Independent Research Project Plan

Applying machine learning to the optimisation of numerical simulations of coupled processes in fractured media

Ye Liu

Supervisor: Adriana Paluszny

June 2019

1 Introduction

1.1 Importance

Coupled processes in fractured media are a series of complex mechanics problems [1], which are usually described by Partial Differential Equations (PDEs). PDEs are usually solved numerically because of the lack of analytic solutions for complex problems [2]. Finite difference, finite element, finite volume, and Monte-Carlo method are commonly used numerical methods. In computational solid mechanics field, finite element method (FEM) is used the most frequently. Due to the requirement of high fidelity simulation, the size of the problems to be solved has been growing larger and larger [3]. Traditional deterministic methods such as the Finite Element Method (FEM) highly depends on computation resources. The time to solve large size real-world problems would become extremely long even when using High Performance Computing cluster. Therefore, it demands new techniques to solve these problems more efficiently [4].

With the development of computer technology, statistical or data-driven method have great progress in recent years. Machine learning especially deep learning techniques have been applied to solve varies complex problems successfully. These problems include image processing, target detection, speech recognition,etc. The deep learning methods are proved to be more efficient and accurate when comparing to the traditional methods in these areas. [5][6] The deep learning techniques establish a surrogate model by training a neural network using existing data. The model mapping the input (observed data) to solutions, which could be used to solve inverse problems. Thus, a much wider class of problems that were believed to be too complex to tackle are solved by using these techniques. It is deserved to apply the Cutting-edge machine learning techniques in computational mechanics fields such as FEM, which are possible to improve computation efficiency and accuracy.

1.2 FEM and Machine Learning

The FEM approximates the unknown function over both spatial and temporal domain. It divides a large system into many small parts that are called finite elements. These finite elements are firstly modeled by simple equations. Then being assembled into a larger system of equations which could model the entire problem. With variational methods from the calculus of variations, a solution is approximated by minimizing an associated error function.FEM is relatively efficient and precise. Also, the boundary conditions are easy to deal with [7].

Machine learning is a large variety of statistical methods. It includes regression (Linear regression, ridge regression, lasso regression), clustering (k-nearest neighbors), order-reducing (Principle Components Analysis), classification (Logistic Regression, support vector machines), neural networks and ensemble methods [8]. Given enough training data, these techniques could conduct different kinds of tasks using statistical methods. By tuning a number of values hyperparameters, the behaviors of methods could be controlled. Deep learning is an important branch of machine learning based on artificial neural networks. It could recognize implicit features from a large amount of training data and classify them into different classes [9].In deep learning, a convolutional neural network (CNN) is a class of deep neural networks, most commonly applied to analyzing visual imagery. Three main types of layers are used to build CNN architectures: Convolutional Layer, Pooling Layer, and Fully-Connected Layer (exactly same as regular Neural Networks). LeNet and Alexnet are typical architectures in the field of Convolutional Networks.

In computational mechanics field, by using machine learning methods, a relationship between activation and response could be established. For example, a well trained neural network could be a surrogate model for Finite Element Methods or optimizing some process of them. Researchers have done lots of work in this area, a current research review would be shown in the next section.

2 Background

Machine learning techniques have been widely used in computational mechanics field in recent years, to improve the performance of traditional numerical methods, especially in FEM [10].

A data-driven computing paradigm was proposed by T. Kirchdoerfer et al as a proof of concept for applying the data-driven method in FEM testing on a linear elasticity problem [4]. The method described in [8], G.Capuano et al used regression and support vector machine to generate a direct relationship between the element state and its forces using finite element data. To achieve real-time structural topology optimization, X.Lei et al proposed a moving morphable component (MMC)-based explicit framework for topology optimization [11]. Machine learning methods were used to improve numerical quadrature for FEA [9], Oishi et al proposed a method to enhance the numerical quadrature rule using deep learning. The neural network is also be applied to the load transfer paths analysis on plate structures. The deep learning method shows higher efficiency and accuracy comparing to FEM [12]. Some researchers try to apply a deeper neural network into the computational mechanics area. A deep convolutional neural network was developed [13] to improve the predicting accuracy of eigenvalue problems in mechanics. A deep-autoencoder was applied to approximate the large deformations of a non-linear, muscle actuated beam, which improve the efficiency real-time finite element simulations [14]. L.Liang et al proposed a deep learning based approach to estimate stress distribution [15]. The approach includes an ML-FE surrogate to conduct fast simulations. Deep learning technique is also used to improve Finite Element Analysis problem such as stress analysis. Z.Nie et al implemented a CNN (Convolutional Neural Network) based method to conduct stress field prediction, which achieved higher accuracy compared to traditional FEA [16].

These works show a strong potential in applying machine learning methods into traditional Finite Element Methods. In this project, we would focus on optimizing some computational heavily processes of FEM. The specific processes would be described in the next sections.

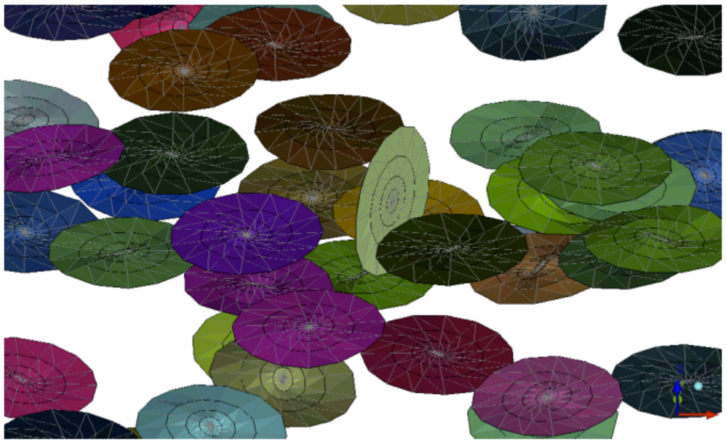

Figure 1: Detail of the fracture pattern with FEM[3]

3 Future Plan

3.1 The problem to solve

Traditional FEM methods require high computation resource when dealing with large size problems. The accuracy and efficiency of the FEM are expected to be improved.

To be more specific, there are processes in FEM that can be possibly optimized. For example, the numerical quadrature process takes an important part in the computation cost of FEM. When dealing with complex problems, solving the models of the element's internal physical field is also a heavy burden for computation, because it relies on iterative numerical methods. These problems would be the main focusing topics of the project.

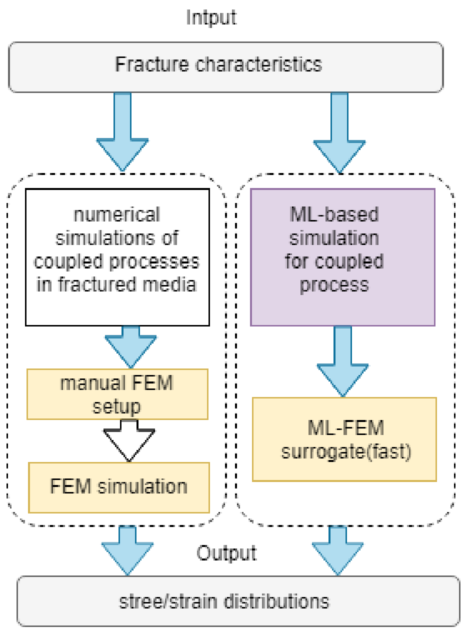
Intput
Fracture characteristics
numerical simulations of coupled processes in fractured media
manual FEM setup
FEM simulation
ML-based simulation for coupled process
ML-FEM surrogate(fast)
Output
stree/strain distributions

Figure 2: Comparison of current FEM and machine-learning based FEM

3.2 Project goals and objectives

Investigate different cutting-edge data-driven computational mechanics methods. Develop novel FEM methods with machine learning techniques. Generate a piece of code using Python based on the novel methods that can solve real world problems.

Choosing optimizing numerical quadrature process as the start point. The standard Gauss–Legendre quadrature is widely used in FEM. The accuracy and efficiency of it rely on two parameters: the number of integration points and the weights corresponding to the integration points. It would be beneficial for developing some methods which can help to choose the optimal points and weights.

After the expected methods developed, apply the method to in-house FEM codes, improve the efficiency and accuracy of FEM simulating coupled processes in fractured media.

3.3 Evaluation

The developed method would first be tested on some simple cases such as linear elasticity problem comparing with traditional FEM (use FEniCS or SfePy). The accuracy and efficiency would be the main comparison targets. If the new method performs better, then more tests and experiments would be conducted based on coupled processes in fractured media problem.

3.4 Related software and libraries

1. FEniCS Version 2019.1.0

   FEniCS is a popular open-source (LGPLv3) computing platform for solving partial differential equations (PDEs). FEniCS enables users to quickly translate scientific models into an efficient finite element code. [17]

   In this project, FEniCS would be used as a rapid testing FEM platform. The novel FEM would be developed by combining cutting-edge machine learning methods into the FEniCS codes.

2. SfePy Version 2019.1: Simple Finite Elements in Python

   SfePy is a software for solving systems of coupled partial differential equations (PDEs) by the finite element method in 1D, 2D and 3D. It can be viewed both as black-box PDE solver, and as a Python package which can be used for building custom applications. The word "simple" means that complex FEM problems can be coded very easily and rapidly [18].

   SfePy is lighter than FEniCS. It could be used when doing or developing more simpler cases.

Related libraries are listed below:

1. NumPy Version 1.16.3

   NumPy is the fundamental package for scientific computing with Python. It contains among other things: a powerful N-dimensional array object; sophisticated (broadcasting) functions; tools for integrating C/C++ and Fortran code; useful linear algebra, Fourier transform, and random number capabilities [19].

2. SciPy Version 1.3.0

   SciPy is a free and open-source Python library used for scientific computing and technical computing. SciPy contains modules for optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers and other tasks common in science and engineering [20].

3. PyTorch Version 1.1

   PyTorch is a Python package that provides two high-level features: Tensor computation (like NumPy) with strong GPU acceleration; Deep neural networks built on a tape-based autograd system [21].

   It is a widely used Python-based machine learning framework. In this project, the codes related to machine learning techniques would be based on PyTorch.

3.5 Schedule

Trello was used to track the process of the whole project. The detailed project plan is described below.

Part1: 1st Jun - 28th Jun

1. Finish the work plan and literature review.

   Read cutting-edge papers related to machine learning techniques applied into FEM topic from peer-reviewed journals and conferences. Select feasible methods as a possible start point.

2. Learn more about FEM and machine learning

   Learn Finite Element Methods systematically, understand the basic theory and write some codes for a simple case for deeper understanding. Review the machine learning methods, get clear about the principle of regression, support vector machine and neural network.

3. Prepare for Implementing

   Install all the related software and libraries. Solve potential platform problems. Do the tutorial of each software to get familiar with them.

Part2: 29th Jun - 29th Jul

1. Implement the selected methods

   - Generate a general dataset. Because this project focuses on improving FEM performance by machine learning techniques, using a general dataset would be convenient and fair to test different kinds of methods. To generate this dataset, some classical and simple FEM cases (such as linear elasticity) could be used as a data source.

   - Train the model and analyze the results, investigate and evaluate the performance of these methods. The training and validation process would both be done on the dataset we generate in the last step.

   - Select the best one as the start point of our original method.

2. Tuning the best method

   Tuning the hyperparameters of the final selected method in detail to get the best performance. Test the generalization ability of the method using different validation methods.

3. Start writing report

   Summarize the current results and start to write the final report.

Part3: 30th Jul - 30th Aug

1. Apply the method to the fracture problems

   The most interesting problem for this project is the FEM simulations of coupled processes in fractured media. The in-house codes have been developed and ready for optimizing. Try to combine the method investigated in the previous work into these codes. Test the performance comparing to the original one using the same case when training, to debug and make sure there are no unexpected errors in the codes.

   Design the experiments using new unseen data. Then test the codes in different scenarios to investigate the generalization ability of this method.

2. Results Analysis

   Collect the results and do analysis. Evaluate the method and draw the conclusion.

3. Finish the final report

   Summarize the whole project and finish the remaining report. Revise the report to a final satisfying version.

4. Prepare for the presentation

   Extract the highlights of the project and make the PPT (or maybe poster). Practice and mock the presentation.

3.6 Progress to date

The tasks in Part1 are almost done up to now. In addition to the preparation works, some implementations work have been done.

An novel method called 'Smart Integration (SI)' is developed using Python. The SI method allows the integration function takes a specified number of integration points and corresponding weights. These two arguments would be the outputs of a neural network, which outputs the optimal number of points and weights.

The method is implemented by re-developing the source codes of numerical quadrature function in Scipy and Numpy. In the next steps, the main tasks are generating the training data and training the neural network. Then the SI would be completed. After that, the SI method would be combined with FEniCS (or SfePy) and tested on simple linear elasticity problems.

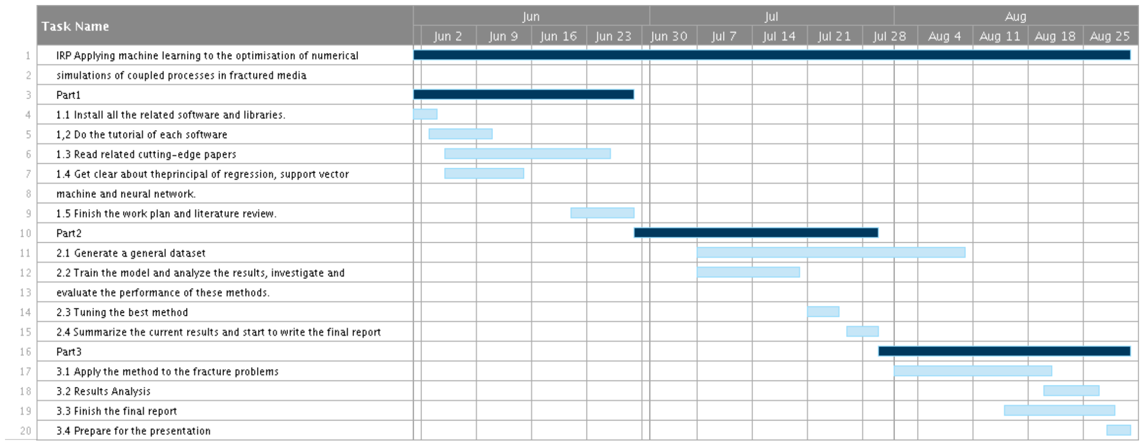

Figure 3: Project Plan Gantt chart

References

[1] S. Salimzadeh, A. Paluszny, H. M. Nick, and R. W. Zimmerman, "A three-dimensional coupled thermo-hydro-mechanical model for deformable fractured geothermal systems," *Geothermics*, vol. 71, p. 212–224, 2018.

[2] A.Paluszny, S.Salimzadeh, and R. W. Zimmerman, "Finite-element modeling of the growth and interaction of hydraulic fractures in poroelastic rock formations," *Hydraulic Fracture Modeling*, p. 1–19, 2018.

[3] A.Paluszny, R. Thomas, and R. Zimmerman, "Finite element-based simulation of the growth of dense three-dimensional fracture networks," *52 US Rock Mechanics / Geomechanics Symposium*, 2018.

[4] T. Kirchdoerfer and M. Ortiz, "Data-driven computational mechanics," *Comput. Methods Appl. Mech. Engrg*, vol. 304, pp. 81–101, 2016.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[6] A. Graves, A. rahman Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," *IEEE international conference on acoustics, speech and signal processing*, pp. 6645–6649, 2013.

[7] D. L. Logan, *A First Course in the Finite Element Method, Fifth Edition.* Global Engineering, 2011.

[8] G. Capuano and J. J. Rimoli, "Smart finite elements: A novel machine learning application," *Comput. Methods Appl. Mech. Engrg*, vol. 345, p. 343–381, 2019.

[9] A. Oishia and G. Yagawa, "Computational mechanics enhanced by deep learning," *Comput. Methods Appl. Mech. Engrg*, vol. 327, p. 327–351, 2017.

[10] W. K. Liu, G. Karniakis, S. Tang, and J. Yvonnet4, "A computational mechanics special issue on: data-driven modeling and simulation—theory, methods, and applications," *Computational Mechanics*, 2019.

[11] X. Lei, C. Liu, Z. Du, W. Zhang, and X. Guo, "Machine learning-driven real-time topology optimization under moving morphable component-based framework," *Journal of Applied Mechanics*, 2019.

[12] Q. Wanga, G. Zhangb, C. Sunc, and N. Wua, "High efficient load paths analysis with u* index generated by deep learning," *Comput. Methods Appl. Mech. Engrg*, vol. 344, pp. 499–511, 2019.

[13] D. Finol, Y. Lu, V. Mahadevan, and A. Srivastava, "Deep convolutional neural networks for eigenvalue problems in mechanics," *arXiv [physics.comp-ph]*, 2018.

[14] F. Roewer-Despr, N. Khan, and I. Stavness, “Towards finite-element simulation using deep learning,” *15th International Symposium on Computer Methods in Biomechanics and Biomedical Engineering*, 2018.

[15] L. Liang, M. Liu, C. Martin, and W. Sun, “A deep learning approach to estimate stress distribution: a fast and accurate surrogate of finite-element analysis,” *J. R. Soc. Interface*, vol. 15, 2017.

[16] Z. Nie, H. Jiang, and L. B. Kara, “Stress field prediction in cantilevered structures using convolutional neural networks,” *arXiv.org*, 2019.

[17] FEniCS. (2019) Fenics website. [Online]. Available: https://fenicsproject.org

[18] Sfepy. (2019) Sfepy website. [Online]. Available: http://sfepy.org/doc-devel/index.html

[19] Numpy. (2019) Numpy website. [Online]. Available: https://www.numpy.org

[20] SciPy. (2019) Scipy website. [Online]. Available: https://www.scipy.org

[21] PyTorch. (2019) Pytorch website. [Online]. Available: https://pytorch.org