Computational Geosciences (2020) 24:1429–1444
https://doi.org/10.1007/s10596-020-09962-5

ORIGINAL PAPER

Physics-informed machine learning for backbone identification in discrete fracture networks

Shriram Srinivasan[1] · Eric Cawi[2] · Jeffrey Hyman[3] · Dave Osthus[4] · Aric Hagberg[5] · Hari Viswanathan[3] · Gowri Srinivasan[6]

Received: 15 August 2019 / Accepted: 27 March 2020 / Published online: 17 May 2020

Abstract

The phenomenon of flow-channeling, or the existence of preferential pathways for flow in fracture networks, is well known. Identification of the channels ("backbone") allows for system reduction and computational efficiency in simulation of flow and transport through fracture networks. However, the purpose of machine learning techniques for backbone identification in fractured media is two-pronged system reduction for computational efficiency in simulation of flow and transport as well as physical insight into the phenomenon of flow channeling. The most critical aspect of this problem is the need to have a truly "physics-informed" technique that respects the constraint of connectivity. We present a method that views a network as a union of connected paths with each path comprising a sequence of fractures. Thus, the fundamental unit of selection becomes a sequence of fractures, classified based on attributes that characterize the sequence. In summary, this method represents a parametrization of the sample space that ensures every selected sample sub-network (which is the union of all selected sequences of fractures) always respects the constraint of connectivity, demonstrating that it is a truly physics-informed method. The algorithm has a user-defined parameter which allows control of the backbone size when using the random forest or logistic regression classifier. Even when the backbones are less than 30% in size (leading to computational savings), the backbones still capture the behavior of the breakthrough curve of the full network. Moreover, there is no need to check for path connectedness in the backbones unlike previous methods since the backbones are guaranteed to be connected.

Keywords Discrete fracture networks · Fractured porous media · Physics-informed · Data-driven · Machine learning · Classification

1 Introduction

The simulation of fluid flow and the associated transport of dissolved chemicals through fractured media is an inherently multiscale phenomenon as the length scale of the fractures spans several orders of magnitude [1]. Geological sites comprised of granite, basalt, and shale are typical examples of such fractured media, possessing an interconnected network of fractures embedded within a rather impermeable rock matrix. The simulation of flow through these networks is necessary for several subsurface applications including hydrocarbon extraction [2–4], environmental restoration of contaminated fractured media [5–7], CO_2 sequestration [8], and aquifer storage and management [9]. Discrete fracture network (DFN) models are one tool to model flow and transport through fractured media. In it, individual fractures are represented as $N-1$-dimensional objects, e.g., lines in two dimensions or planar polygons in three dimensions, embedded within an N-dimensional space, and each fracture is assigned a shape, location, and orientation within the domain based on a site characterization. Each fracture is meshed for computation and the governing equations for flow and transport are numerically integrated on the network. However, the choice to explicitly represent fractures and network structure makes DFN models computationally expensive for networks of even modest size.

As fracture locations and attributes are modeled stochastically, a large number of network realizations are required to bound system uncertainty, which exacerbates the problem of computational expense. The basic principle is that an ensemble is generated using computational models and the Monte Carlo (MC) method is used to determine the likelihood of a scenario. While the standard MC method is

✉ Shriram Srinivasan
shrirams@lanl.gov

Extended author information available on the last page of the article.

the baseline method in uncertainty quantification, it suffers from slow convergence. Variants of MC, such as Multi-Fidelity Monte Carlo (MFMC)[10] and Multi-Level Monte Carlo (MLMC) [11], have been developed to approximate the ensemble mean of the quantity of interest with small variance and faster convergence rates. However, in the case of fracture networks, MLMC [12] relies on several levels of accuracy in the space discretization and utilizes more samples at the cheaper (less accurate) levels while improving the estimates with few samples at the finer levels. The method presumes the existence of a solver and a meshing strategy that can use very coarse meshes, but with fracture networks, meshing remains a major challenge. MFMC [13], on the other hand, utilizes two or more models with varying levels of fidelity and computational efficiency, and the requirement then is lower fidelity models that are highly correlated but significantly increase computational performance. Development of reduced order models or low-fidelity models is itself a challenging issue in fracture networks, and this context provides a segue into an account of their development. Another method to reduce the computational expense is to upscale fracture networks to continua with effective properties [14–17], but it fails to capture the physics accurately.

However, the knowledge that there exist preferential flow pathways due to channeling effects [18–21] in fractured media implied that there are subnetworks within the flow domain that determine overall transport behavior. In turn, significant computational savings are possible if these dominant paths or "backbone" of the network could be identified, for one could then restrict the domain to the identified subnetwork alone and retain a good approximation to the quantities of interest such as the the first passage time distribution of solutes passing through the network. Ideally, one would like to maximize the number of fractures removed from the network while minimizing the discrepancy with the true quantity of interest. However, in the construction of a backbone through the removal of fractures, the most important criterion is the presence of a connected path through the backbone to mimic the flow and transport through the full network. The requirement of a connected path is a constraint on our physical system that limits the space of solutions, which is the power set (set of all subsets) of the set of fractures in the full network. Studies in sparse fracture networks with fracture sizes spanning a range of length scales indicate that network structure [22–25] and flow direction [6] are more dominant in determining regions of flow channelling than geometric or hydraulic properties of the fractures.

The mathematical construct of a graph, a set of vertices connected by a set of edges, is a well-suited tool to represent this key network structure information. The mapping of a fracture network to a graph has been done in two different, but related, ways, and both avenues have been explored toward achieving system reduction. A topology-based bijective mapping (with fractures corresponding to graph vertices and fracture intersections to graph edges) was used by Hyman et al. [26] to predict first-passage times accurately. Other instances of use of this mapping in a different context can be found in the work of Ghaffari et al. [27], Andresen et al. [28], Santiago et al. [29], Sævik and Nixon [30], Hope et al. [31], Hyman and Martínez [32], and Huseby et al. [33]. Aldrich et al. [34] also introduced a weighted graph representation with edge weights based on particle transport through the network. However, in their work, the backbone identification required running simulations on the full network. Both Hyman et al. [26] and Aldrich et al. [34] considered the backbone to be the union of the shortest paths through the network, thus ensuring the connectivity of the backbones obtained.

Karra et al. [35] presented the other "hydrology-based" mapping to simulate flow and transport on the graph representation itself (see [36–39]). In this mapping, intersections in the DFN correspond to vertices of the graph, and fractures are represented as a clique of edges and edge weights are based on the hydrological properties. However, the predictions deteriorated systematically as the networks increased in size and became heterogeneous, and a systematic correction technique was employed to make the predictions accurate and fast [40]. Srinivasan et al. [41] used the same mapping by applying flow physics on the graph and thresholding the graph Darcy fluxes to extract a backbone of the DFN, thus showing that the technique is useful for applications in which the entire distribution of mass breakthrough times is of interest and not just the first passage times. They penalized subnetworks by adding more fractures and making the networks larger if the constraint of connectivity was not satisfied.

The abundance of existing simulation data that can be obtained from high-fidelity simulations on fracture networks, as well as the large number of fractures contained in each network, suggests that a data-driven (machine learning) approach could be useful to address the problem of backbone identification. In this context, the role of machine learning in DFN modeling is not limited solely to reduction of computational expense through backbone identification, for machine learning can also be used to advance our understanding of how the network determines the structure of the flow field within the DFN. There has been a surge in the use of machine learning (ML) in the geosciences [42], fueled by better sensors, improvements in large-scale simulations, and the appearance of large, publicly available datasets stored over the cloud [43]. Sources of these data include remote sensing [44, 45], in situ sensors such as weather stations or instruments on ocean buoys [46], and ensembles of climate models [47]. Aside from applications in the geosciences, machine learning has also been used successfully to model disease spread [48], soil health [49], land cover classification [50], and is even expected to aid clinical medicine [51]. These data present challenges in the form of dimensionality, high spatio-temporal correlation, dynamic domains, small sample sizes, missing data, and rare events of interest [43, 52, 53].

The data used in an ML framework in various applications are often the product of well-studied and documented physical systems, and offer a great opportunity to integrate physics with machine learning methods [54, 55]. Such methods are known as physics-informed machine learning (PIML) techniques. PIML methods [56, 57] combine information about the physics of the phenomena into the machine learning framework by viewing the governing equations or physical principles as constraints on the system or search space of solutions. Machine-learning methods, by themselves, are mathematical constructs that are agnostic to the source of data or the physical phenomena that produce it. The interest in physics-informed methods has occurred as research communities look to replace computationally expensive and complicated mathematical models with data-driven emulators that are often based on machine learning algorithms.

There are two paths to construct a PIML method, well illustrated by analogy with well-known techniques in mathematics. One can enforce constraints either by penalizing variable choices that violate it (Lagrange multipliers and penalty methods in optimization, definitions of curves as level sets), or one can parametrize the search space and define a new one by a change of variables so that there are no constraints in the new search space. An example of the latter would be the parametric representation of most well-known curves. A non-trivial example is that of numerical discretization with divergence-free or curl-free finite elements. The second method, which is considerably difficult to design and not always feasible, is nevertheless preferred when viable, and in this article, we propose such a method for fracture networks.

The first work to utilize machine learning in the context of DFN to obtain backbone subnetworks was that of Valera et al. [58]. They endowed fractures in the network with local and global topological features based on degree and centrality measures calculated from the topology-based bijective mapping, and also simple physical features based on the geometry of the fracture. The backbone fractures were selected by a classification problem posed with respect to the features, and a flow topology graph construction [34] was used to define the backbone fractures in the training set. Valera et al. found that global topological features of the network vertices (fractures) were more important than local topological features in the selection of backbone fractures. However, there were no constraints placed on the search space of fractures so disconnected subnetworks could be identified by the method. While the technique successfully produced small backbones and good agreement with the breakthrough curves, there were also instances of disconnected subnetworks, violating the primary criterion for a suitable backbone. Moreover, there were no parameters to control the size of the backbone network other than the parameters of the classification algorithm.

In a bid to eliminate these shortcomings, Srinivasan et al. [59] proposed a different definition of what constitutes the backbone, based on particle-tracking simulations of a non-reactive tracer advecting with the fluid, and introduced a non-dimensional parameter that controls the size of the backbone. Other than this change, their method is very similar to that of [58], in that there is no attempt to constrain the search space and achieve connected subnetworks. However, they do avoid the problem of disconnected subnetworks for the range of non-dimensional parameter values considered, and explain it as the result of using the precision–recall trade-off in the classification problem to their advantage. They also credited their definition of backbone as being more reflective of the physics. Thus, in the works of both Valera et al. [58] and Srinivasan et al. [59], there is no guarantee that the backbones produced will respect the constraint of connectivity. The drawback in both these methods springs from the same source, namely, that the fundamental unit of selection in the classification algorithm is a fracture; consequently, it is challenging to limit the selection to connected subnetworks.

We address this issue by viewing a DFN a union of sequences of fractures, i.e., connected paths through the network, rather than a union of individual fractures. This choice is motivated by the notion that along a particle pathline, particles travel through a sequence of fractures. Thus, the physical system dictates that a backbone is a sequence of fractures along which particles pass through the system. In terms of our classification method, the fundamental unit of selection is therefore a sequence of fractures, rather than individual fractures. Similarly, instead of features of individual fractures, features for classification are based on the sequence of fractures. This effectively restricts the sample space to ensure that every selected sample subnetwork (which is the union of all selected sequences of fractures) is connected. Thus, we use the physical principle that backbones must be connected subnetworks to constrain the solution space of our classification problem. The quality of the method is demonstrated via the identification of small backbones that are always connected (leading to computational savings), but still capture the behavior of the breakthrough curve of the full network.

In this context, a comparison with other graph-based methods of backbone identification is illuminating. Some graph-based backbone identifications are based on the network topology [26] and do produce connected backbones. However, they do not consider the physics in constructing backbones. Some other methods use the physics of the problem (flux thresholding on pipe-network model [41]) but cannot guarantee connectivity. In Hyman et al. [37], the authors actually use both the graph representations of a DFN so that they get connected backbones with a greedy algorithm that is guaranteed to terminate in finite time. However, the method gives the user no control over the size of the backbones. The machine learning method we describe here uses the graph topology mapping so that features are readily interpretable, and it uses the physics knowledge gleaned from the high-fidelity simulations. Moreover, there is a user-controlled parameter that can be set to control the size of the backbones, which are guaranteed to be connected. The physics inherited by the graphs for backbone identification procedures are limited so far to steady-state, single-phase flow. If one wants to incorporate more advanced physics like transient or multiphase flow, the machine learning framework is the natural choice, wherein it is easy to incorporate the physics into the high-fidelity model, and thus consequently, into the training data for machine learning.

2 Transport behavior in fracture networks

We represent the transport of a solute through a fracture network as a collection of passive, conservative particles moving through the network. In this section, we outline our method to use this Lagrangian approach to identify network backbones.

We use the DFNWORKS [60] suite to generate each DFN, solve the steady-state flow equations, and determine transport properties therein. DFNWORKS combines the feature rejection algorithm for meshing (FRAM) [61], the LaGriT meshing toolbox [62], the parallelized subsurface flow and reactive transport code PFLOTRAN [63], and an extension of the WALKABOUT particle-tracking method [64, 65]. FRAM is used to generate three-dimensional fracture networks. LaGriT is used to create a computational mesh representation of the DFN in parallel. PFLOTRAN is used to numerically integrate the governing flow equations. WALKABOUT is used to determine pathlines through the DFN and simulate solute transport. Details of the suite, its abilities, applications, and references for detailed implementation are provided in Hyman et al. [60].

In the DFN approach, the network of fractures is represented explicitly as intersections of objects with co-dimension 1, i.e., intersections of planar shapes such as rectangles or ellipses in three-dimensional space. Furthermore, each representative fracture is endowed with attributes such as size, orientation, aperture, and permeability. Such attributes are drawn from a probability distribution that is constructed to fit data sampled from a geological site [66, 67].

The equations governing steady flow in a fracture with uniform aperture are the aperture-averaged approximations due to Boussinesq [68] in conjunction with the assumption of incompressibility:

$$\mathbf{Q} = -\frac{b^3}{12\mu}\nabla p \tag{1}$$

$$\mathrm{div}\,(\mathbf{Q}) = 0 \tag{2}$$

where $\mathbf{Q}$ is the volumetric flow rate per unit fracture width normal to the direction of flow, ∇p the pressure gradient, μ the viscosity of the fluid, and b the aperture of the fracture. Defining $\mathbf{q} := \dfrac{\mathbf{Q}}{b}$ as the (volumetric) Darcy flux per unit cross-sectional area normal to the flow gives:

$$\mathbf{q} = -\frac{b^2}{12\mu}\nabla p \tag{3}$$

$$\mathrm{div}\,(\mathbf{q}) = 0 \tag{4}$$

The factor $\frac{b^2}{12}$ plays the role of a permeability which is why the governing equation may be thought of as Darcy's Law, expressing the proportionality between the volumetric flux and the pressure gradient. These two equations are combined to yield:

$$\mathrm{div}\left(\frac{b^2}{12\mu}\nabla p\right) = 0\,. \tag{5}$$

We assume that the domain is a cube of side L meters, and that flow is driven by the pressure gradient resulting from prescribed pressures at the two ends (faces) of the domain, with no-flow boundary conditions at the other boundaries. Aligning a Cartesian coordinate axis with the flow direction, without loss of generality, we assume that flow is along the x-axis so that the inlet and outlet planes have the representations $x = 0$ and $x = L$ respectively. The pressure difference across the inflow and outflow boundaries is set to 1 MPa.

In the next step, an Eulerian velocity field $\mathbf{u}(\mathbf{x})$ consistent with the Darcy fluxes is reconstructed by using the technique outlined in Painter et al. [65] and Makedonska et al. [64]. The trajectory of a particle starting at location $\mathbf{a}$ on the inlet plane with a mass $m(\mathbf{a})$ through $\mathbf{u}(\mathbf{x})$ is given by the kinematic relationship:

$$\frac{d\mathbf{x}(t;\mathbf{a})}{dt} = \mathbf{v}(t;\mathbf{a}), \;\; \mathbf{x}(0;\mathbf{a}) = \mathbf{a}, \tag{6}$$

where $\mathbf{v}(t;\mathbf{a}) := \mathbf{u}(\mathbf{x}(t;\mathbf{a}))$ is the Lagrangian description of the known velocity field. The initial distribution of particles on the inlet plane is assigned using flux-weighting, where the number of particles at a location is proportional to the inflow flux at that location [21]. We take every particle to have the same mass $m(\mathbf{a})$. We assume there is complete mixing at fracture intersections so that the probability to enter an outgoing fracture is weighted by the outgoing flux therein [69].

Thus, once Eq. 6 is integrated to obtain $\mathbf{x}(t;\mathbf{a})$, we can gather the first passage time of a particle to cross the outlet plane, $\tau(\mathbf{a})$. The distribution of $\tau(\mathbf{a})$ across the ensemble represents the probability of solute mass breaking through at a time t, given by the Lebesgue integral:

$$\Phi(t) = \frac{1}{M_0} \int_{\Omega_a} dm(\mathbf{a}) \delta[t - \tau(\mathbf{a})], \tag{7}$$

where Ω_a is the set of all particles, $\delta[\cdot]$ is the Dirac-delta measure, and M_0 is the total mass of the particles. The function $\Phi(\cdot)$ is referred to as the breakthrough curve, and it is the main quantity of interest in this article.

In addition to the time taken for a particle to pass through the system, we can track the sequence of fractures through which the particle travels as it moves through the fracture network. Since the dimensions and location of all fractures in the network are known, the value of $\mathbf{x}(t;\mathbf{a})$ for a particular time t permits us to determine the particular fracture within which the particle is instantaneously located. Starting by representing a DFN composed of n fractures as a set $F = \{f_i\}$ for $i = 1, \ldots, n$, the sequence of fractures traversed by the kth particle is the finite set $S^{(k)} = \{f_i, f_j, \ldots\}$, which we call the particle trace. Once constructed, the collection $S^{(1)}, S^{(2)}, \ldots$ across Ω_a allows us to determine the number of particles (and thus proportion of the fluid mass) that passed through a particular fracture. Intuitively, it is clear that this construction helps identify sequences of fractures that form the network backbone. This concept is developed and formalized to create a flow topology graph that is integral to further developments in this paper, and we describe it in greater detail in relation to the construction of the network backbone.

2.1 Flow topology graph representation of DFN

Thinking of DFN as a collection of intersecting fractures leads to a natural association with the mathematical construct of a graph. A graph is a structure used to depict a set of objects along with the relationship between them (taken in pairs). Specifically, a graph is an ordered tuple $G(V, E, \mathcal{F})$ consisting of a set of vertices $V = \{v_1, v_2, \ldots\}$, a set of edges $E = \{e_{ij} | \mathcal{F}(v_i, v_j) = +1\}$ for "related" vertices v_i, v_j, and a function $\mathcal{F} : V \times V \mapsto \{+1, -1\}$ that defines whether or not two vertices are "related." For brevity, however, the third argument is often suppressed, and a graph is denoted simply by $G(V, E)$. Within this context, for a given problem, one can construct different graphs by changing either the objects that make up the vertex set V or the edge set E, or the definition of when elements in V are said to be related, or both. Both the vertices and edges can also have weights assigned to them to further encode information about the vertices themselves or the connection between vertices.

One construction of a graph based on a DFN would be to use the set of fractures as the vertex set V and define two vertices to be related, i.e., possessing an edge, if the two fractures intersect [28, 31, 33]. Formally, for a DFN F, we can define a mapping that transforms F into a graph $G(V, E)$ composed of $|V| = n$ vertices, and $|E|$ edges as follows: For every fracture f_i in the network, there is a corresponding vertex in V, and if two fractures intersect, $f_i \cap f_j \neq \emptyset$, then there is an edge $e(i, j) \in E$ that represents their intersection. Additionally, representative source (s) and target (t) vertices are added to V to incorporate flow direction. If a fracture f_i intersects the inflow boundary, then an edge is added to E between the vertex corresponding to f_i and the source vertex s to represent their connectivity, likewise for t and the outflow boundary.

A *path* in a graph $G(V, E)$ is a finite sequence $\{v_i, v_j, \ldots v_n\}$ consisting of elements in the vertex set V. In this instance, the path is said to be from vertex v_i to vertex v_n. At this juncture, it becomes clear that each of the particle-traces $S^{(1)}, S^{(2)}, \ldots$ that represent the sequence of fractures traversed by a particle in the DFN is equivalent to a sequence of vertices corresponding to a path through the graph, from the source vertex to the target vertex. Thus, the graph representation has naturally inherited the concepts of Lagrangian transport, and serves as a surrogate to the DFN.

Figure 1 is an example of the mapping and how particle information can be included into the graph. Subfigure (a) shows a DFN composed of 45 fractures colored by the number of particles that pass through each fracture and (b) is the corresponding graph representation of the DFN shown in (a). Here, the vertex size is proportional to the number of particles that pass through the corresponding fracture. The source (inflow) is colored blue and the target (outflow) is red.

Note that multiple graph representations are possible for a DFN. The most general representation is a bipartite graph where fractures and intersections are two disjoint vertex sets of the graph [37] from which other representations previously discussed can be obtained via graph projections. As mentioned earlier, one could also consider the set of fracture intersections to be the vertex set of the graph [35, 41].

2.2 Backbones

The formal nature of this graph representation, and the inclusion of particle information onto the graph, provides a convenient framework to identify the backbone of a DFN.

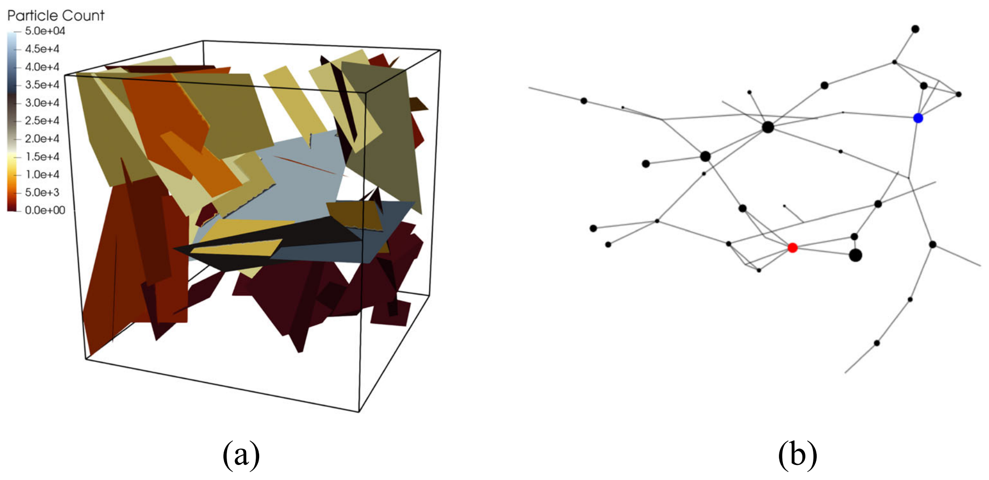


Fig. 1 **a** A DFN composed of 45 fractures colored by the number of particles that pass through each fracture. **b** A graph representation of the DFN shown in (**a**). Vertex size is proportional to the number of particles that pass through the corresponding fracture. The source (inflow) is colored blue and the target (outflow) is red

Given a graph G representing a DFN, denote the set of all vertices in the network as G_N. Choosing a backbone is tantamount to defining a function:

$$\phi : G_N \to \{0, 1\}, \tag{8}$$

Thus, one determines $G_B \subset G_N = \phi^{-1}(\{1\})$ to be the set of vertices (fractures) in the backbone, while the list of excluded vertices (fractures) $G_N - G_B = \phi^{-1}(\{0\})$. Our goal is to construct the mapping that identifies backbone vertices without performing high-fidelity simulations. However, we first construct a backbone that one could obtain based on results from the high-fidelity simulations.

To identify a backbone, we first represent the DFN with a flow topology graph (FTG) using the method provided by Aldrich et al. [34], which we briefly recount for completeness. Given a DFN with n fractures and k particle-traces, we create the vertex set $V = \{v_1, \ldots, v_n, v_s, v_t\}$ consisting of the fractures and source and target vertices as before. Now, corresponding to a given particle-trace, there is a path on the graph that starts at the source vertex and ends at the target vertex. In the particle-trace $S^{(k)}$, each time a particle transitions from vertex v_i to v_j, we create an edge directed from v_i to v_j or add 1 to the edge weight $w_{i,j}$ if it already exists. Once this process is followed for all $\{S^{(i)} : i = 1, 2, \ldots, k\}$, cycles are removed by eliminating the edge with lesser edge weight. Finally, we reset the edge weight to be the reciprocal, i.e., $1/w_{ab}$, so that edges with high proportion of particles flowing through (corresponding to backbones) have low weights. Thus, by computing the $i = 1, 2, \ldots, i_{max}$ shortest paths from source vertex (s) to target vertex (t), one can obtain the backbone. The value of i_{max} is determined by plotting the percent of total mass moving through the paths as a function of i. Thereafter, the derivative is computed to estimate how much flow is added as the number of paths in the graph is increased, and $i = i_{max}$ is chosen such that the derivative is close to 0.

Figure 2a shows the backbone defined for the network provided in Fig. 1, and demonstrates how fractures with few particles on them have been removed. The corresponding graph is shown in Fig. 2b and highlights how there are no dead-end fractures on the backbone and no isolated fractures. Moreover, there are some fractures where there are lower numbers of particles, but are on a path where the total number of particles is high. Thus, the backbone accounts for dispersion and flow channelization of the particles.

3 Machine learning for DFNs

3.1 Problem definition

Our goal is to construct a mapping (8) that identifies backbone vertices without performing high-fidelity simulations on the test networks once supplied with simulation data from the training networks.

To ensure connected backbones, we classify simple paths in the network instead of individual fractures. Recall that a path in a graph was defined earlier as a sequence of vertices so that the edges when taken in order connect the first and last vertex in the sequence. A simple path is a path with no repeating vertices (i.e., a sequence with distinct elements) and the set of all simple paths from source to target is denoted as G_P. The concept of paths on graphs is inherited from that of particle-trace on the corresponding DFN as was described in Section 2.1.

Denoting the set of paths used to construct the backbone as G_{BP}, assign the label "1" to paths in G_{BP}, and "0" to those in $G_P - G_{BP}$.

To find the mapping analogous to Eq. 8 for paths, we proceed in steps. First, transform each path $p \in G_P$ into a

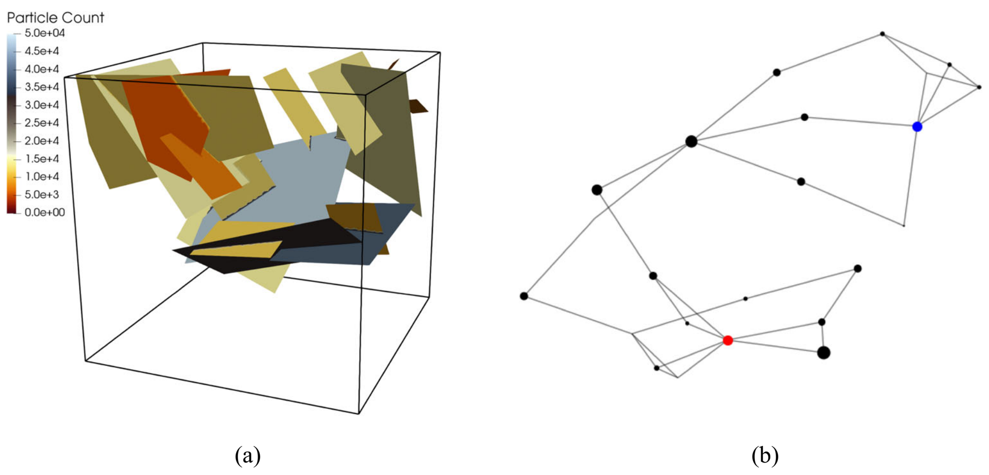


Fig. 2 **a** DFN backbone identified for the network provided in Fig. 1. **b** Corresponding graph

feature vector $x \in \mathbb{R}^{N_P}$ (with length N_P) by the action of a function $\phi(\boldsymbol{.})$.

$$\phi_P : G_P \to \mathbb{R}^{N_P}, \tag{9}$$

The features represent physical and graph-based properties of the paths, and the ones used in this paper are discussed in Section 3.2.

Next, construct a model function $M(\boldsymbol{.})$, which assigns probability of backbone membership corresponding to a feature vector of a path. We explored random forest and logistic regression models, which are detailed in Section 3.4.

$$M_P : \mathbb{R}^{N_P} \to \mathbb{R}, \tag{10}$$

$$\forall\ x\ \in\ \mathbb{R}^{N_P},\ M_P(x) = \text{Prob}(\phi_P^{-1}(x) \in G_{BP}) \tag{11}$$

Finally, define a threshold $T > 0$ and assign backbone membership to paths with probability higher than T through a function $f_P(\boldsymbol{.})$ as follows:

$$f_P : G_P \to \{0, 1\}, \tag{12}$$

$$\forall\ p\ \in\ G_P,\ f_P(p) = \begin{cases} 1 & M_P(\phi_P(p)) > T \\ 0 & \text{else} \end{cases} \tag{13}$$

Therefore, one has $G_{BP} = f_P^{-1}(\{1\})$. Knowing all the paths (sequences of vertices/fractures) that comprise the set G_{BP} translates directly into information about all the fractures that comprise the set G_B, thus completing the description of the backbone.

The backbone set is then the union of all vertices in the selected paths, which is guaranteed to be connected because every vertex will lie on at least one path from source to target and all of the vertices on that path are assigned to the backbone.

Computing all possible simple paths through the graph representation of a DFN is combinatorially impractical because the number of path combinations grows exponentially. However, computing the first k unweighted shortest paths is feasible, and the complexity scales as $O(kn^3)$ [70] for a graph with n vertices. We confirmed that 97% of vertices in the backbone are contained in the first 1000 shortest paths through the graphs. To include all backbone vertices requires roughly double the number of shortest paths. Given that the increase in paths far outweighed the increase in backbone vertices, we elected to use only the 1000 shortest paths.

3.2 Feature selection

For every path $p \in G_p$, we use five features based on individual vertex properties along that path to classify backbone membership: degree centrality, betweenness centrality, current flow betweenness centrality, and both the arithmetic and harmonic means of permeability. The values of degree centrality, betweenness centrality, and current flow betweenness centrality are the sum of these features computed at every vertex on the path; i.e., for a vertex feature $f(i)$, we calculate the path feature to be $\Sigma_{i=1}^{N} f(i)/N$ for a path with N vertices. Analogously, the cost of computing a feature is at most the sum of the costs of computing the corresponding feature for each vertex on the path. Next, we briefly describe each feature and why it was selected. We tested additional features as well, such as the attributes of quotient graphs corresponding to collapsed paths, but they did not improve the quality of classification.

Degree centrality: The *degree centrality* of a vertex is a normalized value representing the number of edges touching a vertex. For a fracture in a network, it is an indicator of the number of fractures that intersect it. High degree centrality vertices tend to be at the core of the network, while low degree centrality vertices are usually on the periphery or the low flow branches. For a vertex i, its degree centrality is given by:

$$D(i) = \frac{1}{n-1}\sum_{j=1}^{n} A_{ij}, \tag{14}$$

where A_{ij} is the ijth element of the adjacency matrix A of the graph and n is the number of vertices in the graph. The calculation of degree centrality has complexity $O(n^2)$ since it can be realized as a matrix-vector product of the adjacency matrix with a vector of ones.

Betweenness centrality: Betweenness centrality is a global topological measure quantifying the extent to which a vertex controls communication in a network by estimating how frequently paths through the network include a given vertex. Define a geodesic path as a path (sequence of vertices connected by edges) between vertices u and v with the fewest possible edges, and denote the number of geodesics as σ_{uv}. Denote $\sigma_{uv}(i)$ as the number of geodesic paths through u and v that pass through vertex i. *Betweenness centrality* is a normalized metric that indicates the fraction of geodesic paths in G that pass through vertex i:

$$B(i) = \frac{1}{(n-1)(n-2)} \sum_{u,v=1,u\neq i\neq v}^{n} \frac{\sigma_{uv}(i)}{\sigma_{uv}} \tag{15}$$

Vertices with high betweenness centrality represent hubs that many paths pass through, and represent either highways or bottlenecks for the flow. For a graph with n vertices and m edges, the computational complexity of calculating the betweenness centrality scales as $O(mn + n^2 \log(n))$ [71].

Current flow betweenness centrality: Source to target current flow is a centrality measure based on an electric circuit analogy, where every edge has unit resistance, a unit current is injected at a "source" node, and the flow is measured from source to target. Define the *Graph Laplacian* as $L = D - A$, where D is a diagonal matrix specifying the degree of each node. Denoting the pseudo-inverse of L as L^+, current flow centrality of a vertex is the total amount of current that flows through the vertex. The potential of the ith vertex is $L^+_{is} - L^+_{it}$, so the magnitude of the current through the edge that connects vertices i, j is $|(L^+_{is} - L^+_{it}) - (L^+_{js} - L^+_{jt})|$. To get the total current, we use the ith row of the adjacency matrix A to sum over all the edges that connect to vertex i, and thus the current flow centrality:

$$C(i) = \sum_{j=1}^{n} A_{ij} |(L^+_{is} - L^+_{it}) - (L^+_{js} - L^+_{jt})| \tag{16}$$

This centrality measure can be modified by using an adjacency matrix that is weighted by the resistance of each edge. The computational complexity scales as $O\left(mn\sqrt{k} + mn\log(n)\right)$ [72], where m, n, k denote the number of edges, number of vertices, and the condition number of the Laplacian matrix of the graph respectively.

Permeability: Another fracture feature is permeability, which is related to the idea of "conductivity," i.e., how much fluid can flow through the material. In the language of resistor networks, $R_{ij} = \frac{1}{k_{ij}}$, where R_{ij} is the resistance of the ijth edge expressed as the reciprocal of the edge permeability. In the representation that maps fractures to graph vertices, the permeability of an edge k_{ij} is given by the harmonic average of the permeabilities (k_i, k_j) of the two fractures along the edge. The harmonic average is used since that is the equivalent permeability of an edge that will transport the same flux under the same pressure gradient. Note that, in this case, along with the mean permeability along the path (defined as the mean of the edge permeabilities along the path), we use the harmonic average of the permeability along the path as a feature too.

3.3 Classification metrics

The majority of simple paths and vertices in the backbone makes up a small subset (roughly 4%) of the vertices of the graph, i.e., the cardinality of G_B is much smaller than G_N. While small backbones are advantageous for the high-fidelity simulations, it makes for a highly imbalanced classification problem for the machine learning algorithm. This class imbalance means models will be biased toward predicting "0" (membership of the set $G_N - G_B$) and still retain "high" accuracy without capturing the paths of interest. To address this issue, we train the models for precision and recall, which are more robust to imbalanced datasets than the traditional sensitivity and specificity.

Define the paths (vertices) correctly predicted to belong in G_{BP} (G_B) as "true positives" (TP), and paths (vertices) predicted to be in G_{BP} (G_B) but are actually in $G_P - G_{BP}$ $(G_N - G_B)$ as "false positives" (FP). The notion of "false negatives" (FN) follows analogously. Precision is given by:

$$p = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{17}$$

and measures how many false positives were predicted compared with true positives. Recall is given by:

$$r = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{18}$$

and represents the total percentage of correctly predicted backbone nodes.

The "F1-score" is a composite measure of both precision and recall, being the harmonic average of the two. The sensitivity of the harmonic average to low values ensures that high scores can result only from classifiers with high values of both precision and recall.

The classification problem is for paths; thus, these measures for paths are immediately relevant, but our eventual goal is the aggregate of the selected paths, i.e., a set of fractures, hence we measure the precision and recall in terms of the fractures too, with the knowledge that the set of fractures, realized as a union of paths, preserves the network connectivity.

To validate the physical utility of the backbones, we also compute the breakthrough curve for the full network, the true backbone, and the predicted backbone. We expect that the tail of the breakthrough curve for the predicted backbone will lie between that of the true backbone and the full network, since adding more nodes will capture more of the particle flow than the backbone network.

3.4 Classification algorithms, data splitting, and cross-validation

We used random forest and logistic regression models, which are both very well-known algorithms for classification, for the model function $M(\cdot)$ in Eq. 13. Moreover, the random forest algorithm is based on decision trees, while logistic regression is based on gradient descent; thus, these two methods are representative of two different families of algorithms. Both algorithms are available in the `scikit-learn` ensemble package [73].

We split our data in a standard way, where 80% is used for training and 20% used for testing. Specifically, out of a total of 100 networks, we randomly select 80 for training and the remaining 20 are set aside for testing. Then, for each of the 1000 simple paths for a network, 5 features and a label are generated. The training set is further split by 10-fold cross-validation, using the subroutines available in `scikit-learn`, to train the hyperparameters of the model for the best "F1-score."

3.4.1 Logistic regression

Logistic regression is a generalized linear model used for binary classification. If $x_i \in R^{N_p}$ is the feature vector for path i, and $Y_i \in \{0, 1\}$ is the binary variable representing backbone membership, then the probability of backbone membership is expressed as:

$$P(Y_i = 1|x_i) = \frac{1}{1 + \exp(\beta_0 + \beta^T x_i)} \qquad (19)$$

where β is a vector of regression coefficients. β and β_0 are typically found by maximum likelihood estimation given the training data. Logistic regression is implemented by using Maximum Likelihood Estimation (MLE) which is performed using a gradient descent algorithm. Maximizing the likelihood function determines the parameters that are most likely to produce the observed data. The coefficient β_k can be interpreted as the influence of the kth feature on the log-odds of backbone membership. For example, if $\beta_k = 5$ then if the kth feature is increased by 1, the log-odds of backbone membership are increased 5-fold. Typically, a regularization parameter C is also added to penalize misclassification, with smaller C corresponding to a stronger regularization [74]. In cross-validation using `scikit-learn`, we found $C = 1.291$.

3.4.2 Random forest

Random forests work by creating an ensemble of decision trees, which then vote on new points to predict backbone membership. Each tree in the ensemble is trained with a subset of data points sampled with replacement from the training data. Then, at each split in the tree, a random subset of the features is used to determine the split [75]. The total number of features in the data is denoted by N_p (see Section 3), and in this case, $N_p = 5$. The parameters of interest are the number of trees in the ensemble, `n_estimators`, and the number of features to use at each split in each tree, `max_features`. We performed cross-validation on a parameter range of `n_estimators` $\in$ [10, 50, 100, 200, 400, 500] and `max_features` $\in \{N_p, \sqrt{N_p}, \log_2(N_p)\}$. The best parameters were found to be `n_estimators` = 100 and `max_features` = $\log_2 N_p$ features used in each split.

4 Results

4.1 Sample networks

In order to test the performance of the algorithm, we chose to create semi-generic DFN with attributes based on data from actual geological sites. While it is known that macroscale network properties influence the flow field strongly [32, 76], when the range of the length scale is wide, it is challenging to link features of the fracture network with observed flow and transport properties. Thus, fracture networks that have fractures spanning a range of length scales offer a good test case for determination of flow channeling and backbone subnetworks. Hence, 100 independent, identically distributed realizations of a fracture network were generated based on data from Bonnet et al. [1] and Ouillon et al. [77]. The data is based on shear mode II faults at geological locations on the western Arabian plate where the bedrock is well exposed and composed of sedimentary and volcanic rocks. Specifically, the study focused on the homogeneous and uniformly thick Palezoic platform and the Cambrian-Ordovician sandstones [77].

It is assumed that the overall domain is cubic, and each edge has a length of 100 m. The fractures are circular with uniformly random orientations, while fracture centers are assumed to be sampled uniformly throughout the domain. The fracture radii are sampled from a truncated power-law distribution whose probability density function has an exponent of 2.1 and upper and lower cutoffs of 2 and 30 m respectively.

Fracture apertures (measured in meters) are constant for each fracture, but correlated with their radii through the relationship $b = 5 \cdot 10^{-5} \cdot \sqrt{r}$ [25]. This leads to variability in hydraulic properties within the network. We require that at least one set of fractures connects the inflow and outflow boundaries in every realization. Isolated clusters that do not contribute to the flow are removed. The networks have a mean P_{32} value (fracture surface area over total volume) of 0.13 (m^{-1}) and standard deviation 0.03. There are 218 fractures on average, with a standard deviation of 60, in each network, and there are multiple paths connecting the inflow and outflow boundaries.

We identified backbones in the network using the FTG methodology previously described. For these networks, we found that we needed the first 40 paths in the FTG. This backbone accurately captures first arrival and peak breakthrough times. Figure 3 shows the breakthrough curve from a single network (solid black line), the backbone (dashed line), and 95% confidence interval of breakthrough curves from the ensemble of networks. From the 100 realizations, 80 are selected for training, and 20 are set aside for testing.

4.2 Classification performance

When we use random forest (RF) or logistic regression (LR) to solve the classification problem described in Section 3, the immediate result is an assignment of probability of class membership for each item being classified. Although the hyperparameters of the RF and LR models were tuned by cross-validation to obtain the best "F1-score," it is more informative to list the precision and recall scores separately and construct the PR curve (precision-recall

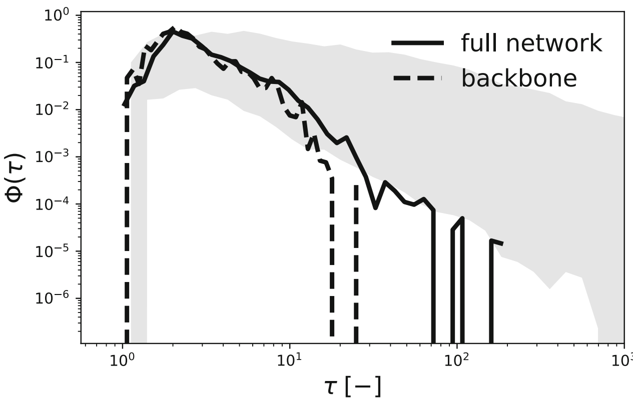


Fig. 3 Representative network from the ensemble showcasing comparison of first arrival and peak breakthrough times for full network and backbone derived from the FTG, with the 2.5 to 97.5 percentiles of times for full networks shown shaded. The abscissa denotes time nondimensionalized by first passage time

curve). We compute PR curves based on the realizations designated for testing alone. To be clear, for each of the 100 realizations, we consider 1000 labeled shortest paths (mentioned in Section 3.1). The testing data comprises 20 realizations chosen out of the 100, and PR curves are computed by comparing the predicted labels with the true labels for each path. The threshold probability T that finally decides membership in Eq. 13 is the user-defined parameter that allows us to control precision and recall, and thus indirectly, the extent of system reduction. By systematically sampling values of the probability $T \in [0, 1]$, precision and recall are computed for each value, thus setting up an approximate one-to-one correspondence between the threshold probability T and the resultant precision p and recall r. Thus, an optimal T can be chosen for a desired value of p or r.

Since the classification problem is for paths, i.e., the identification of the set G_{BP}, these measures for paths are immediately relevant, but our eventual goal is the aggregate of the selected paths, i.e., a set of fractures; hence, we measure the precision and recall in terms of the fractures too, with the knowledge that the set of fractures G_B, realized as a union of paths, preserves the network connectivity. The two PR curves obtained for these cases are shown in Fig. 4. We emphasize that no fracture classification problem is being solved—we are interpreting the results of the path classification problem in terms of both paths G_{BP} and fractures G_B. For the perfect classifier, i.e., one with a precision and recall of hundred percent, the curve would be horizontal at precision $= 1$ across every recall value, but for real classifiers, curves close to the top right corner are desirable. In general, the two algorithms perform very similarly, with logistic regression having higher precision at almost all recall values.

The resultant system reduction, measured as the number of fractures in the backbone network as a percentage of the full network, is shown in Fig. 5. The figure shows that for a given recall, both methods yield backbones of approximately similar size. For flow backbones, we are most interested in high recall, i.e., minimizing the loss of important fractures due to false negative classification, and both algorithms show that small backbones of approximately 30% size can be obtained even at very high recall. It is worth noting that a random sampling of paths out of the candidate set will sometimes generate high recall values, but almost always have low precision, so it is notable that both methods outperform a simple random sample. The figure also shows the size of the backbone that would be obtained by a hypothetical classifier with perfect precision and recall. Both RF and LR yield backbones of comparable size for values of precision and recall around 70%.

The size of the backbone directly informs us as to the computational efficiency of high-fidelity simulations on it

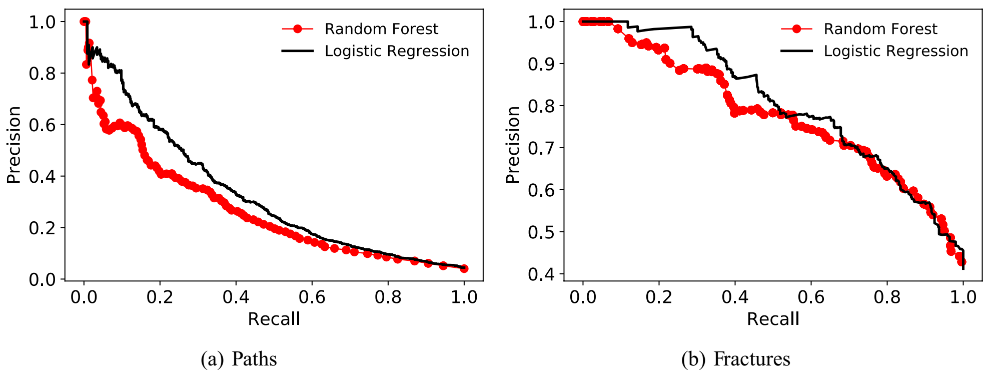


Fig. 4 The precision-recall (PR) curves have been constructed for the original path classification problem in (**a**). The path classification problem results in identification of the set G_{BP} that comprises backbone paths. The aggregate of the selected paths in G_{BP} yields a set of selected fractures G_B. Measures of precision-recall for the derived set of fractures are shown in (**b**)

in comparison with that on the full network. Based on prior observations [41], one can estimate that a subnetwork that is a fraction of the full network will require roughly the same fraction of computational effort. Thus, the computational efficiency gained will depend on the size of the subnetwork, so that small subnetworks are accompanied by large reductions in computational time, yielding as much as 80–90% computational savings for subnetworks that are less than 10% of the full network.

The use of random forest algorithm affords us, as a by-product, values of feature importance. These are shown in Fig. 6, and provide confirmation that while the permeability associated with paths is the most important feature, there are no superfluous features. In this context, it shows a distinction between fracture classification and path classification. In fracture classification [58, 59], the global topological features were overwhelmingly dominant, to the extent that the influence of fracture permeability as a feature was negligible. However, in path classification, we see that permeability does play an important role, albeit as a global property associated with a path, as opposed to an individual fracture. This insight into flow channeling in fracture networks has emerged due to the fundamental change in the way we posed our problem—as path classification rather than fracture classification.

We now compare the performance of the algorithms in more detail by examining the results for chosen recall values of 50%, 80%, and 90% respectively. In Table 1, we compare the system reduction through the number of fractures and the number of degrees of freedom in the mesh needed to simulate flow and transport on the obtained backbone network. We note that for each recall value, LR has greater network size, and the explanation for these numbers lies in the threshold probability T that corresponds to the recall. Thus for RF, values of $T = 0.48, 0.19, 0.11$ yield a recall of $50\%, 80\%, 90\%$ respectively, but the corresponding numbers for LR are $T = 0.30, 0.09, 0.05$. Hence, more paths (consequently fractures) are selected by LR to ensure the algorithm performance results in the desired recall value, and this is directly reflected in Table 1.

The results thus far have focused on metrics that relate to the classification problem, but the end goal is to examine how well do the transport characteristics of the backbone correspond to those of the full network. We do this by comparing the breakthrough curves of the backbone subnetwork produced by the classifiers with that of the full

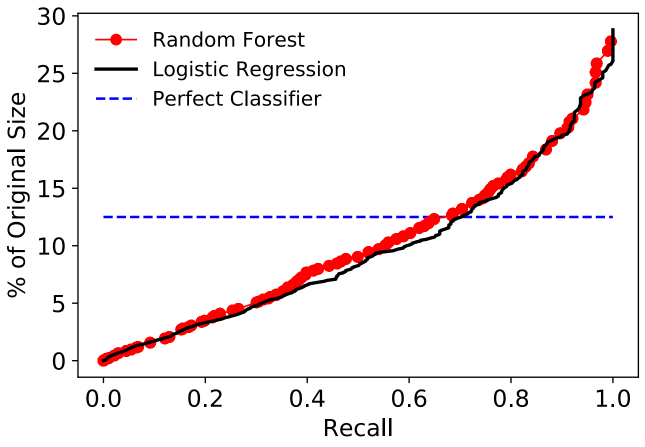


Fig. 5 Size reduction (the number of fractures in the backbone as a percentage of the full network) as a function of recall for both random forest and logistic regression. As recall increases, precision decreases; hence, the backbone includes a larger number of fractures. The perfect classifier is one that would produce neither false positives nor false negatives, and this corresponds to the true backbone of the networks in the test dataset

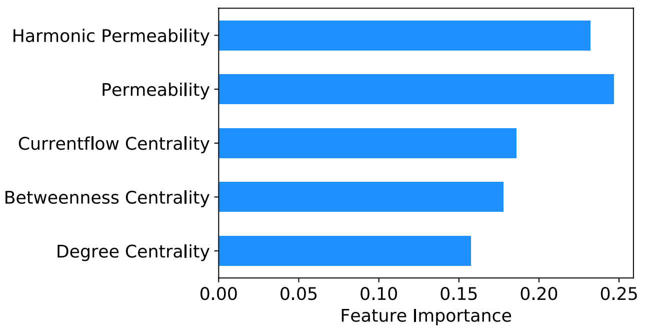


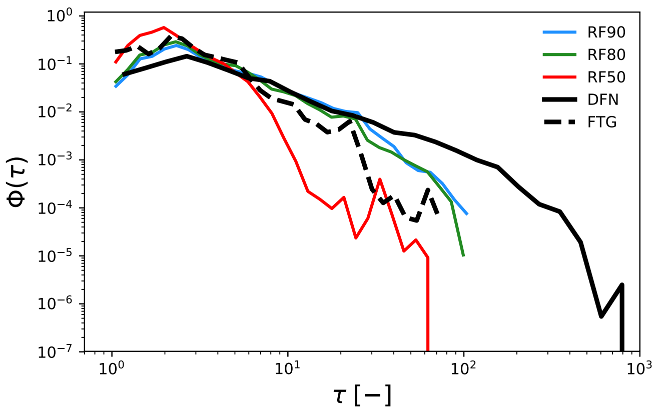
Fig. 6 An estimate of feature importance for the path classification problem obtained from the random forest classifier

Table 1 Reduction measures for random forests and logistic regression at 50%, 80%, and 90% recall thresholds. The precision-recall curves for random forest and logistic regression were almost indistinguishable from each other, and these numbers further emphasize the trend

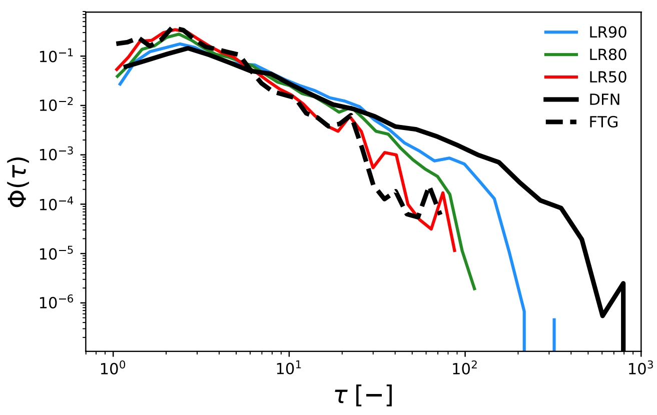
Model	Recall	Network size (fracture) (%)	Network size (mesh) (%)
RF	50	8	16
LOG	50	12	22
RF	80	17	30
LOG	80	20	34
RF	90	20	34
LOG	90	23	38

network as well as the true backbone produced by a perfect classifier. We expect adding more fractures will capture more of the particle flow than the true backbone. Our expectations are borne out by the trend in the breakthrough curves as recall increases for both random forest and logistic regression in Fig. 7a and b. As recall increases, precision drops, so the number of fractures in the backbone increases and it matches the full breakthrough curve better. When comparing the breakthrough curves of random forests and logistic regression for 50% and 90% recall in Fig. 7c, again, we observe that the breakthrough curve of logistic regression matches that of the full network better, and this is due to the same reason mentioned earlier—that logistic regression has a lower threshold probability for a given recall, hence a larger backbone with greater fidelity.

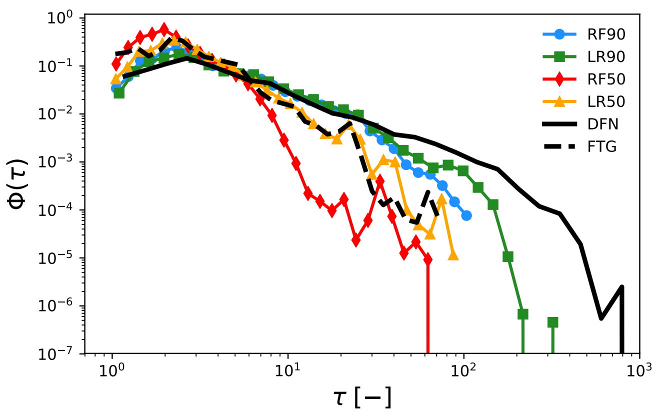
For a quantitative estimate to support our visual observations, the mean Kullback-Leibler (KL) divergence is computed for the full ensemble of breakthrough curves of the tested networks, measuring how close two probability density functions are. The trends in Table 2 agree with what has been seen already, namely, that as recall increases, the corresponding values of the KL divergence with respect to the full network decrease, indicating closer agreement.



(a) Random Forest



(b) Logistic Regression



(c) Both

Fig. 7 An instance showing the breakthrough curves obtained from the classifiers for 50%, 80%, and 90% recall for both random forest (RF) and logistic regression (LR). The numbers in the legend indicate the percentage value of recall. The breakthrough curves obtained from the full network (legend DFN) and the true backbone which is the flow topology graph (FTG) are also shown

Table 2 Table showing KL divergence measures of breakthrough curve of backbone with respect to breakthrough curve of full network and that of the true backbone

Model	Recall	KL Div w.r.t full network	KL Div w.r.t true backbone
RF	50	0.25	0.12
LOG	50	0.17	0.09
RF	80	0.07	0.10
LOG	80	0.05	0.15
RF	90	0.05	0.15
LOG	90	0.04	0.21

However, the KL divergence with respect to the true backbone does not show monotonic behavior because as recall increases, more fractures are added to the backbone, taking it away from the true backbone, but closer to the full network.

5 Conclusion

The end goal of designing the machine learning for backbone identification in discrete fracture networks is system reduction for computational efficiency as well as physical insight into the phenomenon of flow channeling. There are various aspects to this problem, but the most critical is the need to identify backbones that connect inflow to outflow boundaries. In contrast to previous efforts in this direction, such as [58, 59] which cannot guarantee connected backbones because their fundamental unit of selection is a fracture, we presented a novel perspective that views a network as a union of connected paths comprising a sequence of fractures. Thus, the fundamental unit of selection in the machine learning algorithm is a sequence of fractures, rather than individual ones, and the classification process is based on attributes that characterize the sequence. Unlike fracture classification, where the global topological features were overwhelmingly dominant, to the extent that the influence of fracture permeability as a feature was negligible, in path classification, we saw that permeability does play an important role, albeit as a global property associated with a path, as opposed to an individual fracture. This insight into flow channeling in fracture networks has emerged due to the fundamental change in the way we posed our problem—as path classification rather than fracture classification. Moreover, this choice to classify based on paths rather than individual fractures imposes the aforementioned physical constraint that backbones need to be connected. In this light, the proposed method is the first physics-informed machine learning algorithm for backbone identification within discrete fracture networks simulations.

One disadvantage of this method that carries over from previous efforts is the computational expense of the high-fidelity simulations to generate the training data. Future work could examine the possibility of using low-fidelity data from reduced-order models to further increase computational efficiency. System reduction of DFNs using machine learning thus paired with three-dimensional DFN modeling is an efficient work flow, and lends itself to robust uncertainty quantification and characterization of subsurface flow and transport.

Funding information Los Alamos National Laboratory (US) LDRD #20170103DR: All authors
Los Alamos National Laboratory (US) LDRD #20180621ECR: JDH
Department of Energy (DOE) Basic Energy Sciences program DE-AC52-06NA25396: SS, JDH, HV
Los Alamos National Laboratory (US) LDRD #20170508DR: SS
NSF Grant No. DGE-1745038: EC

References

1. Bonnet, E., Bour, O., Odling, N.E., Davy, P., Main, I., Cowie, P., Berkowitz, B.: Scaling of fracture systems in geological media. Rev. Geophys. **39**(3), 347 (2001)
2. Hyman, J., Jiménez-Martínez, J., Viswanathan, H., Carey, J., Porter, M., Rougier, E., Karra, S., Kang, Q., Frash, L., Chen, L., Lei, Z., O'Malley, D., Makedonska, N.: Understanding hydraulic fracturing: a multi-scale problem. Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences **374**(2078), 20150426 (2016)
3. Karra, S., Makedonska, N., Viswanathan, H.S., Painter, S.L., Hyman, J.D.: Effect of advective flow in fractures and matrix diffusion on natural gas production, Water Resour. Res. **51**(10), 8646 (2015)
4. Middleton, R., Carey, J., Currier, R., Hyman, J., Kang, Q., Karra, S., Jiménez-Martínez, J., Porter, M., Viswanathan, H.: Shale gas and non-aqueous fracturing fluids: opportunities and challenges for supercritical CO^2. Appl. Energy **147**, 500 (2015)
5. National Research Council: Rock fractures and fluid flow: contemporary understanding and applications. National Academy Press (1996)
6. Neuman, S.: Trends, prospects and challenges in quantifying flow and transport through fractured rocks. Hydrogeol. J. **13**(1), 124 (2005)
7. VanderKwaak, J., Sudicky, E.: Dissolution of non-aqueous-phase liquids and aqueous-phase contaminant transport in discretely-fractured porous media. J. Contam. Hydrol. **23**(1-2), 45 (1996)
8. Jenkins, C., Chadwick, A., Hovorka, S.D.: The state of the art in monitoring and verification—ten years on. Int. J. Greenh. Gas Control **40**, 312 (2015)
9. Kueper, B.H., McWhorter, D.B.: The behavior of dense, nonaqueous phase liquids in fractured clay and rock. Ground Water **29**(5), 716 (1991)
10. Ng, L.W.T., Willcox, K.E.: Multifidelity approaches for optimization under uncertainty. Int. J. Numer. Methods Eng. **100**, 746 (2014). https://doi.org/10.1002/nme.4761
11. Giles, M.B.: Multilevel Monte Carlo path simulation. Oper. Res. **56**, 607 (2008). https://doi.org/10.1287/opre.1070.0496
12. Berrone, S., Canuto, C., Pieraccini, S., Scialò, S.: Uncertainty quantification in discrete fracture network models:

Stochastic geometry . Water Resour. Res. **54**, 1338 (2018). https://doi.org/10.1002/2017wr021163
13. O'Malley, D., Karra, S., Hyman, J.D., Viswanathan, H.S., Srinivasan, G.: Efficient Monte Carlo with graph-based subsurface flow and transport models. Water Resour. Res. **54**, 3758 (2018). https://doi.org/10.1029/2017wr022073
14. Jackson, C.P., Hoch, A.R., Todman, S.: Self-consistency of a heterogeneous continuum porous medium representation of a fractured medium. Water Resour. Res. **36**, 189 (2000). https://doi.org/10.1029/1999wr900249
15. Painter, S., Cvetkovic, V.: Upscaling discrete fracture network simulations: an alternative to continuum transport models. Water Resources Research 41(2) (2005)
16. Karimi-Fard, M., Gong, B., Durlofsky, L.J.: Generation of coarse-scale continuum flow models from detailed fracture characterizations. Water Resources Research 42. https://doi.org/10.1029/2006wr005015 (2006)
17. Botros, F.E., Hassan, A.E., Reeves, D.M., Pohll, G.: On mapping fracture networks onto continuum. Water Resources Research 44. https://doi.org/10.1029/2007wr006092 (2008)
18. Tsang, C.F., Neretnieks, I.: Flow channeling in heterogeneous fractured rocks. Rev. Geophys. **36**(2), 275 (1998)
19. Abelin, H., Birgersson, L., Moreno, L., Widén, H., Ågren, T., Neretnieks, I.: A large-scale flow and tracer experiment in granite: 2. Results and interpretation. Water Resour. Res. **27**(12), 3119 (1991)
20. Abelin, H., Neretnieks, I., Tunbrant, S., Moreno, L.: Final report of the migration in a single fracture: experimental results and evaluation. Tech. Rep SKB-SP-TR–85-03 (1985)
21. Hyman, J.D., Painter, S.L., Viswanathan, H., Makedonska, N., Karra, S.: Influence of injection mode on transport properties in kilometer-scale three-dimensional discrete fracture networks. Water Resour. Res. **51**(9), 7289 (2015)
22. Frampton, A., Cvetkovic, V.: Numerical and analytical modeling of advective travel times in realistic three-dimensional fracture networks. Water Resources Research 47(2) (2011)
23. de Dreuzy, J.R., Davy, P., Bour, O.: Hydraulic properties of two-dimensional random fracture networks following a power law length distribution 2. Permeability of networks based on lognormal distribution of apertures. Water Resour. Res. **37**(8), 2079 (2001)
24. de Dreuzy, J.R., Méheust, Y., Pichot, G.: Influence of fracture scale heterogeneity on the flow properties of three-dimensional discrete fracture networks. Journal of Geophysical Research-Solid Earth 117 (B11) (2012)
25. Hyman, J., Aldrich, G., Viswanathan, H., Makedonska, N., Karra, S.: Fracture size and transmissivity correlations: implications for transport simulations in sparse three-dimensional discrete fracture networks following a truncated power law distribution of fracture size. Water Resour. Res. **52**(8), 6472 (2016)
26. Hyman, J.D., Hagberg, A., Srinivasan, G., Mohd-Yusof, J., Viswanathan, H.: Predictions of first passage times in sparse discrete fracture networks using graph-based reductions. Phys. Rev. E **96**(1), 013304 (2017)
27. Ghaffari, H., Nasseri, M., Young, R.: Fluid flow complexity in fracture networks: analysis with graph theory and lbm. arXiv:1107.4918 (2011)
28. Andresen, C.A., Hansen, A., Le Goc, R., Davy, P., Hope, S.M.: Topology of fracture networks. Frontiers in physics 1 art (2013)
29. Santiago, E., Velasco-Hernández, J.X., Romero-salcedo, M.: A methodology for the characterization of flow conductivity through the identification of communities in samples of fractured rocks, Expert Syst. Appl. **41**(3), 811 (2014). https://doi.org/10.1016/j.eswa.2013.08.011
30. Sævik, P.N., Nixon, C.W.: Inclusion of topological measurements into analytic estimates of effective permeability in fractured media. Water Resour. Res. **53**(11), 9424 (2017). https://doi.org/10.1002/2017WR020943
31. Hope, S.M., Davy, P., Maillot, J., Le Goc, R., Hansen, A.: Topological impact of constrained fracture growth. Front. Phys. **3**, 75 (2015)
32. Hyman, J.D., Jiménez-Martínez, J.: Dispersion and mixing in three-dimensional fracture networks: nonlinear interplay between structural and hydraulic heterogeneity. Water Resources Research. https://doi.org/10.1029/2018WR022585 (2018)
33. Huseby, O., Thovert, J.F., Adler, P.M.: Geometry and topology of fracture systems. J. Phys. A Math. Gen. **30**, 1415 (1997). https://doi.org/10.1088/0305-4470/30/5/012
34. Aldrich, G., Hyman, J.D., Karra, S., Gable, C.W., Makedonska, N., Viswanathan, H., Woodring, J., Hamann, B.: Analysis and visualization of discrete fracture networks using a flow topology graph. IEEE Trans. Vis. Comput. Graph. **23**(8), 1896 (2017). https://doi.org/10.1109/tvcg.2016.2582174
35. Karra, S., O'Malley, D., Hyman, J., Viswanathan, H., Srinivasan, G.: Modeling flow and transport in fracture networks using graphs. Phys. Rev. E **97**(3), 033304 (2018). https://doi.org/10.1103/PhysRevE.97.033304
36. Viswanathan, H.S., Hyman, J.D., Karra, S., O'Malley, D., Srinivasan, S., Hagberg, A., Srinivasan, G.: Advancing graph-based algorithms for predicting flow and transport in fractured rock. Water Resources Research. https://doi.org/10.1029/2017WR022368 (2018)
37. Hyman, J.D., Hagberg, A., Osthus, D., Srinivasan, S., Srinivasan, G., Viswanathan, H.S.: Identifying backbones in three dimensional discrete fracture networks: a graph-based multi-scale approach. Multiscale Modeling Sim. **16**(4), 5477 (2018)
38. Dershowitz, W., Fidelibus, C.: Derivation of equivalent pipe network analogues for three-dimensional discrete fracture net works by the boundary element method. Water Resour. Res. **35**(9), 2685 (1999)
39. Cacas, M.C., Ledoux, E., Marsily, G.D., Tillie, B., Barbreau, A., Durand, E., Feuga, B., Peaudecerf, P.: Modeling fracture flow with a stochastic discrete fracture network: calibration and validation: 1. the flow model. Water Resour. Res. **26**(3), 479 (1990)
40. Srinivasan, G., Hyman, J.D., Osthus, D., Moore, B., O'Malley, D., Karra, S., Rougier, E., Hagberg, A., Hunter, A., Viswanathan, H.: Quantifying topological uncertainty in fractured systems using graph theory and machine learning. Nature Scientific Reports 8 (11665) (2018)
41. Srinivasan, S., Hyman, J., Karra, S., O'Malley, D., Viswanathan, H., Srinivasan, G.: Robust system size reduction of discrete fracture networks: a multi-fidelity method that preserves transport characteristics. Computational Geosciences. https://doi.org/10.1007/s10596-018-9770-4 (2018)
42. Bergen, K.J., Johnson, P.A., de Hoop, M.V., Beroza, G.C.: Machine learning for data-driven discovery in solid earth geoscience. Science **363**, 6433 (2019). https://doi.org/10.1126/science.aau0323
43. Karpatne, A., Ebert-Uphoff, I., Ravela, S., Babaie, H.A., Kumar, V.: Machine learning for the geosciences: challenges and opportunities. IEEE Transactions on Knowledge and Data Engineering. p. 1. https://doi.org/10.1109/TKDE.2018.2861006 (2018)
44. Zhang, L., Zhang, L., Du, B.: Deep learning for remote sensing data: a technical tutorial on the state of the art. IEEE Geoscience and Remote Sensing Magazine **4**(2), 22 (2016)
45. Cracknell, M.J., Reading, A.M.: Geological mapping using remote sensing data: a comparison of five machine learning algorithms, their response to variations in the spatial distribution

of training data and the use of explicit spatial information. Comput. Geosci. **63**, 22 (2014)
46. National Oceanic and Atmospheric Administration: National Centers for Environmental Information. www.ncdc.noaa.gov (2018)
47. World Climate Research Program: Coupled model intercomparison Project. cmip-pcmdi.llnl.gov (2018)
48. Scavuzzo, J.M., Trucco, F., Espinosa, M., Tauro, C.B., Abril, M., Scavuzzo, C.M., Frery, A.C.: Modeling dengue vector population using remotely sensed data and machine learning. Acta Trop. **185**, 167 (2018)
49. Hengl, T., de Jesus, J.M., Heuvelink, G.B., Gonzalez, M.R., Kilibarda, M., Blagotić, A., Shangguan, W., Wright, M.N., Geng, X., Bauer-Marschallinger, B., et al.: Soilgrids250m: global gridded soil information based on machine learning. PLOS One **12**(2), E0169748 (2017)
50. Dev, S., Wen, B., Lee, Y.H., Winkler, S.: Ground-based image analysis: a tutorial on machine-learning techniques and applications. IEEE Geoscience and Remote Sensing Magazine **4**(2), 79 (2016)
51. Zhang, Z.: When doctors meet with alphago: potential application of machine learning to clinical medicine. Ann. Transl. Med. **4**, 125 (2016). https://doi.org/10.21037/atm.2016.03.25
52. Lary, D.J., Alavi, A.H., Gandomi, A.H., Walker, A.L.: Machine learning in geosciences and remote sensing. Geosci. Front. **7**(1), 3 (2016)
53. Griewank, A., Reich, S., Roulstone, I., Stuart, A.M.: Mathematical and algorithmic aspects of data assimilation in the geosciences. Oberwolfach Reports **13**(4), 2705 (2017)
54. Ravela, S.: A Symbiotic Framework for coupling Machine Learning and Geosciences in Prediction and Predictability. In: AGU Fall Meeting Abstracts (2017)
55. Dell'Aversana, P., Ciurlo, B., Colombo, S.: Integrated Geophysics and Machine Learning for Risk Mitigation in Exploration Geosciences. In: 80Th EAGE Conference and Exhibition 2018 (2018)
56. Karpatne, A., Atluri, G., Faghmous, J.H., Steinbach, M., Banerjee, A., Ganguly, A., Shekhar, S., Samatova, N., Kumar, V.: Theory-guided data science: a new paradigm for scientific discovery from data. IEEE Trans. Knowl. Data Eng. **29**(10), 2318 (2017). https://doi.org/10.1109/TKDE.2017.2720168
57. Pan, S., Duraisamy, K.: Data-driven discovery of closure models. SIAM J. Appl. Dyn. Syst. **17**(4), 2381 (2018)
58. Valera, M., Guo, Z., Kelly, P., Matz, S., Cantu, V.A., Percus, A.G., Hyman, J.D., Srinivasan, G., Viswanathan, H.S.: Machine learning for graph-based representations of three-dimensional discrete fracture networks. Comput. Geosci. **22**, 695 (2018). https://doi.org/10.1007/s10596-018-9720-1
59. Srinivasan, S., Karra, S., Hyman, J., Viswanathan, H., Srinivasan, G.: Model reduction for fractured porous media: a machine learning approach for identifying main flow pathways. Computational Geosciences (2018)
60. Hyman, J.D., Karra, S., Makedonska, N., Gable, C.W., Painter, S.L., Viswanathan, H.S.: dfnworks: A discrete fracture network framework for modeling subsurface flow and transport. Comput. Geosci. **84**, 10 (2015)
61. Hyman, J.D., Gable, C.W., Painter, S.L., Makedonska, N.: Conforming delaunay triangulation of stochastically generated three dimensional discrete fracture networks: a feature rejection algorithm for meshing strategy. SIAM J. Sci. Comput. **36**(4), A1871 (2014)
62. LaGriT: Los Alamos Grid Toolbox, (LaGriT). http://lagrit.lanl.gov (2013). Last Checked : May 20, 2019
63. Lichtner, P., Hammond, G., Lu, C., Karra, S., Bisht, G., Andre, B., Mills, R., Kumar, J.: PFLOTRAN user manual: a massively parallel reactive flow and transport model for describing surface and subsurface processes. Tech. rep. (Report No.: LA-UR-15-20403) Los Alamos National Laboratory. https://doi.org/10.2172/1168703 (2015)
64. Makedonska, N., Painter, S.L., Bui, Q.M., Gable, C.W., Karra, S.: Particle tracking approach for transport in three-dimensional discrete fracture networks. Comput. Geosci. **19**(5), 1123 (2015)
65. Painter, S.L., Gable, C.W., Kelkar, S.: Pathline tracing on fully unstructured control-volume grids. Comput. Geosci. **16**(4), 1125 (2012)
66. SKB: Long-term Safety for the Final Repository for Spent Nuclear Fuel at Forsmark. Tech. Rep, SKB TR-11-01. Swedish Nuclear Fuel and Waste Management Co., Stockholm (2011)
67. Alemanni, A., Battaglia, M., Bigi, S., Borisova, E., Campana, A., Loizzo, M., Lombardi, S.: A three dimensional representation of the fracture network of a co2 reservoir analogue (Latera Caldera, Central Italy). Energy Procedia **4**, 3582 (2011). https://doi.org/10.1016/j.egypro.2011.02.287
68. Boussinesq, J.: Mémoire sur l'in uence des frottements dans les mouvements réguliers des fluids. J. Math. Pures. Appl. **13**(377–424), 21 (1868)
69. Sherman, T., Hyman, J.D., Bolster, D., Makedonska, N., Srinivasan, G.: Characterizing the impact of particle behavior at fracture intersections in three-dimensional discrete fracture networks. Phys. Rev. E. **99**, 013110 (2019)
70. Yen, J.Y.: Finding the k shortest loopless paths in a network. Manag. Sci. **17**(11), 712 (1971)
71. Brandes, U.: A faster algorithm for betweenness centrality. J. Math. Sociol. **25**(2), 163 (2001)
72. Hagberg, A.A., Schult, D.A., Swart, P.: Exploring network structure, dynamics, and function using NetworkX. In: Proceedings of the 7th Python in Science Conferences (SciPy 2008), pp. 11–15 (2008)
73. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. J. Mach. Learn. Res. **12**, 2825 (2011)
74. Hosmer, D.W. Jr., Lemeshow, S., Sturdivant, R.X.: Applied logistic regression. vol. 398, Wiley (2013)
75. Cutler, A., Cutler, D.R., Stevens, J.R.: Random Forests. In: Ensemble machine learning, pp. 157–175. Springer (2012)
76. Grindrod, P., Impey, M.: Channeling and Fickian dispersion in fractal simulated porous media. Water Resour. Res. **29**(12), 4077 (1993)
77. Ouillon, G., Castaing, C., Sornette, D.: Hierarchical geometry of faulting. J. Geophys. Res. **101**(B3), 5477 (1996)

Affiliations

Shriram Srinivasan[1] · Eric Cawi[2] · Jeffrey Hyman[3] · Dave Osthus[4] · Aric Hagberg[5] · Hari Viswanathan[3] · Gowri Srinivasan[6]

[1] Center for Nonlinear Studies, Computational Earth Science Group (EES-16), Los Alamos National Laboratory, Los Alamos, NM 87545, USA

[2] Department of Electrical Engineering, Washington University in St. Louis, St. Louis, MO 63130, USA

[3] Computational Earth Science Group (EES-16), Earth and Environmental Sciences Division, Los Alamos National Laboratory, Los Alamos, NM 87545, USA

[4] Statistical Sciences Group (CCS-6), Computer, Computational and Statistical Sciences Division, Los Alamos National Laboratory, Los Alamos, NM 87545, USA

[5] Computer, Computational and Statistical Sciences Division, Los Alamos National Laboratory, Los Alamos, NM 87545, USA

[6] Verification and Analysis Group (XCP-8), X Computational Physics Division, Los Alamos National Laboratory, Los Alamos, NM 87545, USA