Response to reviewers

The authors would like to thank the reviewers and editor for their time and their helpful comments to improve our manuscript. Accordingly, we have revised the manuscript based on their suggestions and comments. In the following, we have answered all the comments through a point-by-point response and the corresponding corrections in the manuscript have been highlighted in blue. We believe that the revised version is clearer and better than the previous submission.

All the changes in the manuscript are colored in blue to help the reviewers navigate better.

Reviewer #1

This paper extends the original DEM method by training the network in a parameter domain, where NURBS basis functions are adopted. The proposed P-DEM method has been tested in problems of elasticity and strain-gradient elasticity. In particular, the strong form of the strain-gradient elasticity problem is defined by a 3rd-order PDE which cannot easily be solved in traditional FEM due to the C^1-continuity requirement. The paper is well written with strong new technical contributions. The reviewer has a few suggestions for minor revision.

Comment 1: For the strain-gradient elasticity problem, can the authors also compare P-DEM with isogeometric method (IGA)? The reviewer is wondering if the superior performance of P-DEM over FEM is due to the high order NURBS basis functions. Several other basis functions have been studied for IGA such as THB-splines, subdivision and so on. Can these basis functions be used in P-DEM too? In "Deep Learning of Material Transport in Complex Neurite Networks. Scientific Reports, accepted, 2021. arXiv preprint arXiv:2103.08654," IGA results are used in training data collection and its residual is included into the loss function. A discussion or comparison of how basis functions influence the prediction accuracy could be provided.

Our Response: The IGA here is only for the geometry representation and not directly used for the field approximation. The increase of the control points in IGA or choices of basis function shall not significantly affect the training process and hence not dominate the accuracy of P-DEM. Though more accurate geometric representation may be realized by adding more Gauss points where the residual is large as in https://doi.org/10.1016/j.tafmec.2020.102527, we do not think it is dominating. The implementation of IGA formulation for strain gradient problem will entail substantial additional work to the present work since the assembly of the global stiffness matrix is tricky. Therefore, we opt to leave this comparison for future work.
[see Page 10]

Comment 2:
- Page 2, Line 48 and Line 49: the front quotation doesn't show correctly.
- Page 3, Line 21: attracted also also attracted
- Page 30, Line 57: CPU time as the CPU time than the

Our Response: Thank you for your careful reading. We already revised the manuscript and corrected the typos.

Reviewer #2

In this paper, the authors have presented an extension of the existing Deep Energy method. In the proposed approach, the network is trained on the parametric domain instead of the physical domain. The underlying problem and the related integrals are formulated in the physical domain and are mapped to the parametric domain ([0,1]) for generating training data. The authors have adopted the concept from isogeometric analysis. The developed approach would be advantageous since the trained network could be used for other problems with similar setup as the geometry on the parametric domain would always be the same. The topic is of interest and the work has enough novelty. However, the reviewer suggests a major revision to improve the quality of the manuscript before it is considered for publication. Following are the major comments

Thank you very much for your encouraging feedback and positive comments: Below, kindly find our responses.

Comment 1: It would be extremely helpful if the authors could consider providing an algorithm of the approach to see how the mapping is done from physical domain to parametric domain and later back to physical domain. Would PDEM be beneficial for adaptive refinement over conventional DEM?

Our Response: We think that the current version of the paper is extensive. Adding algorithms will cost space. We have presented the mathematical concept for mappings in section 3.3. Moreover, combining with the request from reviewer #3, we, therefore, think that providing code along with the paper will help readers to practice and develop further works if they are interested in the method more efficiently. We do not doubt the efficiency of the method when it is extended for adaptivity. We have proved this possibility in our previous publications [1, 2] using the collocation method though it does not likely contain the exact concept of P-DEM.
[1] - https://doi.org/10.32604/cmc.2019.06641
[2] - https://doi.org/10.1016/j.tafmec.2020.102527
[see Page 10, 34]

Comment 2: In the theory and also in conclusion, the authors have mentioned that they have used Gauss quadrature points to compute the integrals. However, through out the numerical section, the authors have used equidistant points. Please be consistent and clarify.

Our Response:
The concept of transformation between the physical domain and parametric domain is plotted in Figure 3. First, the physical domain is converted to the parametric domain by

the Jacobian J1 constructed by the NURBS basis function. Second, the parametric domain is discretized into smaller domains, each of which is called the parent domain. The mapping from an element in the parametric domain to the parent domain is done by Jacobian J2 constructed by the Lagrange shape function as in Finite Element. Here, weight factors and coordinates of Gauss points generated at the level of the parent domain are used to calculate Jacobians J1, J2, and w in equation (20). The strain energy density is calculated at the corresponding Gauss points in the reference domain. Therefore, in the case of linear Lagrange shape functions being used to interpolate the spatial coordinates, the set of equidistant points in the reference domain is the set of Gauss points. The mapping for the boundary points is done analogously to the domain mapping for the 1D case.

Comment 3: For all the problems, the authors have used L-BFGS which is definitely expensive than Adam because it uses an inverse Hessian matrix to steer the search. Please throw some light if Adam does not work good for PDEM since Adam works efficiently for conventional DEM.

Our Response:
The authors agree that L-BFGS is more expensive than Adam due to the fact that L-BFGS estimates the inverse of the Hessian matrix while Adam, which is an extension of stochastic gradient descent, uses only the first gradient. This factor brings a big benefit especially in the case of dealing with a big dataset. However, in our problem precision is the highest priority and L-BFGS can go deeper in valleys which are the local minima of our optimization problem. In this case, L-BFGS is more efficient than the Adam algorithm. A combination of Adam and L-BFGS would be a trade-off between speed and accuracy. Indeed, this combination was already used in our previous publication [1]. Despite the positive values coming from the combination, our empirical evidence in this study shows that using standalone L-BFGS would yield the same accuracy and save the memory. Other publications [2, 3] have shared the same thoughts with us.
[1] - https://doi.org/10.1016/j.cma.2019.112790
[2] - https://doi.org/10.1016/j.jcp.2018.10.045
[3] - https://arxiv.org/abs/1710.09668

Comment 4: Page 21: Line 41: “the result yielded by P-DEM is convergent with respect to the number of hidden layers.”. In Table 4: the authors can include a case by using 4 hidden layers with 30 neurons each and could show the convergence wrt the number of hidden layers. The current information does not necessarily support the mentioned statement.

Our Response: We did the extra work with 4 hidden layers and we modified the text based on the obtained results.
[see Page 23]

Comment 5: For a fair comparison with the conventional DEM it is desired that the authors show how the predictions have improved when the simulations is done over physical domain retaining the same network architecture. The comparison may include computational time, % accuracy etc. On Page 19: the authors mention that DEM and PDEM both converge after 200 iterations, how is the convergence faster (or better)?

Our Response: We already compared P-DEM training on the reference domain with conventional DEM training on the physical domain and we retained the same network architecture for this purpose in example 4.1.2. For accuracy, we measured the L2-error norm, energy norm. The results in Figures 13b, 13c show the P-DEM curves always go deeper than DEM. That means P-DEM obtains smaller error than DEM, and thus more accurate than DEM if we use the same settings such as dataset, network architecture, hyperparameters. Figure 14 and Table 3 consolidate our statement by giving the point-wise error on the physical domain and the exact values of L2-error norm and energy norm after completing the training process.
Regarding the convergence, in the last version, we have concluded that P-DEM always converges faster than DEM in all cases and explained the reason why P-DEM is superior to DEM. This is because the data are automatically scaled within the unit domain and it avoids the gradient vanishing issue. We do not provide the exact number for computational time because we set the epochs value to be 20 for both P-DEM and DEM as a criterion to complete the training. It would be more or less the same in terms of computational time if both P-DEM and DEM have no difference in the calculation, i.e. the order of derivatives. Note that the forward and backward mappings in P-DEM are executed in pre-processing and post-processing, relatively.
[see Page 19, 20]

Comment 6: In section 4.1.1, it is quite clear that 30X30 grid points and tanh yield optimum results, when the error difference and the training time difference is considered against 50X50 grid points. However, the authors state differently. This should be taken into account.

Our Response: We do agree with the reviewer’s comment and have the same observation. Maybe we make readers confused at some points in the conclusion. We, therefore, have rewritten the conclusion part for example 4.1.1
[see Page 15, 16]

Comment 7: Figure 20: With dataset 2, the error of PDEM and DEM is almost the same. It would be helpful if the authors could show how much significant improvement has been noticed with PDEM

Our Response: In Figure 20. we did not compare the error between PDEM and DEM. We assume that the reviewer mentioned Figure 14 since it contains the exact information which the reviewer pointed out. Knowing that the point-wise error plotting does not clearly show the improvement, we already gave the exact values in terms of L2 error norm and energy norm in 2 cases: dataset 1 and dataset 2. As shown in Table 3, P-DEM has improved the results since the errors measured in L2 norm and energy norm are much smaller than the error of DEM. We have rewritten the conclusion to explain it more clearly.

[see Page 20]

Comment 8: The authors have mentioned it repeatedly mentioned that the convergence of the loss function is faster than conventional DEM. It would be better if this statement could be supported by examples and results.

Our Response: We have replied to this point as an extension of the answer to comment 5. We rewrote the statement on pages 19, 20. Thank you for your very detailed review.

Comment 9: In the problem of the annulus shown as an example for the strain gradient elasticity section: It is not clear why 2 networks were required, when the loss function would contain terms of strain difference computed between the strains computed using the displacement and the strain computed using the second network. Also, it is important to understand that if strain is taken as an output of the network, then strain compatibility conditions need to be checked since it is not satisfied automatically, as is the case when displacement is taken as output. The authors might consider providing results of the same problem when only one network is used, and displacement is the only output.

Our Response: As we have explained in the text that the reason to choose such structure is that displacement and strain have different lengthscale ratios, i.e. one has a unit of measurement while the other one is unitless, sharing the same ansatz functions, which can be considered as the activation functions, might lead to a suboptimal approximation. Besides, to calculate strain gradients the derivative of strain with respect to parametric coordinates needs to be evaluated, i.e. dE/dX = dE/dxi . dxi/dX, using only displacements as the output of the network results in the tedious derivation and computation of this term. Instead, adding more neurons for output strains will help us to fully exploit the potential of our network because it utilizes the automatic differentiation of the computational graph integrated into the machine learning framework. Note that we still calculate the components of strain based on the primary output displacements like in elasticity, denoted by $\hat{\epsilon}^*$, and the additional output strains, denoted by $\hat{\epsilon}$, are enforced by a regularization term, i.e. $||\hat{\epsilon}^* - \hat{\epsilon}||$. Therefore, the compatibility conditions are still satisfied.
[see Page 27]

Comment 10: In the conclusion section. The authors have claimed “In all cases, it yields better results in a shorter CPU time as the original DEM.”: It would be better if the authors could present the computational time and accuracy of the conventional DEM using the same setup and used for the problems presented in the manuscript for a fair comparison.

Our Response: As we have answered in response 5, although we did not provide the exact time values which are not really necessary in our case, we are still able to conclude that P-DEM achieved the optimal value of energy loss and reached the level of smaller errors faster than DEM. This conclusion is based on the observation of the convergence loss of energy, l2 error norm and energy error norm. That being said, we do not want to claim that P-DEM runs faster than DEM because we set the same

epochs value for both P-DEM and DEM as a criterion to stop the training. Therefore, the running time for both methods would be more or less the same. We realized that the conclusion makes the readers confused and did not convey what we intended to present. We, therefore, have rewritten the conclusion to make it clearer.
[see Page 20]

Comment 11: An important advantage that could be shown in this method is that since any domain is mapped onto the square parametric domain, transfer learning could be used train the NN for any similar problem. It would be helpful if the authors could comment on this.

Our Response: Thank you for your idea and vision. This is also the further objective we desire to achieve that we could reuse the knowledge of trained networks due to the same or sharing information among problems, i.e. in our case it is the reference geometry, for the next training. In this spirit, transfer learning could provide the key to obtain the final goal. In fact, we are working on this problem and obtained some first results, and we wish to make a different contribution. However, we still want to share the preliminary results of this direction. Therefore, we add this part as an extension of example 4.1.2 (the annulus) where we applied transfer learning leveraging the network's parameters of example 4.1.1 (the beam) to the training for the annulus problem. We introduced this part and added some results in the appendix.
[see Page 31, 32]

Minor Comments:
1. In the abstract: “pure deep neural network”: Pure does not quite make sense. Should be rephrased.
2. Page 1: Line 25: “converges” should be “convergence”.
3. Page 4: Line 11: “paprameters” should be “parameters”.
4. In Equation 6 and 7: the activation function could be different for each layer. Hence it is better is the activation function is subscripted with the hidden layer number.
5. It is better to use different notations for each indicating different parameters. The learning rate and the test function have the same variable.
6. Page 7: Line 24: L(.) is the loss function and not the neural network definition.
7. Page 7: Line 27: The order is not always halved considering numerous other PDEs. It s better to mention that the variational form has derivatives of lower order than the residual form.
8. Page 7: Line 31: This is not at all a key feature.
9. Page 7: Line 37: Isoparametric finite elements should be isoparametric elements.
10. Page 8: Line 20: “First Basis”: It is better to mention that N i,0 is the basis function when the polynomial degree is 0.
11. Page 9: “The necessary derivations” should be derivatives.
12. Page 14: “Due to dense of dataset”: This is not quite clear what it means.
13. Page 15: In equation 38 and 39, what is û, u and û L ? Please clarify which is the output of the NN and which one is the modified output?
14. Figure 9: Which activation function and dataset was used?
15. Page 8: the outward normal has neither been shown in the equation nor has been shown in the image.

16. Page 15: Line 31: “interal radius” should be “internal radius”.
17. Figure 12: Please plot the exact plots and the point wise error for each of the u_x, u_y and u(r). It will give more clarity on the accuracy.
18. Figure 13: In the convergence plots, it would be better if the x-axis (no. of iterations) is not plotted on the semilogy scale.
19. Figure 15: What is exaction traction? It may be changed to just “traction”.
20. Page 24: Line 45: “In contrast, the basis functions in DEM easily ...” : DEM should be P-DEM
21. Page 24: It is important to clear if “c” is the length scale parameter or the strain gradient coefficient.
22. Page 24: Equation 54 and 55 needs to be checked.
23. Page 25: “thick-wall cylinder from section 4.1.2”. The problem in section 4.1.2 is of an annulus and not a thick wall cylinder.
24. Figure 18: The caption is wrong.
25. What is εll is Equation 48 and 49?
26. Page 30: Line 45: “strain-gradient elastictiy” – elasticity.
27. The manuscript needs thorough revision to eliminate grammatical errors and punctuation mistakes which seem to persist throughout.

Our Response:

1. We rewrote it to “The approach is based on physics-informed neural networks (PINNs) for the solution of the underlying potential energy”.
2. We corrected it already.
3. Thank you for your very careful reading. We already corrected it.
4. We still want to retain the current notation because we want to use the same notation in Nielsen’s book and in our work the activation functions should be the same for each layer.
5. We agree with the reviewer’s comment. Now we use \gamma for the learning rate and \eta for the coordinates in the parent domain.
6. We already corrected it.
7. We agree with the reviewer’s comment and have corrected it already.
8. Through our experience when we dealt with finite-element approaches, this is the key feature. One of the important reasons we continue to pursue this direction is its simplicity in terms of implementation compared to conventional FEM and other numerical methods. Therefore, we want to retain this feature but we rewrote it to make this point clear. Now it is rewritten “Compared to the finite-element approaches, the implementation is simpler because it only requires the definition of the potential energy and bypasses discretization and assembly steps”.
9. 10. 11. Thanks for pointing out this mistake. We already corrected it.

12. We rewrote the sentence to express the idea clearer - “Because the number of data used for training is different, different epoch numbers are set for the corresponding dataset, i.e. 60, 100, and 120.”

13. In our paper, we use \$u\$ to denote the general or analytical displacement, \$\hat{u}^L\$ to denote the unconstrained displacement (the output displacement of neural networks), and \$\hat{u}\$ to denote the constrained displacement which has the general form expressed in equation (10).

14. We have added the information for this figure - “The exact solutions and neural network solutions using \textit{tanh} activation function and Dataset 1”

15. 16. We corrected it in the text.

17. We added Figure 12 for the exact solutions and substituted Figure 13c to the point-wise error ux, uy, u(r) for more clarity as to the reviewer’s suggestion.

18. Normally, the convergence of loss is plotted with the semilogx scale (not semilogy for the x-axis as the reviewer’s comment). However, in our case we want to present the error in L2 norm and energy norm in loglog scale as well. Therefore, to be consistent we plot the convergence of loss with semilogy too.

19. We already changed it to “traction” according to the reviewer’s suggestion.

20. We already rewrote the sentence - “In contrast, the activation functions of the neural networks in the P-DEM easily fulfill the C^1 requirement”

21. It is sometimes called the material length scale parameter or the strain gradient coefficient. We modified our paper and only used the term “strain gradient coefficient” for consistency.

22. We already checked the equations again. They are correct.

23. We already corrected the mistake.

24. We corrected the caption of Fig.18.

25. In index notation, this is the trace of the strain matrix.

26. We already corrected the typo.

27. Thank you for your careful reading. We have gone through the entire manuscript to eliminate as many English mistakes as possible.

Reviewer #3

The authors leverage recent advances in Physics-Informed Neural Networks (PINNs) and the related Deep Energy Method (DEM) to predict the mechanical response of elastic materials to both displacement and loading boundary conditions; the case of strain gradient elasticity is also considered. New in this work, the authors map the physical coordinates of the material to a reference unit-square domain (hence "Parametric" Deep Energy Method, P-DEM), which increases the robustness and convergence speed of the method, and facilitates further training of the same network on similar problems. The present study is a valuable demonstration of the utility of machine learning frameworks to solve/simulate physical problems, and is a logical extension of the authors' previous work.
I have the following comments and questions, which if addressed may help clarify some details of the approach and aid in the interpretation of the results:

Thank you very much for your encouraging feedback and positive comments: Below, kindly find our responses.

Comment 1: The word "for" should be added in the title: "...accounting *for* strain gradient effects".

Our Response: Thank you for your suggestion. We have already changed the title as your correction.

Comment 2: The manuscript contains many spelling and grammatical errors throughout. Please read through and revise carefully.

Our Response: We have revised the manuscript and corrected all the mistakes as much as possible.

Comment 3: How do you choose the "best" NURBS mapping between the physical and parametric domains? In the examples of Figure 4 as well as each of the application problems, the mapping is intuitively chosen such that 4 of the corners in the physical domain always map to the 4 corners of the parametric domain. However, if a different mapping was chosen (e.g. the physical boundary could be mapped to the unit-square boundary in equal arc-length segments, without regard for corners), how would this affect the optimization and accuracy of the results? This is especially relevant for the eventual application of this method to real-world geometries that are much more amorphous. For example, how to best map the body shown in Figure 5 to a unit square is not obvious, so the sensitivity of the results to different mapping choices in such a case should be examined.

Our Response: In general, there is no unique best NURBS parameterization, just like there is no unique best mesh in finite elements, although some measures of quality (such as minimum Jacobian) can be used. We think that this method of combining NURBS mapping with DEM should be less sensitive to mesh distortions compared to IGA or FEM, although it may take some work to do a detailed study. In any case, the general procedure is to decompose the boundary into four curves and map those to the four sides of the square and if the domain has four corners it is natural to map them to the corners of the square. For more complicated geometries, we think a multi-patch approach is needed. The decomposition of the domain into quadrilateral or hexahedral patches is then in some sense related to finding a coarse quad or hex mesh. We think this method could work by using multiple smaller neural networks for each patch together with some way of enforcing continuity either by the penalty or Discontinuous Galerkin type approach and this could be done in future work.

Comment 4: Why was the 2-30-30-30-2 network structure chosen? Was this compared to other network structures?

Our Response: We practiced with several structures to choose the optimal number of hidden layers and neurons. As reported in our previous study [1], three hidden layers are sufficient to yield accurate solutions in 2D problems. The number of neurons is selected based on manual search, which means we run multiple trials in a single training job with each set of hyperparameters, and then we tune them until the best result is achieved. The way we tune the hyperparameters is we try to tune only one

parameter at one time and fix the others to see how to influence that parameter creates in our problem, and we move to the next parameter with the same method. Although there is no rule of thumb to choose such neural network structures, the selection of the width and depth of a network and other hyperparameters can be done in the spirit of optimization. In fact, these hyperparameters can be tuned automatically by Bayesian optimization, genetic algorithms, artificial neural networks, random search, or grid search. However, this topic does not contribute to the paper's objective, and it can be done in a separate publication.
[1] - https://doi.org/10.1016/j.euromechsol.2019.103874
[see Page 15]

Comment 5: In Equation 10, how are A and B chosen and computed? The authors state that "[t]he terms A( ) and B( ) are chosen in such a way that ^u(X) must satisfy the boundary conditions at certain points"; however, this seems not to be explained in detail either here or in the original DEM paper (2020).

Our Response: We have rewritten the explanation and cited the original papers.
"The term A refers to a smooth extension of the boundary data and B is a smooth distance function. Both are chosen in such a way that \hat{u}(X) must satisfy the boundary conditions at certain points. The discussion of how to choose these terms can be found in the works of Lagaris [10] and Berg & Nystrom [15].
[10] https://doi.org/10.1109/72.712178
[15] https://doi.org/10.1016/j.neucom.2018.06.056
[see Page 7]

Comment 6: For all of the benchmark problems, P-DEM had better performance than DEM. Are there any situations where P-DEM might have worse performance than the original DEM?

Our Response: We believe that there are some situations in that DEM will outweigh P-DEM especially in higher dimensions that it takes more time for the mapping in the preprocessing step. However, in the final goal, we want to present the capability of reusing training knowledge for different geometries by training on the same mapping domain. We plan to do this in future work.

Comment 7: How do you anticipate the P-DEM would perform for a nonlinear elastic material undergoing large deformations?

Our Response: In our work, we still keep the spirit of DEM that we use the total potential energy of a system as the loss function, and we try to minimize this loss. This method will maximize its potential as long as a material is characterized by a free energy density. Moreover, the mapping from physical space to reference space does not alter the spiritual method. Therefore, we see no obstruction in applying P-DEM for a nonlinear elastic material undergoing large deformations.

Comment 8: Initialization is only mentioned for the strain gradient elasticity problem (random normal weights and zero biases). Are the weights and biases initialized the same way in each of the other application problems?

Our Response: We have used the Xavier uniform distribution for weights in some other works. But we found that there is not much difference in the final results, at least in our examples.

Comment 9: Is there any regularization used during the optimization? If so, what kind, and what are the regularization parameter values?

Our Response: In the elasticity part, we directly impose boundary conditions to the displacement a priori (Equations 38, 39, 44, 46), and therefore the optimization, in this case, turns into the unconstrained optimization. We do not need to apply regularization to the objective function. However, in strain-gradient elasticity, we add two regularization terms to the loss function in equation (74). The parameters values are already given in the text. Note that if we use the dropout technique, which is normally applied in classification problems, the results will not be correct due to the loss of information about the ansatz functions of some dropout neurons.

Comment 10: It may be good to include in the Conclusion a summary of limitations and future potential improvements/extensions/variations of the method, as well as some practical applications of the method to real-world problems.

Our Response: We have proposed a P-DEM which is an extension of the original DEM and applied it to problems in elasticity and strain-gradient elasticity. Instead of training the DNN in the physical domain, we train the network in a parameter domain. The correspondence between two domains associated with the forward and backward mapping is achieved through NURBS basis functions commonly used in IGA. In this context, Gauss quadrature is employed to calculate the potential energy that is considered as the loss function. Although it requires some effort in dealing with mapping using NURBS to create a reference domain, it as a result brings some benefits to the training due to the same unit domain. One advantage of the presented P-DEM over the original DEM is that the inputs are automatically normalized within the natural domain, and therefore it prevents the gradient vanishing that might appear when the hyperbolic tangent is used as an activation function. Another advantage that could be exploited is using transfer learning for other problems having different geometries \footnote{The preliminary work of this application can be found in the Appendix}. Moreover, the knowledge when solving previous problems can easily be retained for the next training through the concept of transfer learning, 'naturally' integrated into P-DEM, which finally leads to better training performance.
The robustness of the method is demonstrated through several numerical examples in linear elasticity with different geometries. Furthermore, we presented an example for strain gradient elasticity whose strong form is defined by a third-order PDE, which

cannot easily be solved in FEM due to the C^1-continuity requirement. The numerical results show that P-DEM achieves the optimal value of loss energy faster than DEM and yields accurate results in presented benchmark problems.
[see Page 31]

Comment 11: Will the code/implementation of this method be published as supplementary material and/or maintained with proper version control (e.g. on Github)? Is it possible to provide scripts that could be used to independently reproduce all the results in this study?

Our Response: Yes. Like DEM work, we will publish our code on Github so that people who are interested in our method can use it for their purpose. We believe that sharing code will spread our work to the community faster. However, we only share the code after the paper gets accepted. We have added the temporary link in the appendix section.
[see Page 34]