Title: Implementing integrated machine learning strategies to accelerate high accuracy fracture growth simulators

Supervisor: Dr. Adriana Paluszny

Rationale:

Current fracture growth and propagation analysis is performed using Finite-Discrete Element Models (FDEMs) such as the Hybrid Optimization Software Suite (HOSS) (Knight, et al., 2014) or the Imperial College Geomechanics Toolkit (ICGT) (Paluszny & Zimmerman, 2011). These require fine spatial resolution on the cracks, and then perform forward time evolution in accordance to Newtonian mechanics, fluid flow and thermal effects, among others.

Whilst accurate, these models often take prohibitive amounts of time to complete due to the requirement for the solution of several Partial Differential Equations (PDEs) (Paluszny, et al., 2018). Furthermore, due to the infeasibility of knowing the exact distribution of cracks and microcracks in a given sample, for statistical rigor several models have to be run with perturbations on the initial state, further compounding the time inefficiency of these models (Moore, et al., 2018).

As a result of these factors, there is room for improvement with regard to the computational efficiency of these simulators.

Objectives:

The objective for this project is to design and implement a solution to leverage machine learning approaches to increase the computational efficiency of fracture propagation simulators, specifically the ICGT. This would be done by replacing the evaluation of PDEs in the simulator with a machine learning algorithm trained on previous runs of said simulator.

Literature Review:

Extant Approaches:

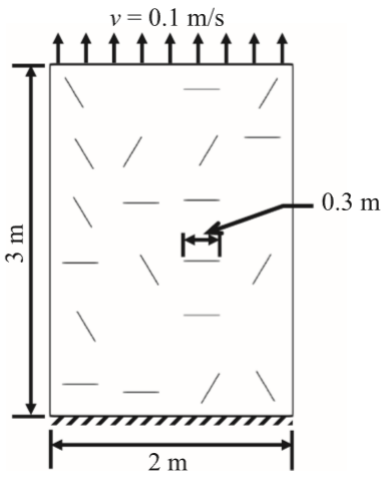


Figure 1: A schematic representation of the abstraction designed in (Hunter, et al., 2019) Source: (Hunter, et al., 2019)

(Hunter, et al., 2019) has the most in depth discussion of alternate approaches to the simulation of fracture propagation. The paper designs an abstraction of the system in order to apply these approaches. The abstraction restricts the domain to two dimensions, populates the domain with a number of one dimensional cracks at one of three orientations, and only considers Mode I (opening) damage. The system has a constant tension applied vertically, and the time to failure is predicted, where failure is defined when a contiguous crack spans the horizontal domain. See figure 1 for a schematic representation. This is a somewhat limited model; however, it allows the comparison of several approaches to the problem on level footing.

In order to train the machine learning aspects of the following approaches, the initial states were run on an FDEM, specifically HOSS, in order to generate a labelled training set.

Orthogonal Projection (OP)

The first approach described in the paper further restricts the model to only consider the orthogonal projection of each crack with respect to the applied tension, which in this case means it only considers the horizontal extent of each crack due to the vertically applied tension. This is due to an *a priori* assumption that the orthogonal projection is the primary driver of Mode I damage.

The approach then uses data from the simulator to train the fracture propagation at each time step. This fracture propagation, by the nature of the abstraction used in this case, is only oriented horizontally, leading to predicted failure paths that span the domain in straight, horizontal lines.

Micro-Crack Pair Informed Coalescence (McPIC)

This approach takes the *a priori* assumption that crack interaction and coalescence is the primary factor in system evolution. The method organises the cracks from the simulator into crack tip pairs, which are then used to train the ML algorithms by tracking their coalescence over time. This training data is then used to predict when two cracks will coalesce into one crack, without the intervening evolution that would actually lead to coalescence.

To represent this behaviour, each crack at system initialisation is represented by a vertex, and edges represent when two cracks have coalesced (in this model, the boundaries are also modelled as cracks, therefore when the two boundaries are connected the system has undergone catastrophic failure).

Network-based Fracture Process Zone (NFPZ)

This method is deterministic and not a machine learning method, however it is similar in principle to the final approach so it will still be discussed here.

The method defines a "process zone" around the crack tips of the horizontally aligned cracks. The horizontally aligned cracks are selected because they are presumed to be a part of any domain spanning fracture due to their dominance in Mode I damage, and this selection process both helps unphysical solutions where horizontal fractures are excluded, and helps improve performance by limiting the number of process zone interactions to be considered (though this efficiency consideration is not relevant for systems of this size).

The process zone represents the region of increased material stresses due to the presence of a fracture tip. If the process zones of two tips overlap, they coalesce. Further analysis can be performed to determine the likely fracture path if there are multiple domain spanning fractures after conclusion of all coalescence operations.

Ellipse Process Zone (EPZ)

The EPZ method is very similar to the NFPZ method with one notable difference; instead of the process zone being a statically defined structure, it is an ellipse, the size of which is determined heuristically by learning from HOSS simulations. The specifics of said heuristic approach are complex and irrelevant to this project.

The effective result of this approach is that fracture tips with a higher SIF (the SIF being approximated heuristically), have larger process zones, and therefore are more likely to coalesce with other fractures. It is salient to note that due to the usage of HOSS data for the process zones, this method, by contrast to NFPZ, does have time dependence. This is because the process zones are recalculated at each time step, in a manner attempting to mimic the operation of HOSS itself.

Evaluation

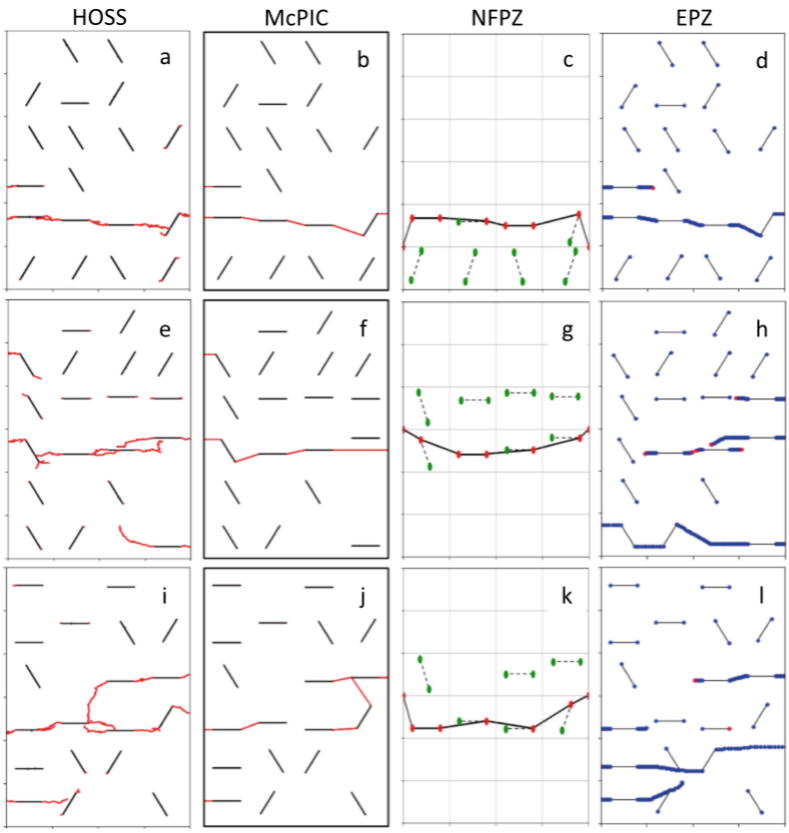
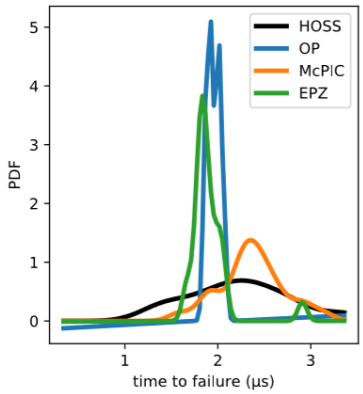


Figure 2: The results of each approach (excluding OP) regarding their predicted failure paths, in addition to the HOSS results for a given initial configuration. Source: (Hunter, et al., 2019)

The OP method almost exclusively produces times to failure of 2µs, which does not follow the distribution generated by HOSS. Furthermore, the restriction to horizontal propagation leads to thoroughly unphysical fracture paths.

The McPIC approach is more complex than the OP approach and contains more data implicit to the generated connectivity graph than simple time to failure. The McPIC approach also best approximates the time to failure as generated by HOSS, being the only method that returns values away from 2µs. However, this method still does not reproduce an accurate approximation of fracture evolution in space, due to the restriction to linear paths between vertices.

Figure 3: A histogram showing the predicted time to failure for each approach that considers temporal data. Source: (Hunter, et al., 2019)



The NFPZ method can generate spatial fracture predictions, which are inferior to McPIC and EPZ, and does not predict any temporal behaviour whatsoever. This is an ineffective method.

EPZ is superior to NFPZ, however its performance is effectively the same as, if not worse than, the OP method. Considering the significantly more complex nature of EPZ compared to OP, this is an ineffective strategy.

Figure 2 shows the failure paths predicted by the methods compared to that predicted by HOSS for three representative initial configurations. Despite largely identifying the cracks constituting the end fracture paths, the limitations of each approach mean that the microscale accuracy is lost.

Figure 3 shows the predisposition of OP and EPZ to predict 2µs as time to failure. McPIC more closely follows the HOSS prediction but still does not maintain a high degree of concurrence.

Conclusions Regarding the Current State of the Field

In summary, whilst these methods produced quite accurate results with regard to the metrics that the authors assigned, with the McPIC approach being the best, none of the approaches discussed here truly fulfil the role that a fully formed fracture simulator can. A large part of this deficiency is the restriction to two dimensions. Furthermore, despite some promising results with regard to the accuracy of macro-scale quantities, such as time to failure and total material damage, the micro-scale accuracy of the physical fractures and their paths taken is still crude compared to those returned by a full simulator. Finally, these methods totally omit any consideration of mode II and III damage. Whilst this is a reasonable abstraction given the system the authors have designed; this may not be the case in certain real-world systems.

Through the combination of these issues, there are many investigations that could not be aided by these applications of machine learning techniques. Therefore, it is proposed that instead of using machine learning to replace most of the physical knowledge inherent to a fracture simulator, one could use machine learning techniques integrated into the simulator in order to accelerate the simulator's execution speed. It is hoped that this would combine the best elements of both simulation strategies; namely the resolution and accuracy of the standard fracture simulator, with the speed of the machine learning approaches discussed in the paper.

Approach:

Simulator Background:

Before investigating how machine learning could be leveraged in order to accelerate a fracture simulator, one must investigate the simulator itself. A fracture simulator incorporates a lot of physics into its operation, namely Newtonian mechanics, fluid flow and thermal effects. The inclusion of these factors leads to a high computational cost, due to the requirement of solving several PDEs. The details of such PDEs are omitted from this project plan as they are superfluous to the operation of the proposed strategy. The salient aspect is that the propagation of any given fracture tip is calculated with respect to the Stress Intensity Factors (SIFs) K_I, K_{II} and K_{III}, which themselves are calculated by the aforementioned PDEs.

Given these quantities, the simulator will then step forward through time and model the evolution of these fracture tips and therefore the growth of the fractures they constitute.

Solution Proposal:

It is proposed here that instead of solving the computationally expensive PDEs to derive the SIFs, one could train a machine learning algorithm on a labelled set of previously solved PDEs, in order to then predict the SIFs for new fracture tips, and then to implement the evolution of these tips. It is anticipated that this will make the simulator significantly less computationally intensive with minimal loss of accuracy with respect to the current method of solving the PDEs.

The features that will be used to predict the SIFs are expected to be the location as a three dimensional vector, the previous displacement of the tip again as a three dimensional vector, and the three components of the previous SIF, leading to an initial total of nine features. There is scope for expansion by including further features of the tip in question, as well as the features of neighbouring fracture tips within a given radius of influence.

As for the specific machine learning algorithm, given the limited number of starting features and output variables (the three components of the new SIF), and the lack of any geometric dependency of

the different features, the intention is to implement a neural network. However, it may become clear through the course of the project that an alternate algorithm (such as a random forest or convolutional network) would be preferable, therefore this selection is liable to change.

Milestones:

Literature Review

Complete a review of the literature surrounding the field, and identify the strengths and weaknesses of the approaches discussed. Attempt to identify papers of direct relevance to the project, and also papers on machine learning in a broader sense, as well as possible applications. Literature regarding the physics of fracture propagation will not be considered, as that knowledge is irrelevant to the implementation of the project.

At the time of writing, this is the point that development has reached – the literature has been reviewed, and a concept for the project has been conceived.

Specific requirements and goals for final solution

Generate a concise list of the requirements that the project is expected to fulfil, in addition to potential extension objectives.

High level design of solution

At this stage of development, it is expected that there will still be very little actual written code. Instead, extended time will be spent analysing the code that this project will operate inside, and the data structures utilised therein.

Using this information, a detailed design for the solution will be developed, including the specifics of the machine learning algorithm employed. Due to this, it is expected that there will be significant testing performed at this stage. The deliverable here will be a comprehensive design document.

First solution prototype

This is the point at which significant quantities of code will be written. The code will be a direct implementation of the design generated in the previous section, with as few deviations as possible. The deliverable from this stage will be a module that can hopefully reproduce the desired functionality, and if it cannot, then it will undergo extensive debugging prior to delivery.

In the case of debugging, it is expected that the design document may have to be revisited in order to implement changes that result from issues not foreseen at the planning stage. This stage of development will continue as long as is necessary for the solution to achieve the desired functionality.

It should be noted that at this stage, desired functionality means the return of vaguely correct values – accuracy will not be considered at this point in development.

Design and implementation of testing suite

With the completion of a functional prototype, a testing suite will be designed in order to test the accuracy and efficiency of the solution. It is expected that these two factors will be inversely proportion at the point of maximum optimisation, as further accuracy will almost certainly require further computational time.

Nonetheless, several strategies will be implemented here to optimise the solution, such as hyperparameter tuning of the machine learning algorithm, loop optimisation, data structure

streamlining and even implementation of alternate machine learning algorithms altogether, if performance is not at the desired level.

The deliverable from this stage will be a review of all tests conducted and their associated results. At the conclusion of this stage it is likely that there will be several concurrent forms of the solution, using slightly alternate strategies.

Final program completion

At this point of development, the final solution will be largely written already. The objective here is to consolidate what was determined to be the optimal approach from the previous stage, and to ensure it is sufficiently documented and sustainable (even though these aspects will also be maintained throughout the entire development process).

The deliverable here will be the final state of the code base. It is expected that no further changes to the code will occur after this point.

Draft report

With the conclusion of direct work on the code, the work on the report will commence. It is hoped that there will be a surplus of material written here, which will then be cut down through the editing process. Deliverable is the draft report.

Final report completion

At this final level, the objective is to edit the draft down so that it is concise and effective. Formatting and other such secondary issues will also be addressed here. The deliverable from this is the final state of the Independent Research Project.

References:

Hunter, A. et al., 2019. Reduced-order modeling through machine learning and graph-theoretic approaches for brittle fracture applications. *Computational Materials Science,* Issue 157, pp. 87-98.

Knight, E. E., Rougier, E., Lei, Z. & Munjiza, A., 2014. *Hybrid Optimization Software Suite,* s.l.: US DoE, OSTI.

Moore, B. A. et al., 2018. Predictive modeling of dynamic fracture growth in brittle materials with machine learning. *Computational Materials Science,* Issue 148, pp. 46-53.

Paluszny, A., Matthai, S. K. & Hohmeyer, M., 2007. Hybrid finite element–finite volume discretization of complex geologic structures and a new simulation workflow demonstrated on fractured rocks. *Geofluids,* Issue 7, pp. 186-208.

Paluszny, A., Salimzadeh, S. & Zimmerman, R. W., 2018. Finite-Element Modeling of the Growth and Interaction of Hydraulic Fractures in Poroelastic Rock Formations. In: *Hydraulic Fracture Modeling.* s.l.:Elsevier, pp. 1-19.

Paluszny, A., Tang, X. H. & Zimmerman, R. W., 2013. Fracture and impulse based finite-discrete element modeling of fragmentation. *Comput Mech,* Issue 52, pp. 1071-1084.

Paluszny, A. & Zimmerman, R. W., 2011. Numerical simulation of multiple 3D fracture propagation using arbitrary meshes. *Comput. Methods Appl. Mech. Engrg.,* Issue 200, pp. 953-966.

Schwarzer, M. et al., 2019. Learning to fail: Predicting fracture evolution in brittle material models using recurrent graph convolutional neural networks. *Computational Materials Science,* Issue 162, pp. 322-332.

Valera, M. et al., 2018. Machine learning for graph-based representations of three-dimensional discrete fracture networks. *Computational Geosciences,* Issue 22, pp. 695-710.