0.025)^2$ and compare the results with ours, as described in Supplementary Information and Supplementary Figure 3. When the nudging strength is small, the results from our method and EP are almost identical. But with the increase of the nudging strength, the gradient error becomes large.*"

The revised Supplementary Information in page 4 is attached as below:

"***Comparison between our method of in situ backpropagation and equilibrium propagation.*** *Equilibrium propagation (EP) is a kind of energy-based learning framework [1]. Compared with other energy-based learning algorithm, such as contrastive learning and coupled learning, EP explicitly introduces the loss function $\mathcal{L}$ to represent the difference between the output and the desired output [2]. The implementation of EP for the MNNs has two phases. In the first phase, similar to our method of in situ backpropagation, the input forces are applied to the MNNs to reach the equilibrium state. In the second phase, EP proceeds by nudging the output nodes using forces $-\beta\frac{\partial\mathcal{L}}{\partial u}$, where β is called the nudging strength. In what follows, we apply EP to our MNNs to approximate the gradient of the loss function. Given the elongation of MNNs in two phases, e and e_{nudged}, the update of spring constant is given by $\Delta k = \frac{\alpha}{2\beta}(e^2 - e^2_{\text{nudged}})$, where α is the learning rate. Algorithmically, the element-wise multiplication between forward elongation and adjoint elongation produces the exact gradient, while the difference between two states approximates the gradient.*
Now we show the link between our method and equilibrium propagation. The first phase is the same as that in our method, described by the governing equation:

$$Du = F.$$

The second phase requires a nudge on the output node, which can be expressed as:

$$Du_{\text{nudged}} = F - \beta\left(\frac{\partial\mathcal{L}}{\partial u}\right)^T.$$

Recall that the second phase for our method is:

$$Du_{\text{adj}} = -\left(\frac{\partial\mathcal{L}}{\partial u}\right)^T.$$

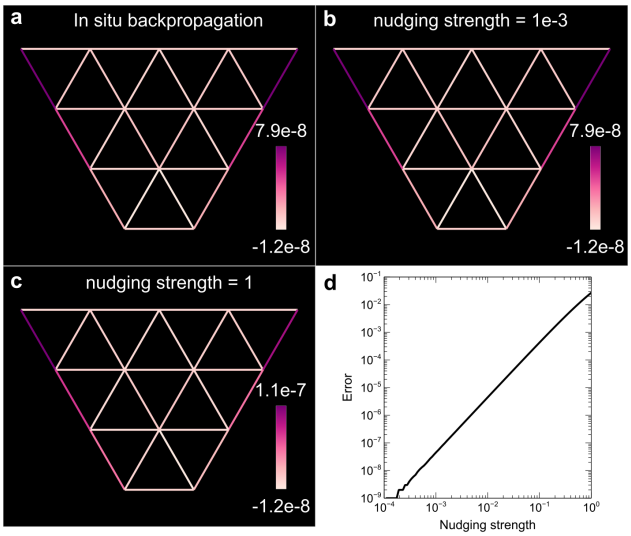
Therefore, in the linear regime, $u_{\text{nudged}} = u + \beta u_{\text{adj}}$ and $e_{\text{nudged}} = e + \beta e_{\text{adj}}$. The update rule for our method is:

$$\begin{aligned}
\Delta k_i &= -\alpha e_i e_{\text{adj},i} \\
&= \frac{\alpha}{\beta}\left[\frac{1}{2}e_i^2 + \frac{1}{2}\beta^2 e_{\text{adj},i}^2 - \frac{1}{2}(e_i + \beta e_{\text{adj},i})^2\right] \\
&= \frac{\alpha}{\beta}\left[\frac{1}{2}(e_i^2 - e_{\text{nudged},i}^2) + \frac{1}{2}\beta^2 e_{\text{adj},i}^2\right] \\
&= \Delta k_{\text{EP},i} + \frac{1}{2}\alpha\beta e_{\text{adj},i}^2.
\end{aligned}$$

The above equation shows the link between our method and the equilibrium propagation. One can expect that when $\beta \to 0$, $\Delta k_{\text{EP},i} \to \Delta k_i$.
In the Supplementary Figure 3a, we show the gradient of the loss function $\mathcal{L} = (u_{Ly} + 0.025)^2$ using the in situ backpropagation, which is the same as the third panel of Fig. 1c. In the Supplementary Figures 3b and c, we present the approximate gradient by EP when the nudging strength are 10^{-3}

and 1, respectively. From the pattern of the gradient obtained from our method and EP, they are similar, especially when the nudging strength is small. However, when the nudging strength becomes large, the value of the gradient has significant change (see the limits of the colorbar in Supplementary Figure 3c). We also quantify the gradient error from EP as a function of the nudging strength, where the linear growth of the gradient error can be observed in the log-log graph."

The revised Supplementary Figure 3 is attached as below:



The experiments in this work were replicas of the simulation (other works use local rules in mechanical networks in simulation, e.g. Stern et. al. PRX 2021), and as far as I could tell there was no feedback between the experiment and simulation. Further, the tasks in this work were extremely similar to those in Dillavou et. al. PR Applied 2022, which was an entirely experimental work.

We thank the reviewer for the comment. We demonstrated the step of in-situ backpropagation in experiment (c.f. updated Fig. 1 for components done physically and in computers), and found excellent agreement between simulations and experiments, which testifies our learning rules and validates our materials model. Because the learning process (i.e., the step of updating the spring constant) is demonstrated on the computer instead of the physical system, there is no dynamic feedback between experiment and simulation at this stage. However, we find the experimental results of the trained MNNs agrees excellently with the simulation and also analyze the experimental gradient error less than 10%. In the revised manuscript we discuss future perspectives of pure experimental implementations of our work.

Regarding the tasks, similar to the paper mentioned by the reviewer, we also choose the benchmark machine learning tasks to demonstrate the regression and classification, which has been widely used in computer-based neural networks and physical neural networks. For example, so far, the Iris flower classification dataset has been cited 352 times in different machine learning papers for demonstration according to UCI Machine Learning Repository. Moreover, since our network has a small size and limited learning degrees of freedom, the small classification dataset is considered,

which is unfortunately limited in the community of machine learning. We added another classification task, Penguin classification, in the Supplementary Information. Our MNNs also function well on this classification task, reaching the 90% accuracy for training set and 92% accuracy for testing set on average.

Compared with the paper on PR Applied, first, we propose a new learning rule, realizing *back-propagation* in MNNs, a well-tested method for machine learning in conventional neural networks, whereas the coupled learning rule is used in PR Applied paper.

Second, the nature of physical systems is different. Their work and our work can represent two different kinds of physical networks: Dillavou et. al. proposed a electric network which is essentially a scalar system, while we proposed a mechanical network which is a vector system. This can been seen from the difference of the degree of freedom of each node in two systems. For example, in the regression task, each node in our MNNs can represent two regression tasks (u_x and u_y, also see Eq.(9), even more regression tasks in higher dimension), but each node in electric networks can only represent one regression task regardless of the dimension. The matrices describing the connectivity are also different (the incidence matrix for electric networks and the compatibility matrix for mechanical networks).

Third, although the coupled learning is effectively used in PR Applied paper, a very recent paper (Scellier et. al., NIPS, 2024) points out and proves that coupled learning rules neither perform gradient descent on $\mathcal{L} = \frac{1}{\beta}(E_{nudge} - E)$ nor optimize the mean squared error $\mathcal{L}_{MSE}$. Our method unambiguously performs the gradient descent to minimize loss function. The difference on implementation of our method and the coupled learning can be seen in the previous response.

The revised manuscript in page 8 is attached as below:

"*We then define the gradient error as* $1 - g_t \cdot \hat{g}_t$*, where* g_t *and* $\hat{g}_t$ *are normalized measured gradient and exact gradient, respectively. Compared with the simulated gradient, which represents the exact gradient, our experimental gradient error is less than* 0.1*, as shown in the inset in Fig.1e.*"

The revised manuscript in page 14 is attached as below:

"*In addition, we demonstrate another classification task implemented on our MNNs, penguin classification[48], to further justify the capability, as detailed in Supplementary Information and Supplementary Figure 8. With the decrease of the loss, the accuracies for the training set and testing set increase to* 90% *and* 92%*, respectively.*"

The revised manuscript in page 16 is attached as below:

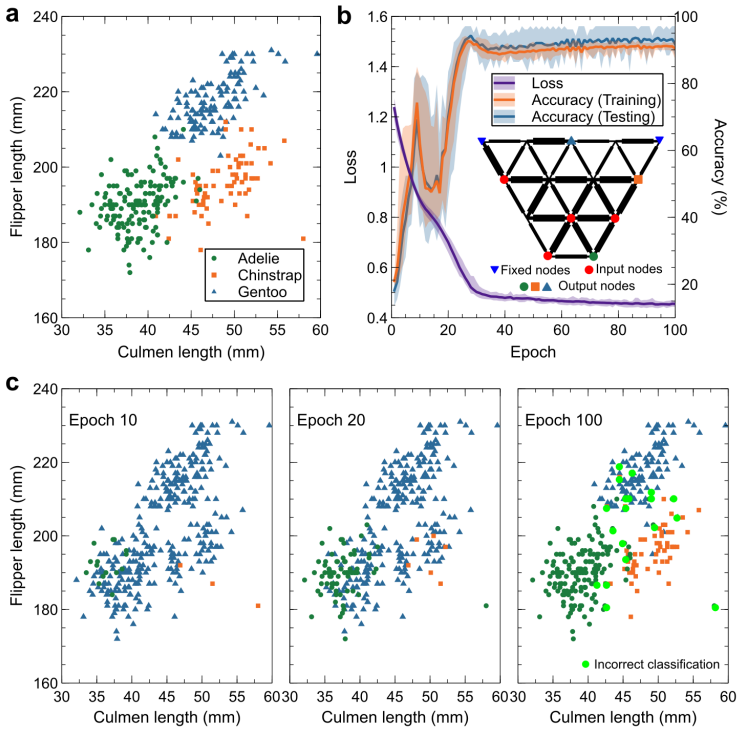
"*Besides, unlike fluid networks[29], chemical signaling networks[42] and electric networks[49], where both input and output are scalar quantities, our mechanical networks feature the vector input and output. This characteristic allows for more opportunities in higher dimensions.*"

The revised Supplementary Information in page 5 is attached as below:

"*In the main text, we demonstrate the Iris flower classification task. To further demonstrate the capability of our MNNs, we show the penguin classification task[3], aiming to classify three types of penguins – namely, Adelie, Chinstrap and Gentoo – using four distinct features: culmen length, culmen width, flipper length, and body mass. Supplementary Figure 8a visually illustrates the relation between culmen length and flipper length among three species. Similar to the classification process*

of Iris flower, we consider four scaled features, each corresponding to a downward input force applied on the nodes marked by red dots in the inset of Supplementary Figure 8b. The indicator of a specific species is determined by the node with the largest horizontal displacement among the three nodes in the inset marked by the corresponding symbols. The dataset is randomly partitioned into a training set (70%) and a testing set (30%). As the loss steadily decreases over epoch, the classification accuracy for the training dataset approaches 90% on average. Meanwhile, the accuracy for the testing dataset, which is unseen during the training process, also approaches 92% on average. The trained MNNs is shown in the inset with different width of the bonds. Supplementary Figure 8c illustrates the classification results at different epoch. At the beginning, most are classified as Gentoo penguin. With the increase of epoch, three species are distinctly separated in the feature space. Note that the incorrect classification results are marked by lime dots and the third panel of Fig. 4c exhibits good agreement with Supplementary Figure 8a which is the ground truth, affirming the efficacy of our classification model."

The revised Supplementary Figure 8 is attached as below:



Finally, much of the authors' claims of value rest on their method's superiority to other methods, yet show no comparison to other local methods like Coupled Learning. Further, systems trained in silico using any method (regular backpropagation, equilibrium propagation, coupled learning, the authors' scheme) would have the retrainable properties shown in Fig 5.

We thank the reviewer for the comment. We mention in the main text "Different from the local learning rule which relies on approximated gradient, our method shows the advantage of obtaining

exact gradient". In the Stern et. al., Physical Review X, 2021, they show the difference between their gradient and the exact gradient. Note that the primary goal of our work is to introduce a fundamentally new method to train MNNs, instead of conducting comprehensive comparative study among methods. To have a fair comparison, we compare the gradient between our method and EP because similar to our method, EP can define the arbitrary loss function and the nudge can be defined as a force. We show the gradient obtained by EP in the Supplementary Information and see the error compared with the exact gradient as a function of the nudging parameter. The relation between EP and our method is also derived to show in Supplementary Information (see the first response).

We agree with the reviewer that the retrainability has been demonstrated using coupled learning rules and in silico. However, since our method is new, it is very necessary to test whether the retrainability is still valid under our method and in our MNNs. And the results show that our method is as effective as other methods to retrain physical neural networks. Besides, for a certain task, different bond has different importance. Some bonds do not affect the accuracy much while others have significant impact. Moreover, we find a unique property in our system, zero mode, which has never been mentioned in other physical learning systems. If the damage happens in some certain bonds, the zero mode will emerge in the system, resulting in the failure of the retraining and the breakdown of the function of MNNs. This can provide the insights on how to design more robust and resilient MNNs. We revised the manuscript by emphasizing this point. Also, following Reviewer #2's suggestion, we add a experimental proposal using programmable device to realize retrainability.

The revised manuscript in page 13 is attached as below:

"*While the pruning of one bond is demonstrated above and previous works[47], our Supplementary Information delves into the effects of pruning different bonds on classification accuracy, where distinct bonds exhibit varying importance in classification tasks.*"

"*Moreover, the pruning of certain bonds can render the MNN mechanically unstable, leading to the emergence of mechanical zero modes, which causes the function failure of the MNNs.*"

The revised manuscript in page 17 is attached as below:

"*Our method opens the possibility to build a pure experimental setup of trainable MNNs without a central computer. This can be done using a general programmable device that tunes the spring constant using a microcontroller for experimental realizations without computer simulations. According to our learning rule, input force is applied to the device, and node displacements and elongation (e) are read by sensors. The microcontroller calculates the Jacobian of the loss from the output node displacements and applies an equivalent adjoint force to the device, with (e_{adj}) read by sensors. The microcontroller then performs element-wise multiplication of e and e_{adj}, conducting the update of the spring constant based on the calculated gradient. As discussed above, these calculations are of very low time complexity and feasible for microcontrollers without a central computer. This pure experiment setup can also avoid simulation-reality gap and realize retrainability.*"

As such I do not think this study meets the high bar for novelty and broad interest required for Nature

Communications.

With the improved discussion of the comparison between our work and the literature, thanks to the suggestions of all reviewers, and the fact that we propose a fundamentally new method based on backpropagation to train MNNs and fill the void of experiments in mechanics, we believe that our work now meets the high bar for novelty. Besides, considering the interdisciplinary nature of our research which bridges data science and mechanics, our work meets the broad interest and scope of Nature Communications. we hope that the reviewer will now find this manuscript suitable for publication in Nature Communications.

Below I include minor points that I hope will help improve the manuscript.

1. In several places the authors could make it easier to evaluate their experimental results. Specifically, in figures 3 and 4, the loss and accuracy of the experiment are not shown, but are arguably the most important quantities in the manuscript.

   We thank the reviewer for the suggestion. In the regression task, the experimental loss is 1.99×10^{-7} on average and the accuracy is 85.27% on average. In the classification task, the experimental loss is 0.5955 on average and the accuracy is 100%. We revised our manuscript by adding the loss and accuracy of the experimental results.

   The revised manuscript in page 12 is attached as below:

   "*Moreover, the experimental loss is* 1.99×10^{-7} *and the accuracy defined by the* l^2*-norm error* $1 - \frac{\|u-\hat{u}\|}{\|\hat{u}\|}$ *is* 85.27% *on average, respectively.*"

   The revised manuscript in page 13 is attached as below:

   "*Remarkably, in both simulation and experiment, the largest horizontal displacement occurs at the corresponding node, indicating correct classifications with the experimental loss* 0.5955 *and accuracy* 100%."

---

2. The authors imply on page 2 that mechanical neural networks may improve the speed of traditional neural networks, which I do not believe is supported by their references. Neural networks are extremely fast! The (I believe incorrect) speed advantage is mentioned again at the start of section 2.3.

   We thank the reviewer for the comment. Our primary intention is to show the advantage of physical neural networks such as optical and mechanical neural networks over computer-based neural networks, such as better energy efficiency and greater speed, especially for the extensively-studied optical neural networks. For the MNNs, the rigorous study is rare. The inference time is determined by the speed of sound of the materials, which varies depending on materials. This may or may not be faster than the computer-based neural networks. To avoid giving the impression that MNNs have greater speed, we revised the manuscript by deleting this statement.

3. MNN should be defined at its first use.

We thank the reviewer for the suggestion. Reviewer #2 also has such comment. The MNN in our work refer to as a network with n nodes and m linear springs in d dimension. We also ensure MNNs do not have zero mode under the applied boundary condition, meaning the compatibility matrix has full rank, in order to avoid the zero elongation under the applied force. We revised our manuscript by defining MNN at its first use.

The revised manuscript in page 4 is attached as below:

"*Here we have n nodes embedded in d-dimensional space, located at position $\{x_j\}$. The numbers of input and output nodes are n_{in} and n_{out}, respectively. The nodes are connected by m linear springs, each of which has a spring constant k_i. Also, zero modes are prohibited by properly designing the network connectivity such that the compatibility matrix C is fully ranked, which will be introduced later. Compared with the typical computer-based neural networks with layer-by-layer structures, MNNs are regarded as entire networks.*"

4. I am not familiar with the phrase "behaviors learning" (in contrast to machine learning tasks) if it is not a standard term I suggest the authors define it explicitly.

We thank the reviewer for the suggestion. Behaviors learning refers to designing materials to have desired behaviors under the load. We revised the manuscript by explain this concept.

The revised manuscript in page 3 is attached as below:

"*In contrast, the physical change induced in MNNs under static forces offer a promising solution to address these challenges, which has been demonstrated to learn behaviors (design desired response of materials under load)[25,26]*"

The revised manuscript in page 4 is attached as below:

"*Beyond their applications as computational devices, these MNNs also offer unprecedented opportunities in materials science and mechanical engineering as sustainable and autonomous materials systems, because they can be trained to learn certain behaviors to adapt to different environments and tasks.*"

The revised manuscript in page 10 is attached as below:

"*The example demonstrated above shows the MNNs can learn different behaviors under the applied force, where the utilization of in situ backpropagation offers a straightforward methodology.*"

5. Backpropagation should be defined when it is first used in the text.

We thank the reviewer for the suggestion. Backpropagation refers to a method to compute the gradient of the loss function. We revised the manuscript by defining backpropagation at its first use.

The revised manuscript in page 4 is attached as below:

"*We start from introducing the theoretical basis to conduct in situ backpropagation in MNNs, namely, obtaining the gradient of a loss function with respect to spring constants of MNNs.*"

---

6. The authors state that they retrieve the exact gradient of the loss "essentially without a computer." I understand the spirit of this claim and agree with its importance, but I do not think this phrasing is convincing or appropriate, as a computer was certainly used. "using local rules" or "without taking derivatives" or another more precise claim would work better.

We thank the reviewer for the suggestion. We agree with the reviewer that the digital computer is used, especially in the calculations of Jacobian and of element-wise multiplication between elongations. But since Jacobian is sparse and commonly-used loss functions are simple, the computational cost is as lower as $\mathcal{O}(n)$ (n is the number of the output nodes). Besides, the element-wise multiplication is an elementary arithmetic operation with the computational cost $\mathcal{O}(m)$ (m is the number of the bonds). This is not computationally heavy. To avoid the confusion that no computer is used, also combining the suggestion from Reviewer #2, we revised the manuscript by making it more precise.

The revised manuscript in page 3 is attached as below:

"*By using 3D-printed MNNs, we demonstrate the feasibility of obtaining the exact gradient of the loss function experimentally solely from the bond elongation of MNNs in only two steps, using local rules.*"

---

7. I found derivation of eqs 3-6 difficult to follow. There are several small but key details left out that would greatly help readability. For example, pointing out that $\frac{dF}{dk} = 0$ helps get to eq (3), D being symmetric allows $D = D^T$, necessary for multiple steps, etc. Further, it is unclear why u^*, then u^{*T}, then u_{adj} are all defined in quick succession – are all needed? Could the authors just start with u^T_{adj}?

   Perhaps point out earlier (before eq 5 or 4?) that the adjoint displacement field can be thought of (and indeed will be) the response to an imposed force at the output dependent on the output and the loss function? Or perhaps define "adjoint problem" explicitly.

We revised the derivation and the revised manuscript in page 5 is attached as below:

"*We show as below that this term can be derived from the differentiation of $Du = F$ on both sides:*

$$\frac{du}{dk} = -D^{-1}\frac{dD}{dk}u,$$

with $\frac{d}{dk}F = \mathbf{0}$. *Then plug Eq.(3) to Eq.(2):*

$$\nabla\mathcal{L} = \frac{\partial\mathcal{L}}{\partial u}\left(-D^{-1}\frac{dD}{dk}u\right) = u_{\text{adj}}^{T}\frac{dD}{dk}u.$$

Here we use the transpose of the adjoint displacement u_{adj}^{T} *to represent* $-\frac{\partial\mathcal{L}}{\partial u}D^{-1}$ *to obtain* $u_{\text{adj}}^{T}D = -\left(\frac{\partial\mathcal{L}}{\partial u}\right)$. *Note that* D *is a symmetric matrix such that* $D = D^{T}$. *Therefore, after taking the transpose on both sides, the adjoint problem can be defined as below:*

$$Du_{\text{adj}} = -\left(\frac{\partial\mathcal{L}}{\partial u}\right)^{T}.$$

Apparently, the two problems (forward and adjoint) differ in their forces. Likewise, the adjoint problem can be understood as the response u_{adj} *of a MNN under the adjoint force* $-\left(\frac{\partial\mathcal{L}}{\partial u}\right)^{T}$. *Then, we utilize the compatibility matrix[39,40],* $C \in \mathbb{R}^{m\times dn}$ *that maps the node displacement* u *to the bond elongation* e *in the linear regime, such that,* $e_i = \hat{I}_{j_1j_2}\cdot(u_{j_1} - u_{j_2})$, *where* $\hat{I}_{j_1j_2}$ *is a unit vector pointing from node* j_1 *to node* j_2, *which determines each entry in* C. *Besides, the stiffness matrix* D *can be decomposed into* $C^{T}KC$, *where* K *is the diagonal matrix with* k *as the diagonal entries. The gradient of* $\mathcal{L}$ *in Eq. (7) can be further expressed as:*

$$\nabla\mathcal{L} = u_{\text{adj}}^{T}\frac{dD}{dk}u = u_{\text{adj}}^{T}\frac{d(C^{T}KC)}{dk}u = u_{\text{adj}}^{T}C^{T}\frac{dK}{dk}Cu = e_{\text{adj}}\circ e,$$

where $\circ$ *is the Hadamard product (i.e., element-wise product).* $\frac{dK}{dk}$ *is a tensor* $\delta_{pql} \in \mathbb{R}^{m\times m\times m}$, *where the entry is* 1 *when* $p = q = l$ *and otherwise* 0. *Eq.(6)) implies that the gradient of the loss function* $\mathcal{L}$ *equals to the element-wise multiplication of elongations of the bonds in the forward problem* $Du = F$ *and the adjoint problem* $Du_{\text{adj}} = -\left(\frac{\partial\mathcal{L}}{\partial u}\right)^{T}$."

---

8. I am unsure what the "unexpected spikes in the decreasing loss when using coupled learning rules" refers to in references 31,32. In the former, the authors physically damage the system, which produces spikes in error, but that is similar to figure 5 in this work.

   We thank the reviewer for the comment. What we refer to the "unexpected spikes" is that the decreasing loss has many fluctuations in the training process. We do not observe many fluctuations in our training process. Besides, in our previous response, the most recent paper (Scellier et. al., NIPS, 2024) points out and proves that coupled learning rules neither perform gradient descent on contrast loss nor optimizes the squared error loss. Therefore, our best guess is that our exact gradient performing the gradient descent on the loss function can produce the smoother training curves. To avoid the impression that this is a solid evidence, we delete this statement in our manuscript.

---

9. "our experimental gradient achieves over 90% accuracy" is confusing – do the authors mean that

the average error is less than 10% of the true value? How is the averaging done?

We thank the reviewer for the comment. The error is defined by $1 - g_t \cdot \hat{g}_t$, where g_t and $\hat{g}_t$ denote the normalized obtained gradient and exact gradient, respectively. The experimental error is less than 10%, indicating over 90% accuracy. The averaging is done by calculating the average value of 6 measurements (3 measurements in main text and 3 in Supplementary Information.)

The revised manuscript in page 8 is attached as below:

"*We then define the gradient error as* $1 - g_t \cdot \hat{g}_t$*, where* g_t *and* $\hat{g}_t$ *are normalized measured gradient and exact gradient, respectively. Compared with the simulated gradient, which represents the exact gradient, our experimental gradient error is less than* 0.1*, as shown in the inset in Fig. 1e.*"

"*These results are summarized and averaged by three independent experiments described in the main text, along with additional three independent experiments for another loss function, which is detailed in the Supplementary Information (Supplementary Figure 1).*"

---

10. The authors' method is an approximation, and a key piece of information required to understand this ("in the linear regime" after eq 5) should be stated more clearly. I also think the authors should remove "In sharp contrast, the numerical implementation of our method does not produce error" for this reason. Further, the numerical implementation of the authors' method certainly produces roundoff error, as every computational calculation does – perhaps showing numerical error as a function of adjoint force, the relevant small parameter in this situation. This should be added to Fig. 1e for a fair comparison, or most of the x range should be removed, leaving only the experimentally feasible region, and experimental error.

We thank the reviewer for the comment. Our method produces the exact gradient theoretically in the linear regime which is the element-wise multiplication between forward elongation and adjoint elongation. However, as long as the deformation is not infinitesimal, error always exists except 1D MNNs. But note that in the simulation we can avoid this by just solving $Du = F$. Moreover, we agree with the reviewer that the numerical calculation can always produce the roundoff error depending on the machine precision. For two reasons above, our method is also an approximation numerically and experimentally. Nonetheless, what we indicate is that our method does not depend on the external parameters such as the nudging parameter in the equilibrium propagation or coupled learning. Our method only depends on the machine precision which is high enough for most computers. In our simulation, we use 16 digits of precision as default. The convergence condition is usually much greater than the machine precision. When the adjoint force becomes smaller and smaller, the learning is about to converge, so the roundoff error does not affect the learning and results. We revised our manuscript by making it clear and by adding the numerical error as a function of adjoint force in Fig. 1e. Indeed, if we use the real adjoint elongation in the simulation rather than

the linear elongation, we can find the gradient error in the simulation around 3×10^{-5} when adjoint force is 0.005×9.8 N.

The revised manuscript in page 7 is attached as below:

"*Different from the coupled learning rules and equilibrium propagation which rely on approximated gradient by the nudge strength, our method, as an alternative option to train MNNs, shows the advantage of obtaining exact gradient.*"

The revised manuscript in page 8 is attached as below:

"*Although numerically the gradient in machine precision can be obtained, the assumption in the linear regime requires infinitesimal deformation, indicating that our method used in the experiment is always an approximation. To explore the validity of our method in experiments, we perform the nonlinear analysis numerically and show the gradient error as a function of the adjoint force represented by weights in Fig.1e. When the weight is small, the adjoint field is in the linear regime, where* $e_i = \hat{I}_{j_1 j_2} \cdot (u_{j_1} - u_{j_2})$*, leading to the small gradient error. However, when the weight becomes large, the adjoint field deviates from the linear regime, where* $e_i = \|x_{j_1} + u_{j_1} - x_{j_2} - u_{j_2}\| - \|x_{j_1} - x_{j_2}\|$*, resulting in the gradient error from* 10^{-4} *to* 10^{-3}*. This low gradient error in the large adjoint force implies that our method can produce gradient in high precision and efficiency. Experimentally, we choose the weight to be small enough for linearity and large enough to produce measurable displacements, leading to an error estimate shown in Figs.1b and e.*"

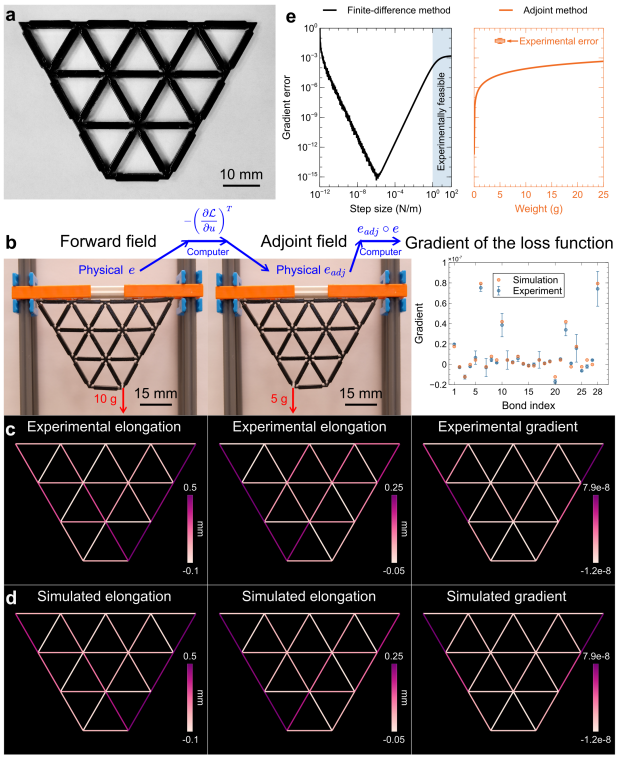
The revised manuscript in page 9 is attached as below:

"*Note that the roundoff error will also occur in our method, but thanks to the user-defined convergence condition, the minute adjoint force can be avoided.*"

The revised caption of Fig. 1 is attached as below:

"*The numerical error of the adjoint method depends on the machine precision in the linear regime. The orange curve in the right panel represents the gradient error when the deformation is not infinitesimal and the response of the MNN is nonlinear.*"

The revised Figure 1 is attached as below:



11. In figure 1b, perhaps the authors could use the empty space in the top right block to plot experimental vs simulated gradient against each other – it is very hard to get a quantitative sense of the quality of gradient from the heatmaps. Additionally it would help readability of the heatmaps to make the network edges thicker.

    We thank the reviewer for the suggestion. This suggestion is similar to that of Reviewer #2. We revised the Fig. 1 shown above by adding an explicit comparison between simulated and experimental results.

12. "another advantage of our MNNs is the potential utilization of the same node as both input and output node" – is this not feasible in other physical networks? E.g. in electrical networks, imposing current as an input and retrieving voltage?

    We thank the reviewer for the comment. Based on our learning rule and the governing equation of statics, the input is the force and the output is the displacement. Because the input force and the output displacement are two different physical quantities, the input and output can be defined on the same node and do not affect each other during the training. In other networks system, it may work as well. However, in the electrical networks, based on the Kirchhoff's circuit laws, one can relate the current along the bonds and voltage of the node by impedance matrix or admittance matrix, where current is defined on the bonds instead of

the node. In our mechanical case, both force and displacement can be defined on the node.

The revised manuscript in page 9 is attached as below:

"*In addition, another advantage of our MNNs is the potential utilization of the same node as both input and output node, owing to the force and displacement being different physical quantities that can be defined on the same node.*"

---

13. I enjoyed the supplementary videos very much, however it is hard to tell what is going on in the network, since all that is shown is the structure changing. Additionally showing the state when force is applied would help greatly!

We thank the reviewer for the suggestion. The change of the bond width in the Supplementary video represents the learning processing of the MNNs. The state of the output nodes in the behaviors learning task is the displacement, and that in the regression task is the regression result. However, it is difficult to show it in classification tasks, because each data represents a set of forces applied on the MNNs, and thus one needs to iterate over all data in the dataset for one epoch, which greatly hinders the fluency of videos. So we only show the normalized displacement of the output nodes under three sets of forces corresponding to three Iris flowers. We revised the Supplementary videos by adding the state of output nodes when force is applied.

---

14. "It is noteworthy that using the cross-entropy function as the loss encourages the maximization of the displacement difference between two nodes, rather than a specific value when using mean-squared error (MSE) as the loss." Why did the authors choose not to use MSE of the difference between the node heights? Would that not have worked as well?

We thank the reviewer for the comment. We use the cross-entropy loss in the main text and also use the MSE loss in the Supplementary Information. They both work well. The cross-entropy loss is demonstrated when we have a "vague" target such that one node has larger displacement than another, rather than designating a specific value for these two nodes. When we want two nodes to develop the displacement with certain values, MSE loss is used, as shown in the Supplemental Information.

The revised manuscript in page 10 is attached as below:

"*It is noteworthy that using the cross-entropy function as the loss encourages the maximization of the displacement difference between two nodes rather than considering the designated values when using mean-squared error (MSE) as the loss. In addition, we demonstrate the precise control of displacements of two nodes under the applied force* $F = 0.005 \times 9.8$ N *by using MSE, as detailed in the Supplementary Information and Supplementary Figure 5. Both loss functions can be leveraged to conduct behaviors learning as needed.*"

---

15. It would be helpful if the authors could cite a regression benchmark for machine learning. The most famous ones are all classification (e.g. MNIST, CIFAR-10, etc).

We thank the reviewer for the suggestion. We revised our manuscript by citing Abalone age prediction and wine quality model as regression tasks.

The revised manuscript in page 11 is attached as below:

"*Regression stands as benchmark tasks in the field of machine learning with typical examples of Abalone age prediction and wine quality model[45, 46], serving as a cornerstone for evaluating model performance and predictive capabilities.*"

---

16. Why does the data in Fig. 3a include upward forces if they cannot be used in the experiment?

We thank the reviewer for the comment. Our regression task is $u_{Rx} = 0$ and $u_{Ly} = 4u_{Lx}$ and we use the synthetic data in a reasonable range that remains linear deformation to train MNNs. However, due to our experimental setup where the force is applied by the gravity of the weights, we can only testify the lower part of the regression target which is shown in the third panel of Fig. 3(c).

The revised manuscript in page 12 is attached as below:

"*Due to the applied force from the gravity of weights in our experimental setup, we test regression results under the downward force.*"

---

17. The training and test error in Fig. 4 look to be about 95%, which is excellent for a linear system, but I think the authors should remove "nearly 100%" and be more quantitative about these results. Additionally, it is hard to see the final accuracy with such a large y scale (Fig. 4b).

We thank the reviewer for the suggestion. The classification accuracy for the training set on average is 95% and that for the testing set is 96%. We also make the limit of y axis to be from 20% to 100%.

The revised manuscript in page 13 is attached as below:

"*As the loss steadily decreases over epoch, the accuracy of the classification defined by the ratio of the correct classification for the training dataset approaches 95% on average. Meanwhile, the accuracy for the testing dataset, which is unseen during the training process, also converges to 96% on average...*"

---

18. It is difficult to tell how well the system is doing from the plots in Fig. 4c at a non-approximate level. Perhaps the authors could highlight the incorrectly identified flowers?

We thank the reviewer for the suggestion. We show the confusion matrix in the Supplementary

Information, which could help to understand the performance of the model. In addition, we follow the reviewer's suggestion to highlight the incorrectly identified flowers.

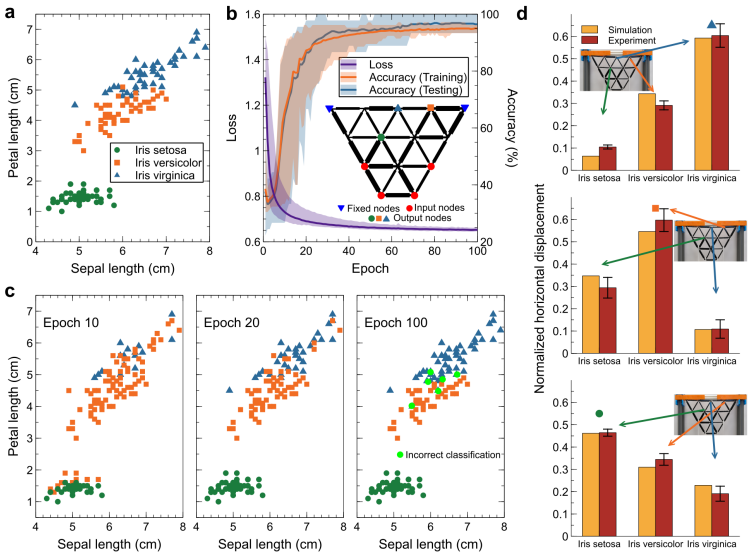
The revised manuscript in page 13 is attached as below:

"*The incorrect classification is marked by lime dots in the third panel of Fig. 4c.*"

The revised caption of Figure 4 is attached as below:

"***c*** *The classification results when the epoch is* 10, 20 *and* 100 *are shown from left to right, respectively. The lime dots in the third panel represent the incorrect classification data.*"

The revised Figure 4 is attached as below:



19. Are there differences between the networks in the images Fig. 4d? If so they are too subtle to tell. What is the experimental classification accuracy?

We thank the reviewer for the comment. The images in the inset show the MNNs under the corresponding input. Under the applied forces from flower features, the three nodes indicating flower types have different displacements, the normalization of which are shown in Fig. 4d. However, MNNs is implemented in the linear regime, resulting in the small deformation which is hard to tell from the images only. We draw arrows in Fig. 4d shown above to indicate the displacement of the corresponding nodes. Currently, the experimental classification accuracy is 100%.

The revised manuscript in page 14 is attached as below:

"*In both simulation and experiment, the largest horizontal displacement occurs at the corresponding node, indicating correct classifications with the experimental loss* 0.5955 *and accuracy* 100%."

20. In Fig. 4, was a different input/output configuration attempted? If so did it succeed? The success at this task may depend on the specific node placements, since there are only so many connections in the network.

Good question! We did try different input/output configurations in the Iris flower classification task. Some are successful with high accuracy while others fail with low accuracy. There are several reasons: First, like what the reviewer mentioned, our small MNNs have limited bonds (learning degree of freedom). Second, for the purpose of fabrication and experiments, we limit the width of the bonds in a range from 1.5 mm to 2.5 mm. Third, the input force is only limited to downward force due to the direction of the gravity. Fourth, for the convenience of using camera to take pictures and to accurately extract the nodal displacement, there should not be any overlap between nodes and applied weights. Given all of these consideration above, we finally choose the configuration shown in Fig. 4 to experimentally implement the classification task. However, if we do not have the consideration of the experimental setup (i.e., the positions of applied weights and observed nodes), we can have more input/output configurations. We added three more configurations as well as their learning process in the Supplementary Information shown in Supplementary Figure 7.

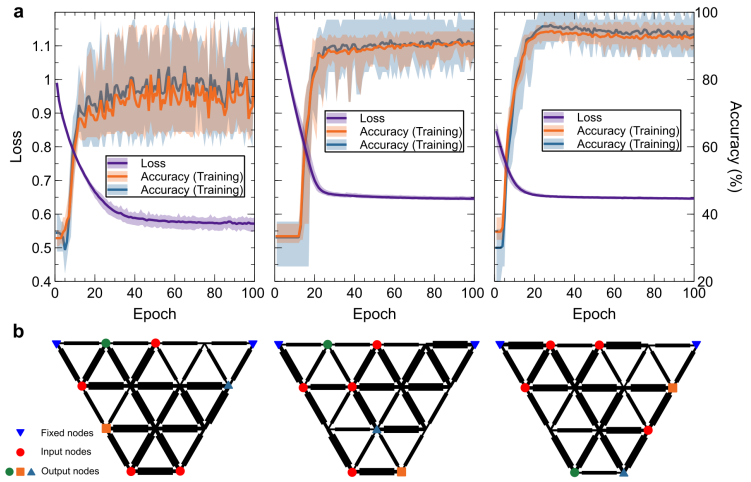
The revised manuscript in page 14 is attached as below:

"*Note that the choice of the input nodes and output nodes is crucial for the successful training of MNNs. The configuration shown in Fig. 4b is ideal for experimental implementation and measurement. In the Supplementary Information and Supplementary Figure 7, we show three more choices which may not be ideal for experiments under our setup but are still able to reach high accuracy after training.*"

The revised Supplementary Information in page 6 is attached as below:

"*Moreover, the choice of input and output nodes in our MNN is important to ensure the success of the classification because of the limited learning degree of freedom and the experimental considerations. We demonstrate three more choices of input and output nodes for the successful classification with high classification accuracy, with training process shown in Supplementary Figure 7a and corresponding trained MNN shown in Supplementary Figure 7b. Notably, all three cases experience decreasing loss and increasing classification accuracy, with 89%, 91% and 93% accuracy in the final epoch, respectively. The input and output nodes are marked on the MNNs.*"

The revised Supplementary Figure 7 is attached as below:



21. The task switching figure and supplementary video are great.

We thank the reviewer for the positive comment.

22. The authors point out retainability several times as an advantage, but their experimental apparatus is manufactured with specific weights and must be entirely remanufactured if any changes are made. I think these claims are premature for this work.

We thank the reviewer for the comment. We demonstrate the retainability of MNNs after switching tasks and damage, which shows the potential of MNNs. If MNNs are made of programmable device like Reviewer #2 said, it is feasible to realize without re-manufacturing. To avoid any confusion, we revised our manuscript by making it clear.

The revised manuscript in page 15 is attached as below:

"*Both retrainable properties rely on the tunable spring constants in MNNs, the experimental implementation of which will be discussed in the Conclusion.*"

The revised manuscript in page 17 is attached as below:

"*Our method opens the possibility to build a pure experimental setup of trainable MNNs without a central computer. This can be done using a general programmable device that tunes the spring constant using a microcontroller for experimental realizations without computer simulations. According to our learning rule, input force is applied to the device, and node displacements and elongation (e) are read by sensors. The microcontroller calculates the Jacobian of the loss from the output node displacements and applies an equivalent adjoint force to the device, with (e_{adj}) read by sensors. The microcontroller then performs element-wise multiplication of e and e_{adj}, conducting the update of the spring constant based on the calculated gradient. As discussed above, these calculations are of very low time complexity and*

feasible for microcontrollers without a central computer. This pure experiment setup can also avoid simulation-reality gap and realize retrainability."

---

23. "In contrast to other physical neural networks, such as optical neural networks, our MNNs offer greater ease of operation in experiments and real-world applications across diverse environments." I do not think this claim is justified and should be removed.

    We thank the reviewer for the comment. MNNs show the better resistance in electromagnetic environment over optical counterparts, which is mentioned in the Introduction. We follow the reviewer's suggestion to remove the claim here.

---

24. I could not access the code through the above DOI.

    We migrate our code to github for better accessibility and revise the Code availability in the main text.

    The revised manuscript in page 19 is attached as below:

    "*The codes for demonstration can be found in the following link:* `https://github.com/mao-research-group/Mechanical-neural-networks`"

Reviewer #1

The authors have sufficiently addressed many of my comments. However, one important comment remains, having to do with the relations of in-situ backpropagation and the equilibrium propagation algorithm. I still maintain that both are the same for linear systems. They certainly suggest the same learning rule. The rebuttal offered by the authors does not contradict this observation. Specifically, the comment that both algorithms are the same in the limit $\beta \to 0$ is indicative, as equilibrium propagation is after all defined in that limit. I do appreciate that in situ backpropagation is derived in a different way then the original EqProp, and that this derivation technique may give unique learning rules for other physical systems, yet in linear networks both are in fact the same. Still, I believe that the current work is sufficiently novel, both in its derivation of the in situ backprop rule, and the way that it was used in theory and experiments of elastic structures. I'd be happy to recommend the paper for publication once the authors addressed this issue, dropping the claim that in situ backpropagation gives a different result (and certainly not superior) result than equilibrium propagation.

We are glad that we have addressed many of the reviewer's concerns in the previous response and we thank the reviewer for the comment. According to our derivation of the relation between our method and EP in the Supplementary Information: $\Delta k_i = \Delta k_{\mathrm{EP},i} + \frac{1}{2}\alpha\beta e_{\mathrm{adj},i}^2$, we agree with the reviewer that nudging strength $\beta \to 0$ in the linear regime, so $\Delta k_i = \Delta k_{\mathrm{EP},i}$. Although they produce the same results in the linear regime, from the perspective of the training algorithm, the in situ backpropagation uses the element-wise multiplication between forward elongation and adjoint elongation $\Delta k_i = -\alpha e_i e_{\mathrm{adj},i}$, while EP uses the difference between two states $\Delta k_i = \frac{\alpha}{2\beta}(e_i^2 - e_{\mathrm{nudged},i}^2)$. The lack of a small parameter β in our method can be useful in practice, as suggested by Reviewer #3.

Besides, for the implementation in simulations and experiments, in situ backpropagation uses forward pass and backward (adjoint) pass, while EP uses forward pass and a nudge in the presence of the forward pass. We make it clear by adding the statement about the same results in the linear regime produced by our method and EP.

The revised manuscript in page 6 is attached as below:

"*Notably, in the linear regime, our method and EP produce the same results despite different procedures and algorithms.*"

The revised Supplementary Information in page 5 is attached as below:

"*Therefore, in the linear regime, our method and EP can produce the same results, although procedures and algorithms are distinct.*"

The revised manuscript in page 7 is attached as below:

"*Apart from EP and coupled learning, our method serves as an alternative option to train MNNs locally.*"

The revised manuscript in page 9 is attached as below:

"*When the nudging strength is small in the linear regime, the results from our method and EP are identical.*"

I did not review the code as I do not have access to Matlab at the moment.

The code is open source and available in the GitHub page of our group, so the reviewer and readers can get access to it at their convenience.

Reviewer #2

I am more or less satisfied with the author's revisions and for my part am ready to see the work published. I expect there to be some further debate and discussion over exactly the nature of the novelty in the work, but I think this is resolved sufficiently to justify publication (and any further debate or refinement would, I think, benefit the community anyway).

Thanks to the authors for the care in revising the manuscript and addressing my concerns.

We thank the reviewer for the time spent in reviewing our manuscript and we are glad that the reviewer is satisfied with the revisions.

Reviewer #3

The authors have answered the bulk of my comments thoroughly, and I thank them for their hard work. The text is much improved and the claims have been appropriately toned down, with needed additional clarifications about the linearity of their method and scope of their experimental work.

I now agree that their method and Equilibrium Propagation are, strictly speaking, distinct, for which their derivation was very helpful. The lack of a small parameter in their method is indeed a plus. However, as the authors state, their method is 'exact' only in the linear regime, therefore requiring small deformations of the system, which is in effect enforcing a small parameter. I believe extending this method to nonlinear systems will be difficult (and is the reason EP, Coupled Learning, and Contrastive Learning compare two like-states instead of applying the adjoint directly), however as long as the authors are clear about the scope of validity of their method, this seems like reasonable future work.

We are glad that we have addressed many of the reviewer's concerns in the previous response and we thank the reviewer for the comment. We agree with the reviewer that extending it to nonlinear systems would be nontrivial and could be future work to explore. We stated it in the original manuscript in page 17: "*Given the derivation of our learning rules based on the matrix decomposition and linear operation, the nonlinear case cannot be naturally generalized. Therefore, exploring the nonlinear regime of MNNs by using nonlinear materials and geometric nonlinearity presents an opportunity not only to theoretical interest, but also to tackle nonlinear datasets and tasks, which is worth exploration in the future study*".

Notably, the abstract has remained unchanged from the previous version, which I believe is still a bit misleading. Specifically the mention of "exact" gradient without qualification and ambiguity about what

was done in experiment and what was done in simulation.

We thank the reviewer for the comment. The exact gradient can be obtained theoretically, but as mentioned in the previous response, the roundoff error from the digital computer and the deformation in experiments make gradient inaccurate. In our work, we experimentally demonstrate the in situ backpropagation to obtain the gradient and machine learning inference in MNNs. The learning process, i.e., updating spring constants, is demonstrated in simulations using the bar model.

The relevant part of the revised abstract is attached as below:

"*We theoretically prove that the exact gradient can be obtained locally in MNNs, enabling learning through their immediate vicinity, and we experimentally demonstrate this backpropagation to obtain gradient with high precision. With the gradient information, we showcase the successful training of MNNs in simulations for behavior learning and machine learning tasks, achieving high accuracy in experiments of regression and classification.*"

Further, in the second to last paragraph of the introduction the statement "we showcase the successful training of a mechanical network" comes directly after mentioning experimental verification, and seems to imply the training was in the mechanical network. There are several instances of this implication in the paper, which confused and irked me as a reader. Making this aspect much clearer would help readers like me (and from the review, like Reviewer 2 as well) to not feel like there are details being hidden. I agree with Reviewer 2's assessment of much of this manuscript – there is excellent work here, but it was confusing as to what was actually done and in some places oversold. In its improved form, I will not stand in the way of this work's publication, provided appropriate changes are made to the abstract and introduction.

We thank the reviewer for the comment. We revised the manuscript by making it clear that what is done numerically and what is done experimentally in each learning task.

The revised manuscript in page 3 is attached as below:

"*Here, we present a highly-efficient training protocol for MNNs through mechanical analogue of in situ backpropagation, derived from the adjoint variable method, in which theoretically the exact gradient can be obtained from only the local information.*"

The revised manuscript in page 4 is attached as below:

"*we showcase the successful training in simulations of a mechanical network for behaviors learning and various machine learning tasks...*"

"*The trained MNNs are then validated both numerically and experimentally.*"

The revised manuscript in page 10 is attached as below:

"*Fig.2b shows the progressive reduction in loss during training in simulations until convergence.*"

"*Notably, experimental measurements of the trained MNNs align closely with simulated displacements across all scenarios, validating the efficacy.*"

The revised manuscript in page 12 is attached as below:

"*...for the noise-free and noisy datasets exhibit consistent decrease in simulations over epoch until convergence.*"

The revised manuscript in page 13 is attached as below:
"*As the loss steadily decreases over epoch in simulations...*"
The revised manuscript in page 14 is attached as below:
"*Furthermore, we perform classification experiments to validate the model stored in the trained MNNs.*"
"*Here, we highlight the retrainability in two key scenarios through simulations...*"

A few additional comments:
The authors should clarify that [33] and [35] are simulations of Equilibrium Propagation, and not experimental demonstrations.
We thank the reviewer for the comment. We revised the manuscript by clarifying these two works are simulations. Note that the sequence of references changes in the revised manuscript, and now [33] and [34] are simulations.
The revised manuscript in page 3 is attached as below:
"*This method has been demonstrated in various physical systems, numerically in nonlinear resistor networks[33] and coupled phase oscillators[34], experimentally on Ising machines[35].*"

Coupled learning cannot precisely resolve an "arbitrary loss function" (though strictly speaking neither can any of the other discussed methods), but it can be used to solve arbitrary tasks. Could the authors rephrase this sentence?
We thank the reviewer for the comment. Coupled learning is effective in solving arbitrary tasks. We revised the manuscript by rephrasing this sentence.
The revised manuscript in page 3 is attached as below:
"*One method, termed coupled learning, uses the contrast of two equilibrium states (free and nudged states) to update the system and is able to solve arbitrary tasks [28-31].*"

The authors denote "100%" accuracy in reference to Fig 4d, by which I believe they mean for the three shown examples only? This is confusing as the simulation and experiment do not have 100% accuracy overall, as shown by 4b and 4c.
We thank the reviewer for the comment. By 100% accuracy, we mean the accuracy for the tested examples is 100%. We revised the manuscript by clarifying it.
The revised manuscript in page 14 is attached as below:
"*In both simulation and experiment for three tested cases, the largest horizontal displacement occurs at the corresponding node, indicating correct classifications with the experimental loss* 0.5955."