Layer-wise relevance propagation for backbone identification in discrete fracture networks

Journal of Computational Science 55 (2021) 101458

Contents lists available at ScienceDirect

Journal of Computational Science

journal homepage: www.elsevier.com/locate/jocs

Layer-wise relevance propagation for backbone identification in discrete fracture networks

Stefano Berrone [a,c,d], Francesco Della Santa [a,c,d,*], Antonio Mastropietro [a,c,e], Sandra Pieraccini [b,d], Francesco Vaccarino [a,c,f]

[a] Department of Mathematical Sciences, Politecnico di Torino, Turin, Italy
[b] Department of Mechanical and Aerospace Engineering, Politecnico di Torino, Turin, Italy
[c] SmartData@PoliTO center for Big Data and Machine Learning technologies, Politecnico di Torino, Turin, Italy
[d] Member of the INdAM-GNCS research group, Italy
[e] Addfor Industriale s.r.l., Turin, Italy
[f] ISI Foundation, Italy

ARTICLE INFO

MSC:
65D40
68T07
68T37
76-10
76-11

Keywords:
Layer-wise Relevance Propagation
Deep Learning
Neural Networks
Discrete Fracture Network
Feature selection

ABSTRACT

In the framework of flow simulations in Discrete Fracture Networks, we consider the problem of identifying possible backbones, namely preferential channels in the network. Backbones can indeed be fruitfully used to analyze clogging or leakage, relevant for example in waste storage problems, or to reduce the computational cost of simulations. With a suitably trained Neural Network at hand, we use the Layer-wise Relevance Propagation as a feature selection method to detect the expected relevance of each fracture in a Discrete Fracture Network and thus identifying the backbone.

1. Introduction

Discrete Fracture Networks (DFNs) [1–3] are popular models adopted for performing flow simulations in underground fractured media, in which each fracture of the network is represented by a 2-dimensional polygon into a 3-dimensional domain and characterized by its own geometrical and hydro-geological features (namely, position, size, orientation, fracture transmissivity, etc.). In this paper we propose a new strategy, based on flux-regression Neural Networks (trained on datasets generated via DFN flow simulations) and Layer-wise Relevance Propagation, to identify backbones of DFNs, namely, suitable sub-networks where the transport characteristics approximate the ones of the original network [4]. These sub-networks can be used in many applications and furnish relevant information for estimating probability of clogging problems or leakage phenomena, which are crucial issues for example for geological waste storage applications (e.g. geological storage of CO_2) and enhanced oil production.

In [5], backbones are identified through particle tracking methods that find the fractures where most of the transport of particles occurs. However, the computational domain characterizing a DFN can be quite large and exhibit a great deal of geometrical complexity, therefore transport and flow simulations turn out to be very costly, even if recent literature has proposed several approaches to overcome these problems; to mention a few: we recall here papers based on reformulations as lower-dimensional problems [6–8]; papers based on the use of partially conforming meshes [9–12]; other interesting geometrical approaches are proposed in [13–18]; approaches consisting in a reformulation of the problem as a PDE-constrained optimization problem [19–24], that can be used in conjunction with several space discretizations [25, 26]. Nonetheless, computational simulation on a large DFN is still a costly task, and it may be prohibitive to perform a large number of simulations.

Due to the expensive cost of particle tracking simulations, other backbone identification methods based on graph topology and Machine Learning (ML) have been developed [4,27–31]. These methods usually train the learning algorithms as binary classifiers for single fractures

* Corresponding author at: Department of Mathematical Sciences, Politecnico di Torino, Turin, Italy.
E-mail address: francesco.dellasanta@polito.it (F. Della Santa).

https://doi.org/10.1016/j.jocs.2021.101458
Received 21 February 2021; Received in revised form 12 September 2021; Accepted 21 September 2021
Available online 2 October 2021

(backbone fracture or not) on datasets built using particle flow simulations; in particular, in [30] the fractures are labeled with respect to the Flow Topology Graph (FTG), while in [4] with respect to the mass flux. Then, the quality of the backbones (identified through a classification of the DFN's fractures) is quantitatively evaluated measuring the similarity between the breakthrough curves of the DFN and of the backbones; i.e., the time spent by particles to flow from the source to the sink of the network. The great advantage of the overmentioned ML methods is that, given a sufficiently large dataset of classified fractures for the training, the backbone identification is performed in a fast and accurate way, that is extremely useful in the framework of Uncertainty Quantification (UQ), where a large number of simulations is required; indeed, one of the main issues related to DFNs is the lack of deterministic information about geometrical and hydro-geological fracture features. This information is typically only known by means of probability distribution, and data needed for actual simulations are typically sampled from the available distributions. All the cited backbone identification methods take into account, as quality criterion of the identified backbone, the time spent by particles to flow from the source to the sink of the network, such that the breakthrough curve or the first passage time of the particles through the "backbone-reduced" DFN is approximately equal to the one of the full DFN. However, in some problems the quantity of interest (QoI) may be, for example, the total flux exiting the DFN or flowing in some direction; in this work, for identifying the backbone, we focus on the total flux outflowing the network. Then, for a given DFN, the target is to identify a backbone such that its exiting flux approximates the one of the full DFN. The method is tested and applied in the framework of a DFN with fixed geometry but with stochastic fracture transmissivities; in particular, the method illustrated is able to identify a backbone of the given DFN sufficiently good (with respect to the approximation of the QoI) for every possible sample of fracture transmissivities, being useful for analyzing clogging conditions or preventing leakage problems.

The method that we propose is based on the Layer-wise Relevance Propagation (LRP) algorithm [32,33] applied to Neural Networks (NNs) trained for flux regression of DFNs [34,35]. LRP is part of the family of the eXplainable AI algorithms, introduced in recent years [36] to gain insight about NN predictions. While most applications of LRP concern NNs trained on image datasets for classification tasks (e.g. [32,37]), here we apply LRP to a NN trained on physical simulation data for a regression task. Furthermore, LRP usage is here characterized by an innovative way of application; indeed, we do not run the LRP on input data one by one, looking at the most relevant features for each single prediction of the NNs, but we compute an approximation of the *expected relevance scores* of all the features in the domain space. In this way, we are able to use the LRP as a feature selection method and therefore identify the backbone fractures of the DFN as the ones with higher expected relevance score. Finally, we highlight the effectiveness of the LRP-based feature selection method by checking the quality of the identified backbone, running the DFN simulations on the corresponding subnetwork.

The method proposed herein can be very useful in applications requiring the identification of the most conducting fractures of a DFN in which the fracture transmissivities are described by means of a statistical distribution; in particular the backbone obtained with this method can be extremely effective in the analysis of clogging conditions (see e.g., [38]) and leakage problems for polluting substances (for example CO_2, see e.g. [39] and [40]). Indeed, since the backbones returned by our method are identified in the framework of UQ, they are statistically robust with respect to changes in fracture transmissivities; then, for example, a user is able to statistically know which fractures are more relevant for flux and should be avoided for waste storage problems, or which fractures are most critical in flux propagation and can be important in clogging problems. On the other hand, the backbones obtained with the method proposed are not useful in UQ analyses of the fluxes, since the trained NNs required by the method are clearly faster and more precise than flow simulations on backbones.

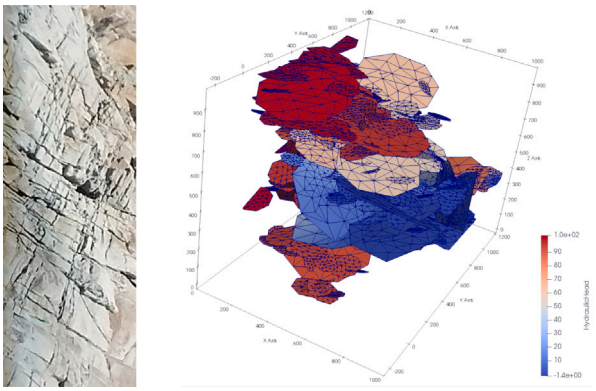

Fig. 1. External surface of a natural fractured medium (left) and a DFN (right).

The paper is organized as follows. In Section 2 the DFN model is briefly recalled. In Section 3 the use of Neural Networks for approximating the DFN simulations is described, after a brief introduction about the notations used for the NNs. In Section 4 the LRP method is described; in particular, in the section the method is recalled and in the Appendix A we introduce some new formal definitions in place of the typical examples used in literature (e.g., see [33]). At the end of Section 4, the new usage of LRP as feature selector is described. In Section 5 the application of NNs for flux regression and LRP to identify backbones is proposed, and numerical results are presented on a case study consisting of two test case DFNs. We end with some conclusions in Section 6.

2. Brief description of discrete fracture network model

We recall here, for the reader's convenience, the model problem of the Discrete Fracture Networks (DFNs) flow simulations. For full details, we point the interested reader to [19,21], while a sketch of the numerical approach here used in the simulations is given in the supplementary materials of this work.

A DFN is a discrete model used to describe and characterize a network of underground connected fractures in a fractured rock medium as a set of 2D polygons in the 3D space $\mathbb{R}^3$ (see Fig. 1). Each polygon represents a fracture and is labeled with an index in a set I; then, each fracture is denoted by $\mathcal{F}_i$, with $i \in I$. A DFN is composed by the union of all the fractures:

$$\cup_{i \in I} \mathcal{F}_i.$$

Each fracture is endowed with its own size and orientation in the 3D space and with its own transmissivity parameter κ_i for the flux characterization; all these data are typically sampled from suitable distributions. Segments given by the intersection of two or more fractures are called *traces* and characterize the connectivity of the network; then, DFNs can be represented as graphs with fractures as nodes and traces as edges.

The flow simulations in a DFN are characterized not only by its geometry but also by the hydrogeological properties conferred to the fractures, such as the transmissivity. The transmissivity parameter κ_i of $\mathcal{F}_i$, for each $i \in I$, represents the flow facilitation through the fracture and it is fundamental for the flow characterization in the DFN; in the next Section we focus on the NN regression problem for predicting the outflowing fluxes of a DFN given the transmissivities of its fractures.

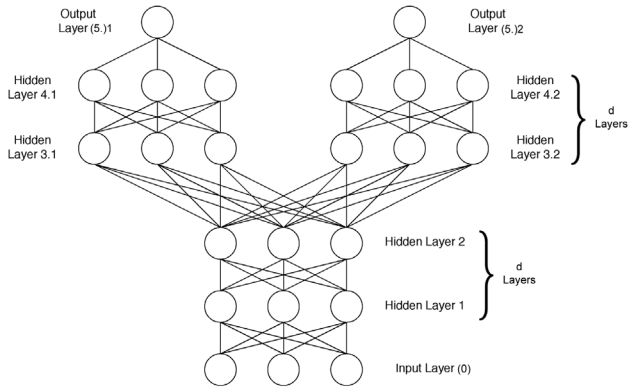

Fig. 2. Example of NN built for vector valued regression concerning flux prediction ($n = 3$, $m = 2$, $d = 2$). For simplicity, biases have not been represented.

3. Neural networks for flux regression in discrete fracture networks

In this section we briefly recall the method used in [34] to address the problem of flux prediction in DFNs using NNs, with an improvement that enhance flux approximation; we start fixing the notation adopted in this work for the description of a generic Feedforward Neural Network (FNN).

Given a FNN $\mathcal{N}$ made of $L + 1 \in \mathbb{N}$ layers, we denote them as $U^{(0)}, \dots, U^{(L)}$ where:

- $U^{(0)}$ is the input layer;
- $U^{(L)}$ is the output layer;
- $U^{(1)}, \dots, U^{(L-1)}$ are the hidden layers.

Let $\mathcal{N}$ be a FNN trained for approximating ($\approx$) a function $\boldsymbol{F} : A \subseteq \mathbb{R}^n \to \mathbb{R}^m$, then we let $\widehat{\boldsymbol{F}}$ denote the function corresponding to $\mathcal{N}$ at the end of the training; therefore, assuming that $\mathcal{N}$ is "well-trained", it holds $\boldsymbol{F} \approx \widehat{\boldsymbol{F}}$ or, more precisely, $\boldsymbol{F}(\boldsymbol{x}) \approx \widehat{\boldsymbol{F}}(\boldsymbol{x})$ for each $\boldsymbol{x} \in A \subseteq \mathbb{R}^n$.

In the following subsection we introduce the multi-task architecture at the core of our approach, showing its performances.

3.1. Multi-task architecture

In this work we use the same architecture introduced in [34] for the approximation of a function $\boldsymbol{F} : A \subseteq \mathbb{R}^n \to \mathbb{R}^m$. These NNs are characterized by a "tree-shaped" structure (see Fig. 2), obtained by extending the one described in [41, chapter 7.7] for multi-task learning. In particular, given a hyperparameter $d \in \mathbb{N}$, the NN architecture is given by:

- One input layer $U^{(0)}$ of n units;
- A sequence of d hidden layers $U^{(1)}, \dots, U^{(d)}$, each one made of n units with softplus ($f(x) = \log(1 + e^x)$) activation function, such that $U^{(\ell-1)}$ is fully connected to $U^{(\ell)}$, for each $\ell = 1, \dots, d$. We call "*trunk* of the NN" the sequence $U^{(0)}, \dots, U^{(d)}$;
- m sequences of d hidden layers $U_j^{(d+1)}, \dots, U_j^{(2d)}$, each one made of n softplus units, followed by one output layer $U_j^{(2d+1)}$ made of one linear unit for each $j = 1 \dots, m$. These layers are such that $U^{(d)}$ is fully connected to $U_j^{(d+1)}$ and $U_j^{(\ell-1)}$ is fully connected to $U_j^{(\ell)}$, for each $\ell = d + 2, \dots, 2d + 1$, for each $j = 1, \dots, m$. We call "*branches* of the NN" all the m sequences $U_j^{(d+1)}, \dots, U_j^{(2d+1)}$.

The choice of using softplus functions for the hidden layers was made after a preliminary investigation in [34], comparing the performances obtained also with other activation functions.

3.2. Regression problem setting

The problems addressed in this paper are characterized as follows. We consider DFNs consisting of n fractures, with fixed geometrical properties, immersed in a cubic matrix block with a 1000 meters long edge. See Fig. 3 for two examples.

We set the boundary conditions in such a way that two opposite faces of the block represent an inlet and outlet face, respectively; namely, we impose a Dirichlet boundary condition $H = 10$ on fracture edges created intersecting the DFN with the leftmost face of the domain, corresponding to $x = 0$, and $H = 0$ on the edges obtained intersecting the DFN with the rightmost face ($x = 1000$). The fractures intersecting such faces are called inflow and outflow fractures, respectively. All other fracture edges are insulated (homogeneous Neumann condition). The boundary conditions and the geometry of the DFN mainly affect the flux directionality, and the transmissivities have a great impact on the flow intensity on each outflow fracture. We consider here, as a target, the prediction of the exiting fluxes and their distributions among the outflow fractures.

The fracture transmissivities are assumed to be isotropic parameters $\kappa_1, \dots, \kappa_n$ modeled as random variables with log-normal distribution [28,42]:

$$\log_{10} \kappa_i \sim \mathcal{N}(-5, 1/3)\,. \tag{1}$$

We consider in particular two test cases (DFN158 and DFN202) characterized by $n = 158$ and $n = 202$ fractures, respectively (see also [34]). The fractures are assumed to be octagons, and have been randomly generated using the following distribution for the geometrical features [43,44]: fracture radii have been sampled with respect to a truncated power law distribution, with exponent $\gamma = 2.5$ and upper and lower cut-off $r_u = 560$ and $r_0 = 50$, respectively; the fracture orientations have been sampled from a Fischer distribution having mean direction $\boldsymbol{\mu} = (0.0065, -0.0162, 0.9998)$ and dispersion parameter 17.8; uniform distribution has been used for mass centers.

The resulting number of outflow fractures for DFN158 and DFN202 is $m = 7$ and $m = 14$, respectively.

3.2.1. Dataset characterization

Let us introduce the following notation. Let $\boldsymbol{\kappa} = [\kappa_1, \dots, \kappa_n]^\top \in \mathbb{R}^n$ be the vector collecting transmissivities of all fractures of the given DFN and let $\boldsymbol{\varphi} = [\varphi_1, \dots, \varphi_m]^\top \in \mathbb{R}^m$ be the vector collecting all the exit flows. Let $\boldsymbol{F} : A \subseteq \mathbb{R}^n \to \mathbb{R}^m$ be a function defined by

$$\boldsymbol{\varphi} = \boldsymbol{F}(\boldsymbol{\kappa}), \tag{2}$$

that is the function that provides the vector of outflows associated to the transmissivity input $\boldsymbol{\kappa}$.

Let us consider $D \in \mathbb{N}$ samples $\boldsymbol{\kappa}_k \in \mathbb{R}^n$, $k = 1, \dots, D$, drawn according to distribution (1). The dataset $\mathcal{D}$ used for the creation of the training set, the test set and the validation set is

$$\mathcal{D} = \left\{(\boldsymbol{\kappa}_k, \boldsymbol{\varphi}_k) \in \mathbb{R}^n \times \mathbb{R}^m \mid \boldsymbol{F}(\boldsymbol{\kappa}_k) = \boldsymbol{\varphi}_k\,,\ \forall\ k = 1, \dots, D\right\}. \tag{3}$$

The test set $\mathcal{P}$ is created by randomly picking approximately 30% of the elements in $\mathcal{D}$. The remaining elements are then randomly split into two subsets $\mathcal{T}$ and $\mathcal{V}$, representing the training set and the validation set, respectively, and such that $|\mathcal{V}| \sim 20\%\, |\mathcal{D} \setminus \mathcal{P}|$.

3.3. Neural network training and performances

We consider the two test cases described in the previous subsection (DFN158 and DFN202). We recall that they are respectively characterized by $n = 158$ and $n = 202$ total fractures, and $m = 7$ and $m = 14$ outflow fractures.

The multi-task architecture previously described is used to build, for each DFN, suitable NNs. We consider the following values, for some hyper-parameters already tested in [34], yielding four different NNs for each DFN:

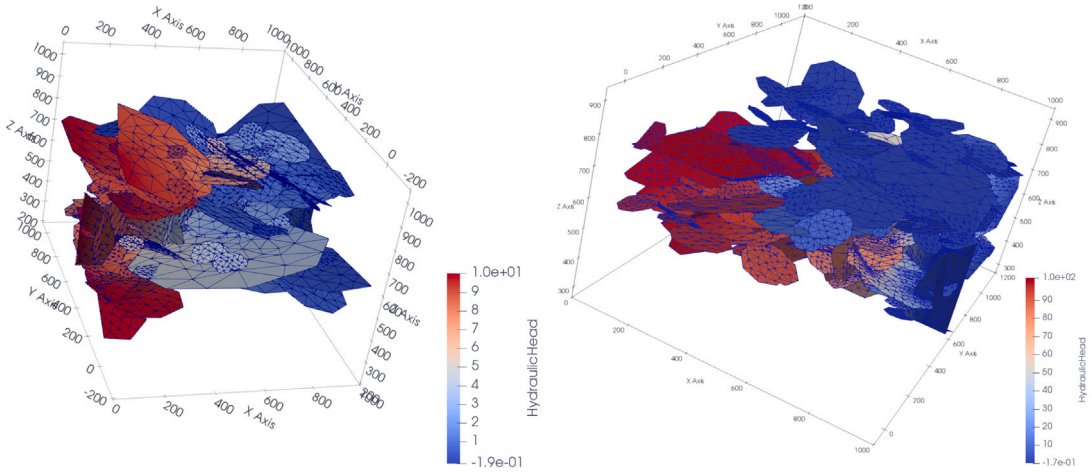

Fig. 3. 3D view of DFN158 (left) and DFN202 (right).

Table 1
Mean relative errors $\mathbb{E}[e^r(\mathcal{N}^B_{n,d}; \mathcal{P}_n)]$ for several values of depth parameter d and mini-batch size B for all the test cases ($n = 158, 202$).

	DFN158		DFN202	
	$d = 1$	$d = 3$	$d = 1$	$d = 3$
$B = 10$	0.0104	0.0085	0.0060	0.0070
$B = 30$	0.0099	**0.0082**	0.0057	**0.0055**

- *depth parameter* $d \in \{1, 3\}$ (the depth of the NN being equal to $2d$);
- *mini-batch size* $B \in \{10, 30\}$;
- a number n of units for the input layer and hidden layers coinciding with the number of fractures;
- the "tree-shaped" structure has $m = 7$ branches for DFN158 and $m = 14$ branches for DFN202.

We refer to these NNs and options as

$$\mathcal{N}^B_{n,d}\,, \quad \forall\, d \in \{1,3\}\,, \quad \forall\, B \in \{10,30\}\,, \tag{4}$$

they are trained and tested with respect to a dataset $\mathcal{D}_n$ (see Section 3.2.1) of 10 000 pairs $(\boldsymbol{\kappa}_k\,, \boldsymbol{\varphi}_k) \in \mathbb{R}^n \times \mathbb{R}^m$, in order to make predictions of the outflowing fluxes of DFN158 and DFN202. The training is made using the optimizer `adam` [45], with a maximum number of epochs $c_{\max} = 1000$, mini-batch size B and two regularization methods: early stopping method, with patience parameter $p^* = 150$, and "minimum validation error" method. Then, for each fixed $n = 158, 202$, the two networks $\mathcal{N}^*_n$ with best performances are selected, taking into account a grid search approach with respect to the values of d and B and using as performance measure the mean value of the global relative errors of the predictions of $\mathcal{N}^B_{n,d}$ on the test set $\mathcal{P}_n$ (see Table 1); for simplicity we indicate this quantity as $\mathbb{E}[e^r(\mathcal{N}^B_{n,d}; \mathcal{P}_n)]$, defining the vector of relative errors of a prediction $\widehat{\boldsymbol{\varphi}}$ with respect to the total exiting flux [34] as

$$\boldsymbol{e}^r(\widehat{\boldsymbol{\varphi}}) = \frac{|\widehat{\boldsymbol{\varphi}} - \boldsymbol{\varphi}|}{\sum_{i=1}^m \varphi_i} = \frac{1}{\sum_{i=1}^m \varphi_i} \left[\, |\widehat{\varphi}_1 - \varphi_1|\,, \ldots, |\widehat{\varphi}_m - \varphi_m| \,\right]^\top . \tag{5}$$

The results of the grid search are $\mathcal{N}^*_{158} := \mathcal{N}^{30}_{158,3}$ and $\mathcal{N}^*_{202} := \mathcal{N}^{30}_{202,3}$.

The approach here adopted for building and training the NNs is the same used in [34] but with a main difference: a pre-processing phase for the input data has been introduced. Indeed, NN performances often increase when some transformations are applied that normalize input data to have zero mean and standard deviation equal to 1 [46]. Hence, we introduced a function $g : \mathbb{R} \to \mathbb{R}$:

$$g(\kappa_i) = \frac{\log_{10}(\kappa_i) - \mu}{\sigma} =: \widetilde{\kappa}_i \sim \mathcal{N}(0,1)\,, \ \forall\, i = 1, \ldots, n \tag{6}$$

where $\mu = -5$ and $\sigma = 1/3$ are the mean and the standard deviation used in (1), respectively. Then, we trained the NNs as in [34] with respect to a "normalized" version of $\mathcal{D}_n$, that is the dataset $\widetilde{\mathcal{D}}_n$ characterized by "normalized" inputs such that

$$\widetilde{\mathcal{D}}_n = \left\{ (\widetilde{\boldsymbol{\kappa}}_k\,, \boldsymbol{\varphi}_k) \in \mathbb{R}^n \times \mathbb{R}^m \mid (\boldsymbol{\kappa}_k, \boldsymbol{\varphi}_k) \in \mathcal{D}_n \right\}\,, \tag{7}$$

where $\widetilde{\boldsymbol{\kappa}}_k = \boldsymbol{g}(\boldsymbol{\kappa}_k)$ and $\boldsymbol{g} : \mathbb{R}^n \to \mathbb{R}^n$ is the element-wise application of the function g to the components of $\boldsymbol{\kappa}_k$.

In Tables 2 and 3, we report the measurements of the Jensen–Shannon divergence (D_{JS}) for the actual flux distributions and the predicted flux distributions given by the NNs $\mathcal{N}^*_n$ with respect to the inputs of the test set $\widetilde{\mathcal{P}}_n$ (see Fig. 4 for visualizing an example of distribution comparison); we report also an additional relative dissimilarity measure for distributions, the ratio $D_{\mathrm{KL}}/\mathcal{E}$, increasing the interpretability (and comparability) of the values. The dissimilarity measure introduced is defined as the ratio between the Kullback–Leibler divergence (D_{KL}) and the entropy ($\mathcal{E}$) of the actual distribution (see [34] and [41, chapter 3.13]), namely:

$$\frac{D_{\mathrm{KL}}(P \parallel Q)}{\mathcal{E}(P)} = \frac{\mathbb{E}_{x\sim P}\left[\log\left(P(x)/Q(x)\right)\right]}{\mathbb{E}_{x\sim P}\left[\log P(x)\right]}\,, \tag{8}$$

where P is the actual flux's probability distribution of a fracture and Q is the one of corresponding predictions. The interpretability advantages of using ratio (8) derive from the relationship between the D_{KL} and the cross-entropy $\mathcal{E}(P, Q)$ [41, chapter 3.13], since we have:

$$\frac{D_{\mathrm{KL}}(P \parallel Q)}{\mathcal{E}(P)} = \frac{\mathcal{E}(P,Q) - \mathcal{E}(P)}{\mathcal{E}(P)} = \frac{\mathcal{E}(P,Q)}{\mathcal{E}(P)} - 1\,, \tag{9}$$

where

$$\mathcal{E}(P, Q) := \mathbb{E}_{x\sim P}\left[\log Q(x)\right]\,. \tag{10}$$

Indeed, $\mathcal{E}(P, Q)$ measures the average information needed to describe the entropy $\mathcal{E}(P)$ (that is the average information rate of P) using a random sampling from Q; then the ratio (8) represents a sort of "relative information error", measuring the relative missing information when we want to describe the average information rate of P using a random sampling from Q.

4. Layer-wise relevance propagation

Deep Neural Networks, like those considered in this work, are very powerful regression models but often appear as black-box models that, fore each input, return predictions computed through a very complicated function. The difficulties in obtaining and understanding the explicit formula of this function led many users to define NNs "difficult to be explained", since no information about input–output

Table 2
DFN158. Jensen–Shannon divergence and dissimilarity measure between actual and predicted flux distributions.

	$\mathcal{F}_8$	$\mathcal{F}_{12}$	$\mathcal{F}_{14}$	$\mathcal{F}_{78}$	$\mathcal{F}_{90}$	$\mathcal{F}_{98}$	$\mathcal{F}_{107}$
D_{JS}	0.0050	0.0018	0.0063	0.0012	0.0196	0.2144	0.0060
$D_{\mathrm{KL}}/\mathcal{E}$	0.0009	0.0003	0.0010	0.0002	0.0033	0.0379	0.0010

Table 3
DFN202. Jensen–Shannon divergence and dissimilarity measure between actual and predicted flux distributions.

	$\mathcal{F}_8$	$\mathcal{F}_{15}$	$\mathcal{F}_{18}$	$\mathcal{F}_{31}$	$\mathcal{F}_{61}$	$\mathcal{F}_{73}$	$\mathcal{F}_{93}$
D_{JS}	0.0050	0.0941	0.0226	0.0033	0.0021	0.0014	0.0324
$D_{\mathrm{KL}}/\mathcal{E}$	0.0007	0.0177	0.0040	0.0005	0.0003	0.0003	0.0055
	$\mathcal{F}_{115}$	$\mathcal{F}_{156}$	$\mathcal{F}_{162}$	$\mathcal{F}_{173}$	$\mathcal{F}_{176}$	$\mathcal{F}_{180}$	$\mathcal{F}_{187}$
D_{JS}	0.0256	0.0928	0.0059	0.0016	0.0327	0.0069	0.0570
$D_{\mathrm{KL}}/\mathcal{E}$	0.0042	0.0165	0.0014	0.0002	0.0039	0.0016	0.0097

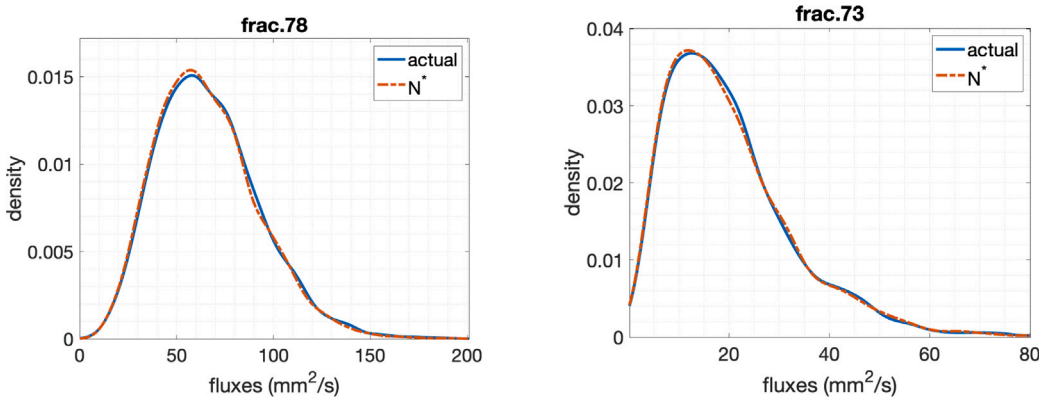

Fig. 4. DFN158 case (left) and DFN202 case (right). Example of two comparisons between probability density functions of the actual flux distribution (continuous line) and the predicted flux distribution (dotted line) for one of the outflowing fractures, done by $\mathcal{N}_n^*$.

relationships is given or easily understood and, moreover, the users do not have a clear view of the mathematical operations performed on input data by the NN in order to return the output. The lack of transparency of the inner computations was considered as one of the greatest limits of these instruments, therefore several methods has been recently introduced to tackle this issue [32,33,47–49].

In this work we focus on the Layer-wise Relevance Propagation (LRP) [32]. LRP defines a subclass of "eXplainable AI" (XAI) algorithms [36] that, given a trained NN with function $\widehat{\boldsymbol{F}}$, looks at a prediction $\widehat{\boldsymbol{F}}(\boldsymbol{x})$ and assigns relevance scores to the components x_k of $\boldsymbol{x}$, taking into account the weights and the architecture of the NN. The computed relevance scores indicate how much each x_k contributed in computing the prediction $\widehat{\boldsymbol{F}}(\boldsymbol{x})$. Given a NN model, an input $\boldsymbol{x}$ and a value $R \in \mathbb{R}$ (defined score) related to $\widehat{\boldsymbol{F}}(\boldsymbol{x})$, the method propagates the score R backward through the NN from the output layer $U^{(L)}$ to the input one $U^{(0)}$, redistributing it among all the input units of $U^{(0)}$. The redistribution of R depends on a propagation rule that takes into account the forward propagation of $\boldsymbol{x}$ through the NN and, therefore, depends on the weights and the network connectivity. The final result describes the components x_k of $\boldsymbol{x}$ that have higher influence in the computation of the corresponding prediction made by the NN. Actually, these components are the list of values that come up to the input units from the propagation of R.

In literature, LRP is always applied with respect to one input at a time: given one input $\boldsymbol{x}$ and a score R, LRP computes the relevance of the components x_k of $\boldsymbol{x}$ for the prediction of the output. Then, the relevance is considered as a characteristic of the input and, therefore, the most relevant components can vary changing the input taken into account. A major difference in this work is that the authors not only compute the relevance scores of all the inputs in a given dataset but they also aggregate this information (see Section 4.2 and Section 5). The result is an approximation of the expected relevance score vector that furnishes a description about how the NN looks at the domain space. Then, this vector can be used to perform feature selection, identifying the most relevant fractures for the predictions and, therefore, the backbone of the DFN.

4.1. The propagation rule

To understand the LRP method, we describe the mechanism that regulates the propagation of the score R backward through the NN; this sequence of operations is called "propagation rule" of the method and can vary according to the NN architecture and the regression or classification problem considered. In this section we describe the main characteristics of the propagation rule and we introduce the α-β rule adopted for the experiments illustrated in the next Section.

One of the most important aspects that characterizes the propagation rule is the criterion behind the choice of the score R with respect to the input $\boldsymbol{x}$ of the NN, at the beginning of the process, as several choices can be made. In this work we consider the simplest and most used case in literature, taking the starting score equal to the NN prediction corresponding to $\boldsymbol{x}$ [50]; see Section 5 for a discussion about this choice in the framework of backbone identification. If the output layer $U^{(L)}$ of the network is given by more than one unit, the criterion is generalized in such a way that it assigns to each output unit $u_j \in U^{(L)}$ a relevance score $R_j^{(L)}$ equal to the jth component of the prediction vector corresponding to input $\boldsymbol{x}$, i.e.:

$$R_j^{(L)}(\boldsymbol{x}) = (\widehat{\boldsymbol{F}}(\boldsymbol{x}))_j \,. \tag{11}$$

Then the total starting score is

$$R(\boldsymbol{x}) = \sum_{u_j \in U^{(L)}} R_j^{(L)}(\boldsymbol{x}) =: R^{(L)}(\boldsymbol{x}) \,. \tag{12}$$

For the ease of notation, from now on we drop the dependency of the relevance scores on the input $\boldsymbol{x}$.

Assuming for simplicity to have a FNN, LRP method defines a rule to propagate these scores from $U^{(L)}$ to $U^{(L-1)}$ and, more generally, from each layer to the previous one. The rule comprises the definition of messages, for each pair $(u_i, u_j) \in U^{(\ell)} \times U^{(\ell+1)}$, for each $\ell = 0, \dots, L-1$, such that the message $R_{i \leftarrow j}^{(\ell,\ell+1)} \in \mathbb{R}$ is the amount of score $R_j^{(\ell+1)}$ that spread to unit $u_i \in U^{(\ell)}$ from unit $u_j \in U^{(\ell+1)}$. Actually, the message $R_{i \leftarrow j}^{(\ell,\ell+1)}$ represents how much the output of unit u_i sent to u_j is relevant for the prediction computation $\widehat{\boldsymbol{F}}(\boldsymbol{x})$; then, the relevance score $R_i^{(\ell)}$ of u_i with respect to $\widehat{\boldsymbol{F}}(\boldsymbol{x})$ is given by the sum of all the "partial" relevances (i.e. the messages):

$$R_i^{(\ell)} = \sum_{u_j \in U^{(\ell+1)}} R_{i \leftarrow j}^{(\ell,\ell+1)}, \tag{13}$$

for each $\ell = 0, \dots, L-1$. Therefore, the relevance of the component x_i of the input $\boldsymbol{x} \in \mathbb{R}^n$ is given by the quantity $R_i^{(0)}$ computed starting from $R(\boldsymbol{x})$.

Due to the empirical origin of the LRP method, in literature (e.g., see [33]) the computation of the messages is usually described through examples that show many possible arbitrary formulas for the computation of $R_{i \leftarrow j}^{(\ell,\ell+1)}$ but, to the best of the authors knowledge, no formal definitions exist. Then, to facilitate the understanding of the problem, in Appendix A we introduce a more general and formal definition of the messages $R_{i \leftarrow j}^{(\ell,\ell+1)}$ characterizing the propagation rule. The content of Appendix A can be useful to the interested reader that is new to the LRP algorithm and, in general, for a formal generalization of LRP to FNNs characterized by multi-task architecture.

4.2. Expected relevance score: LRP for feature selection in NNs

LRP method has been widely used in applications for explanation of NNs concerning images but rarely for explanation of regression NNs (e.g. [32,37]). Probably due to this reason, to the best of the authors knowledge, the usage of LRP as feature selection method proposed in this paper has never been considered before.

The main idea behind the feature selection performed using LRP is the following: compute the *expected relevance scores*

$$\mathbb{E}_{\boldsymbol{x}\sim q}[R_1^{(0)}], \dots, \mathbb{E}_{\boldsymbol{x}\sim q}[R_n^{(0)}] \tag{14}$$

for the components $x_1, \dots, x_n$, respectively, of a random input vector $\boldsymbol{x} \in \mathbb{R}^n$ with distribution q, with respect to a given FNN $\mathcal{N}$ with function $\widehat{\boldsymbol{F}} : \mathbb{R}^n \to \mathbb{R}^m$. Then, the most important features for $\widehat{\boldsymbol{F}}$ are the ones characterized by a higher expected relevance score. Furthermore, assuming that $\widehat{\boldsymbol{F}}(\boldsymbol{x}) \approx \boldsymbol{F}(\boldsymbol{x})$ for each $\boldsymbol{x} \in A \subseteq \mathbb{R}^n$, LRP allows to perform indirectly (through $\mathcal{N}$) a feature selection also with respect to the exact function $\boldsymbol{F} : A \subseteq \mathbb{R}^n \to \mathbb{R}^m$.

From a practical point of view, we can approximate the vector of expected relevance scores

$$\bar{\boldsymbol{r}} = \bar{\boldsymbol{r}}^{(0)} := \mathbb{E}_{\boldsymbol{x}\sim q}[\boldsymbol{r}^{(0)}] = \mathbb{E}_{\boldsymbol{x}\sim q}\left[\left[R_1^{(0)}, \dots, R_n^{(0)}\right]^\top\right] = \left[\mathbb{E}_{\boldsymbol{x}\sim q}[R_1^{(0)}], \dots, \mathbb{E}_{\boldsymbol{x}\sim q}[R_n^{(0)}]\right]^\top \tag{15}$$

computing the vector of mean relevance scores with respect to a given set $S \subset \mathbb{R}^n$ of S samplings of $\boldsymbol{x}$, i.e.:

$$\bar{\boldsymbol{r}}(S) = \bar{\boldsymbol{r}}^{(0)}(S) := \mathbb{E}_S[\boldsymbol{r}^{(0)}] = \frac{1}{S}\sum_{\boldsymbol{x}\in S} \boldsymbol{r}^{(0)} = \frac{1}{S}\sum_{\boldsymbol{x}\in S}\left[R_1^{(0)}, \dots, R_n^{(0)}\right]^\top \approx \bar{\boldsymbol{r}}, \tag{16}$$

where $\bar{\boldsymbol{r}}(S) = \bar{\boldsymbol{r}}$ for $S = |S| \to +\infty$.

4.2.1. Testing of the expected relevance scores

We end this section introducing the strategy used to evaluate the effectiveness of our approach. Since we aim at using the expected relevance scores returned by LRP to select an arbitrary number $p \in \mathbb{N}$, $p \le n$, of most relevant features in the domain of $\boldsymbol{F}$, the strategy used in this work to evaluate the behavior of our method is different from the ones usually adopted for LRP.

A common criterion used to evaluate a local XAI algorithm, like LRP, is the so called *excluding criterion*. This criterion consists in analyzing how the predicted NN outcome changes if the p most important features of an input $\boldsymbol{x}$, identified by the XAI algorithm, are modified to an uninformative neutral value, obtaining an altered input $\boldsymbol{x}'$ [47,49]. Then, given a second altered input $\boldsymbol{x}_{\text{rand}}$ obtained from $\boldsymbol{x}$ setting p random components to the neutral value, the excluding criterion analyzes how the predictions $\widehat{\boldsymbol{F}}(\boldsymbol{x}')$ and $\widehat{\boldsymbol{F}}(\boldsymbol{x}_{\text{rand}})$ change with respect to $\widehat{\boldsymbol{F}}(\boldsymbol{x})$: if the p most important input features of $\boldsymbol{x}$ have been properly identified by the algorithm, the distance between $\widehat{\boldsymbol{F}}(\boldsymbol{x}')$ and $\widehat{\boldsymbol{F}}(\boldsymbol{x})$ is significantly greater than the distance between $\widehat{\boldsymbol{F}}(\boldsymbol{x}_{\text{rand}})$ and $\widehat{\boldsymbol{F}}(\boldsymbol{x})$. In computer vision, where LRP has been mainly applied, each input feature is a pixel color and the neutral value is usually assumed to be the gray color.

However, in the DFN context the excluding criterion is not easily applicable due to the difficulty to define a suitable neutral value, and to the obstacles to extend the criterion from a local XAI method to a global one based on the expected relevance scores. Therefore, here we propose a novel criterion, called *retaining criterion*, that involves the usage of the DFN simulator represented by the approximated function $\boldsymbol{F} : A \subseteq \mathbb{R}^n \to \mathbb{R}^m$. Let $\boldsymbol{F}_{|p} : A_{|p} \subseteq \mathbb{R}^p \to \mathbb{R}^m$ be a restriction of $\boldsymbol{F}$ with respect to the p most relevant features of a given input $\boldsymbol{x}$, identified by the XAI algorithm; analogously, let $\boldsymbol{x}_{|p} \in A_{|p}$ be the restriction of $\boldsymbol{x} \in A \subseteq \mathbb{R}^n$. Then, the criterion consists in comparing $\boldsymbol{F}_{|p}(\boldsymbol{x}_{|p})$ and $\boldsymbol{F}(\boldsymbol{x})$: if $\boldsymbol{F}_{|p}(\boldsymbol{x}_{|p}) \approx \boldsymbol{F}(\boldsymbol{x})$, the algorithm has correctly identified the p most relevant features of $\boldsymbol{x}$. In DFN terms, given a set of p most relevant fractures, we run a DFN flow simulation on the subnetwork given by these fractures, with the corresponding transmissivity values $\boldsymbol{x}_{|p}$, while removing all the other fractures; then, we compare the total outflowing flux of this subnetwork with the one of the full DFN.

The retaining criterion can also be extended easily to a global XAI algorithm like the one we defined in Section 4.2. In this case the p most relevant features are the p ones with highest expected relevance score and, therefore, are fixed for each input $\boldsymbol{x} \in \mathbb{R}^n$. Then, the expected relevance scores returned by LRP perform a good feature selection of p features if $\boldsymbol{F}_{|p}(\boldsymbol{x}_{|p}) \approx \boldsymbol{F}(\boldsymbol{x})$ for each $\boldsymbol{x} \in A \subseteq \mathbb{R}^n$.

In general, we observe that the new retaining criterion is characterized by the following advantages over the excluding criterion:

- absence of a neutral value usage and definition;
- easily extendable to a global XAI algorithm for feature selection evaluation;
- the possibility to compare the effects of different choices of p directly looking at actual simulator outputs instead of model predictions.

Then, the retaining criterion is adopted for evaluating the quality of the backbones identified with the new method described in the next section.

5. Main results

Let us consider a NN $\mathcal{N}_n^* \in \{\mathcal{N}_{158}^*, \mathcal{N}_{202}^*\}$ (see Section 3.3) and a general input vector $\tilde{\boldsymbol{\kappa}}$ of (normalized) transmissivities for $\mathcal{N}_n^*$. Since we have $\widehat{\boldsymbol{F}}(\tilde{\boldsymbol{\kappa}}) \approx \boldsymbol{F}(\boldsymbol{\kappa})$ up to a good accuracy (Tables 1–3), the application of LRP method to $\mathcal{N}_n^*$ with respect to a given $\tilde{\boldsymbol{\kappa}}$ returns a vector of relevance scores

$$\boldsymbol{r}^{(0)} = \left[R_1^{(0)}, \dots, R_n^{(0)}\right]^\top$$

that is a vector characterizing the relevance of fractures $\mathcal{F}_i$ in the DFN during the computation of the fluxes $\boldsymbol{\varphi} = \boldsymbol{F}(\boldsymbol{\kappa})$, for each $i = 1, \dots, n$. In

particular, due to the conservation property of LRP (see Appendix A), we can observe that initialization of the scores as in (11) allows the relevance scores of the most relevant fractures $\mathcal{F}_i$ to increase when the sum of the predicted fluxes $\sum_{j=1}^{m} \widehat{\varphi}_j = R(\widetilde{\boldsymbol{\kappa}})$ is large; the ratio behind choice (11) is that fracture relevances related to larger (predicted) fluxes will give a greater contribution to the estimate of the expected relevance score of the fractures.

Given the interpretation of LRP relevance scores as fracture relevances in the DFN for the single simulation $\boldsymbol{F}(\boldsymbol{\kappa})$, we can use LRP as feature selection method with respect to the fluxes provided by $\mathcal{N}_n^*$ (see Section 4.2) and assuming $\widehat{\boldsymbol{F}}(\widetilde{\boldsymbol{\kappa}}) \approx \boldsymbol{F}(\boldsymbol{\kappa})$ for each transmissivity vector $\boldsymbol{\kappa} \in \mathbb{R}^n$, we can interpret the vector of expected relevance scores $\bar{\boldsymbol{r}} \in \mathbb{R}^n$ as a measure of the expected relevance of the fractures $\mathcal{F}_1, \dots, \mathcal{F}_n$ in the DFN, for any random $\boldsymbol{\kappa}$ sampled.

As a consequence, if this interpretation of LRP relevance scores is correct, a collection of the "most relevant" fractures (e.g. the ones with $\mathbb{E}_{\boldsymbol{\kappa}\sim q_{\boldsymbol{\kappa}}}[R_i^{(0)}]$ greater than an arbitrary threshold) can be interpreted as a possible backbone of the DFN, where the target Quantity of Interest (QoI) to be preserved is the total flux exiting from the DFN. For this reason, in order to validate the approach proposed, in this section we analyze the fluxes obtained running simulations on sub-networks of both DFN158 and DFN202, obtained selecting the fractures through LRP applied on $\mathcal{N}_{158}^*$ and $\mathcal{N}_{202}^*$, respectively, and comparing them with "full" simulations on the whole DFNs. From these comparisons, we obtain a validation of both the LRP-based feature selection's quality (see Section 4.2.1) and the sub-networks as backbones.

5.1. Direct method

Let us consider the NNs $\mathcal{N}_{158}^*, \mathcal{N}_{202}^*$ and the corresponding datasets $\widetilde{\mathcal{D}}_{158}, \widetilde{\mathcal{D}}_{202}$ (see (7)), each one with cardinality $|\widetilde{\mathcal{D}}_{158}| = |\widetilde{\mathcal{D}}_{202}| = 10\,000$ (see Section 3.3). For each $\mathcal{N}_n^* \in \{\mathcal{N}_{158}^*, \mathcal{N}_{202}^*\}$ we compute the vector of mean relevance scores with respect to the set $\widetilde{\mathcal{D}}_n$, i.e. the vector

$$\bar{\boldsymbol{r}}_n := \bar{\boldsymbol{r}}(\widetilde{\mathcal{D}}_n) = \mathbb{E}_{\widetilde{\mathcal{D}}_n}[\boldsymbol{r}^{(0)}] = \frac{1}{|\widetilde{\mathcal{D}}_n|} \sum_{(\widetilde{\boldsymbol{\kappa}},\boldsymbol{\varphi})\in\widetilde{\mathcal{D}}_n} \boldsymbol{r}^{(0)}, \tag{17}$$

using an LRP method characterized by the α-β rule with $\alpha = 1$ and $\beta = 0$ (see (A.10), Appendix A). Then, looking at the values of $\bar{\boldsymbol{r}}_n$ and recalling that $\bar{\boldsymbol{r}}_n \approx \mathbb{E}_{\boldsymbol{\kappa}\sim q_{\boldsymbol{\kappa}}}[\boldsymbol{r}^{(0)}]$, we create a hierarchy for the fractures of the DFN such that $\mathcal{F}_i$ is "less relevant than or as relevant as" $\mathcal{F}_j$ if

$$\left(\bar{\boldsymbol{r}}_n\right)_i = \mathbb{E}_{\widetilde{\mathcal{D}}_n}[R_i^{(0)}] \le \mathbb{E}_{\widetilde{\mathcal{D}}_n}[R_j^{(0)}] = \left(\bar{\boldsymbol{r}}_n\right)_j, \tag{18}$$

for each $i, j \in \{1, \dots, n\}$. In Fig. 5 we visualize the sorted set of fractures with the corresponding mean relevance scores $\left(\bar{\boldsymbol{r}}_n\right)_i$, both for DFN158 and for DFN202.

A first observation about the element values of $\bar{\boldsymbol{r}}_n$, for each $n = 158, 202$, is that all the boundary fractures of the DFNs with exiting flux belong to the set of fractures in the top 25% with highest relevance scores. This observation has non-trivial consequences; in fact it is an important clue that the NNs $\mathcal{N}_n^*$ learned to approximate $\boldsymbol{F}$ coherently with the topology of the network of fractures characterizing the DFNs. Indeed, although one may think that it is obvious that NNs mainly look at the inputs corresponding to the fractures with exiting flux, we should remember that the NNs $\mathcal{N}_n^*$ have no information about relationships between inputs and outputs, with the exception of the coupling they "observe" during the training; in particular, assuming that $\mathcal{F}_i$ is a boundary exit fracture, no information about the strict physical-based relationship between the transmissivity κ_i and the computed flux have been given to the NN.

The only exception to the general behavior observed for the outflux fractures, is fracture $\mathcal{F}_{156}$ in DFN202; $\mathcal{F}_{156}$ is indeed an exit fracture, but looking at the actual flux statistics for DFN202, we can observe that $\mathcal{F}_{156}$ is characterized by an extremely low flux, especially with respect to the ones of the other outflowing boundary fractures. With reference to the box in Fig. 5 (bottom), the fracture corresponds to the leftmost column with crossing-lines texture. In general we can observe that, for fractures with exiting fluxes, the mean relevance score is characterized by a non-negligible dependence on the mean value of the flux (Tables 4, 5); indeed an average monotonically increasing trend is observed and reported in Fig. 6.

We continue our analysis focusing on the sub-networks of both DFN158 and DFN202 given by the set of fractures in the top 25%, 50%, 75% relevance scores, respectively. First of all, we analyze the topology of the graphs that characterize the networks. Comparing the graph of the full DFNs and the graphs of the sub-DFNs given by the 25%, 50%, 75% "most relevant" fractures (Figs. 7 and 8) we can observe what follows:

1. The less relevant fractures are in general those belonging to "dead-end" branches of the networks, since they are the first fractures removed when pruning the DFN graphs to keep only the 75% most relevant ones;
2. Pruning further the DFN graphs (50% of most relevant fractures), other fractures are removed, which are not dead-end fractures but belong to source–sink paths (i.e., paths in the graph that start from any inlet fracture and end in any outlet fracture); removing this fractures is likely to reduce the number of source–sink paths. However, it is worth noting that at least one source–sink path is always left. In particular, the bottleneck fractures (i.e., the cut nodes of the graphs) belonging to source–sink paths are not removed. A clear example is the single fracture that keeps connected the two main halves of DFN158 (see Fig. 7);
3. Pruning too much the graphs (25% of most relevant fractures), some outflow fractures get disconnected, but the overall network connectivity is preserved. Some bottleneck nodes can also be removed from the graphs (see Fig. 8);
4. In the DFN158 case, the LRP algorithm seems to be more sensitive to the actual physic relevance of the fractures in the flux problem than in the DFN202 case. Indeed, for DFN202, in the set of 25% most relevant fractures five of the outflowing ones belong to a connected component without inlet fractures; these fractures ($\mathcal{F}_{15}, \mathcal{F}_{18}, \mathcal{F}_{115}, \mathcal{F}_{180}, \mathcal{F}_{187}$), disconnected from any source of flux, are considered (by the LRP algorithm) less relevant than the other outflowing ones that are at the end of a source–sink path. Moreover, fractures $\mathcal{F}_{15}, \mathcal{F}_{18}, \mathcal{F}_{115}, \mathcal{F}_{180}, \mathcal{F}_{187}$ are characterized by a mean relevance score smaller than one (see Table 4).
   The explanation of this behavior is likely to dwell into the higher number of bottleneck fractures characterizing DFN202 with respect to DFN158. Indeed, the more the bottleneck nodes, the more little differences in the relevance scores can bring to inaccurate fracture removals.

As a final analysis to confirm that the process outlined is able to detect a subnetwork of the DFN that can be considered its backbone, we run numerical simulations on all the sub-DFNs previously introduced (we denote them as DFN158$|_{25\%}$, …, DFN202$|_{75\%}$). More precisely, these simulations run using as input parameters the same transmissivity values of the full DFNs, but restricted to the remaining fractures of the sub-DFN.

After running these simulations for the sub-DFNs, we compare the total flux exiting from the sub networks with the one of the corresponding full DFN, observing very similar behaviors (see Fig. 9) that are confirmed by the values reported in Tables 6 and 7. In general we can observe that for DFN158 the sub-DFNs approximate better the behavior of the full-DFN total flux than in the DFN202 case; the reason is attributable to the observations written at item 4. In general (coherently with the physics of the problem) we observe a general decrease of the total exiting flux while pruning the DFN fractures in the graph; however, we observe in both cases (DFN158 and DFN202) a good conservation of the mean total flux and a quite perfect conservation

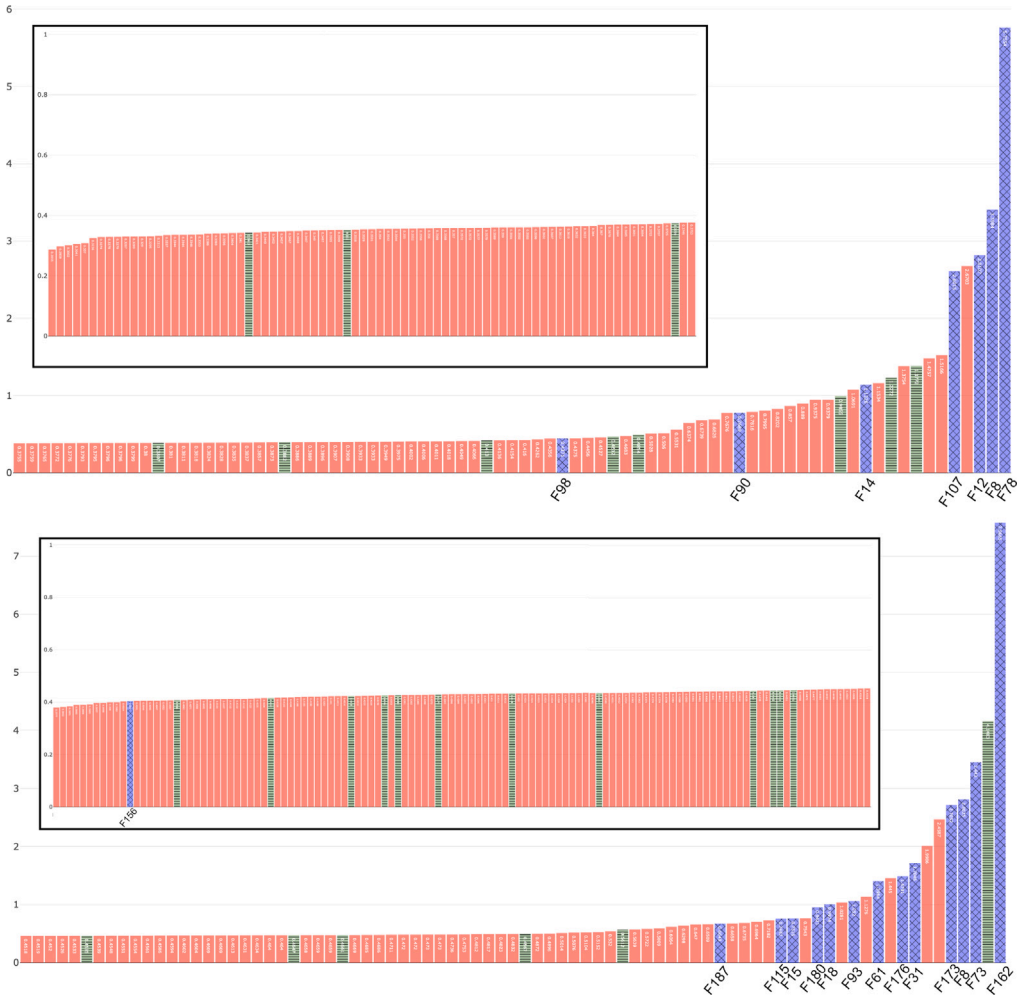

Fig. 5. Mean relevance scores $(\bar{r}_n)_i$ (y axis) and corresponding fractures $\mathcal{F}_i$ (x axis), sorted in ascending order w.r.t. the score values. Top: DFN158; bottom: DFN202. The box in the top-left corner contains the first 60% of the sorted mean relevance scores. Boundary fractures of the DFN have been highlighted with a crossing-lines texture (exiting flux) and a horizontal-line texture (entering flux). Only outflowing fractures are labeled.

Table 4
DFN158. Mean relevance scores and mean flux values (mm^2/s) of outflux boundary fractures (on $\widetilde{D}_{158}$). Columns sorted w.r.t. the mean relevance score (ascending order).

	$\mathcal{F}_8$	$\mathcal{F}_{12}$	$\mathcal{F}_{14}$	$\mathcal{F}_{78}$	$\mathcal{F}_{90}$	$\mathcal{F}_{98}$	$\mathcal{F}_{107}$
$\mathbb{E}_{\widetilde{D}_{158}}[R_i^{(0)}]$	0 .4371	0.7689	1.1326	2.6015	2.8107	3.3984	5.7554
$\mathbb{E}_{\widetilde{D}_{158}}[\varphi_j]$	0.5372	1.8071	7.0173	13.4984	24.6603	26.8003	67.2993

Table 5
DFN202. Mean relevance scores and mean flux values (mm^2/s) of outflux boundary fractures (on $\widetilde{D}_{202}$). Columns sorted w.r.t. the mean relevance score (ascending order).

	$\mathcal{F}_{156}$	$\mathcal{F}_{187}$	$\mathcal{F}_{115}$	$\mathcal{F}_{15}$	$\mathcal{F}_{180}$	$\mathcal{F}_{18}$	$\mathcal{F}_{93}$
$\mathbb{E}_{\widetilde{D}_{202}}[R_i^{(0)}]$	0.4020	0.6648	0.7487	0.7516	0.9421	0.9974	1.0520
$\mathbb{E}_{\widetilde{D}_{202}}[\varphi_j]$	0.2118	1.4390	2.6950	1.3839	1.9175	2.9906	2.4751
	$\mathcal{F}_{61}$	$\mathcal{F}_{176}$	$\mathcal{F}_{31}$	$\mathcal{F}_{173}$	$\mathcal{F}_8$	$\mathcal{F}_{73}$	$\mathcal{F}_{162}$
$\mathbb{E}_{\widetilde{D}_{202}}[R_i^{(0)}]$	1.3964	1.4791	1.7046	2.7052	2.8025	3.4430	7.5665
$\mathbb{E}_{\widetilde{D}_{202}}[\varphi_j]$	5.2791	6.5818	6.5228	11.4715	27.0487	19.9980	56.0594

of the standard deviation of the same quantity. In particular, looking at Tables 6 and 7 we see that removing 75% of the fractures we lose only 14.38% of the flux in DFN158 and 29.01% of the flux in DFN202.

The simulations run on all the sub-DFNs proved that backbones for both DFN158 and DFN202 have been discovered thanks to the NNs $\mathcal{N}^*_{158}$ and $\mathcal{N}^*_{202}$, respectively, since we have identified sub-networks of fractures in the DFNs that approximate the flux behaviors of the full networks with respect to 10 000 samplings of transmissivity vectors $\boldsymbol{\kappa} \in \mathbb{R}^n$ ($n = 158, 202$). Then, we showed that is possible to identify a backbone for a given DFN with a given NN that approximates sufficiently well the function $\boldsymbol{F}$ for the actual exiting fluxes computation.

The backbones built using the method illustrated in this section are characterized by the property of being robust with respect to different choices of transmissivity vectors, preserving (approximately) the total flux behavior; therefore, this backbones theoretically could be able to conserve also the first passage time of particles for transport problems

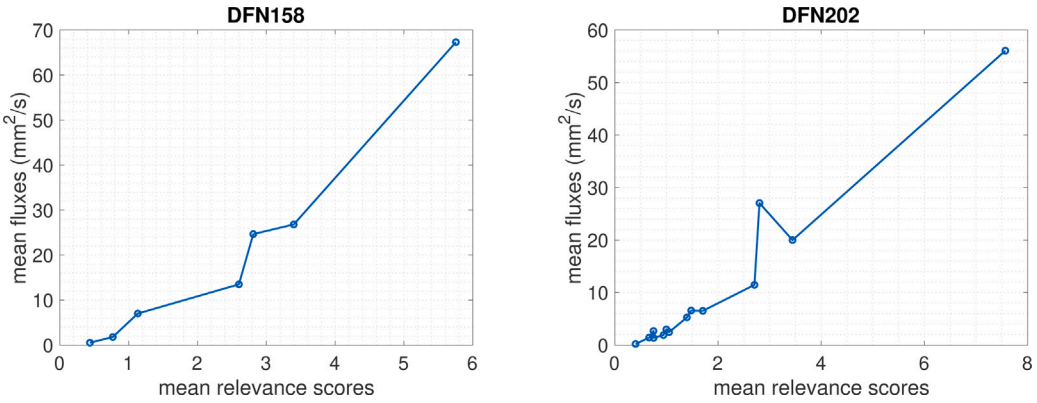

Fig. 6. DFN158 case (left) and DFN202 case (right). Plot of Mean relevance scores versus mean flux values (mm^2/s) of boundary fractures with exiting flux, computed with respect to $\widetilde{D}_n$.

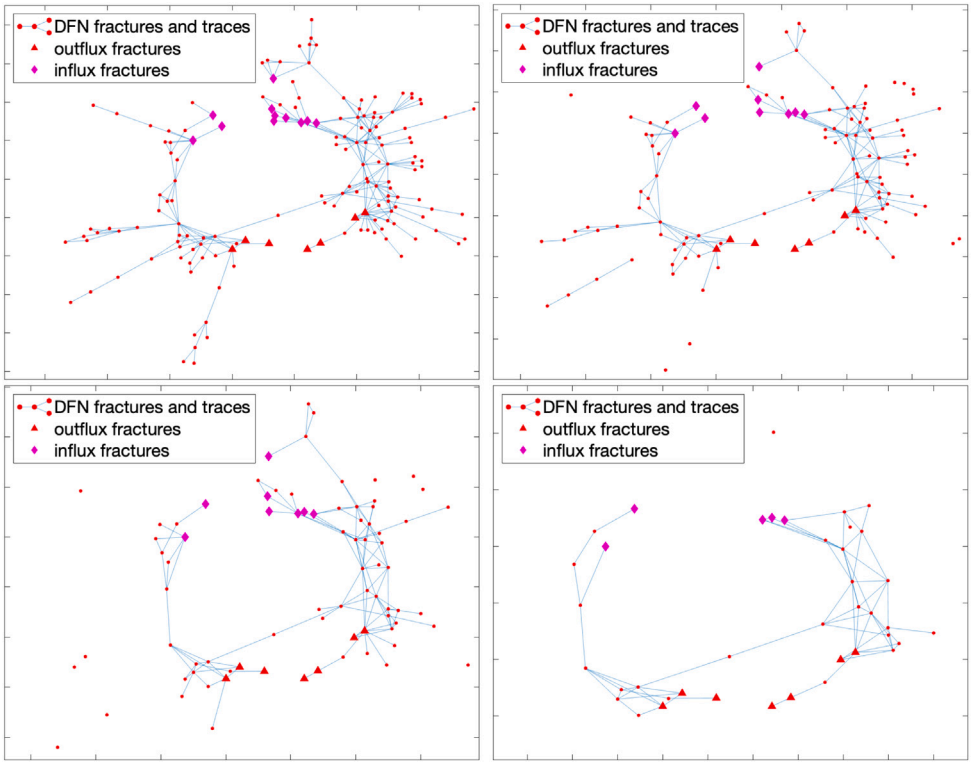

Fig. 7. Graphs representing networks of fractures. From left to right, from top to bottom: DFN158, $DFN158|_{75\%}$, $DFN158|_{50\%}$, $DFN158|_{25\%}$. Triangles/diamonds are fractures with exiting/entering fluxes.

Table 6
DFN158. Mean value and standard deviation of the total flux for the DFN and its sub-DFNs (percentages are computed w.r.t. the DFN values). Last row shows $D_{\mathrm{KL}}/\mathcal{E}$ for the flux distributions of sub-DFNs with respect to the one of the DFN.

	DFN158	$DFN158\|_{75\%}$	$DFN158\|_{50\%}$	$DFN158\|_{25\%}$
$\mathbb{E}[\sum\varphi]$	141.6738	136.8680 (96.61%)	133.5348 (94.26%)	121.2997 (85.62%)
$\sigma[\sum\varphi]$	27.5962	27.5345 (99.78%)	27.3233 (99.01%)	27.0040 (97.85%)
$D_{\mathrm{KL}}/\mathcal{E}$	–	0.0028	0.0077	0.0451

Table 7
DFN202. Mean value and standard deviation of the total flux for the DFN and its sub-DFNs (percentages are computed w.r.t. the DFN values). Last row shows $D_{\mathrm{KL}}/\mathcal{E}$ for the flux distributions of sub-DFNs with respect to the one of the DFN.

	DFN202	$DFN202\|_{75\%}$	$DFN202\|_{50\%}$	$DFN202\|_{25\%}$
$\mathbb{E}[\sum\varphi]$	146.0742	142.9892 (97.89%)	128.2563 (87.80%)	103.6958 (70.99%)
$\sigma[\sum\varphi]$	37.4519	37.2404 (99.44%)	36.3729 (97.12%)	37.4810 (100.08%)
$D_{\mathrm{KL}}/\mathcal{E}$	–	0.0007	0.0217	0.1045

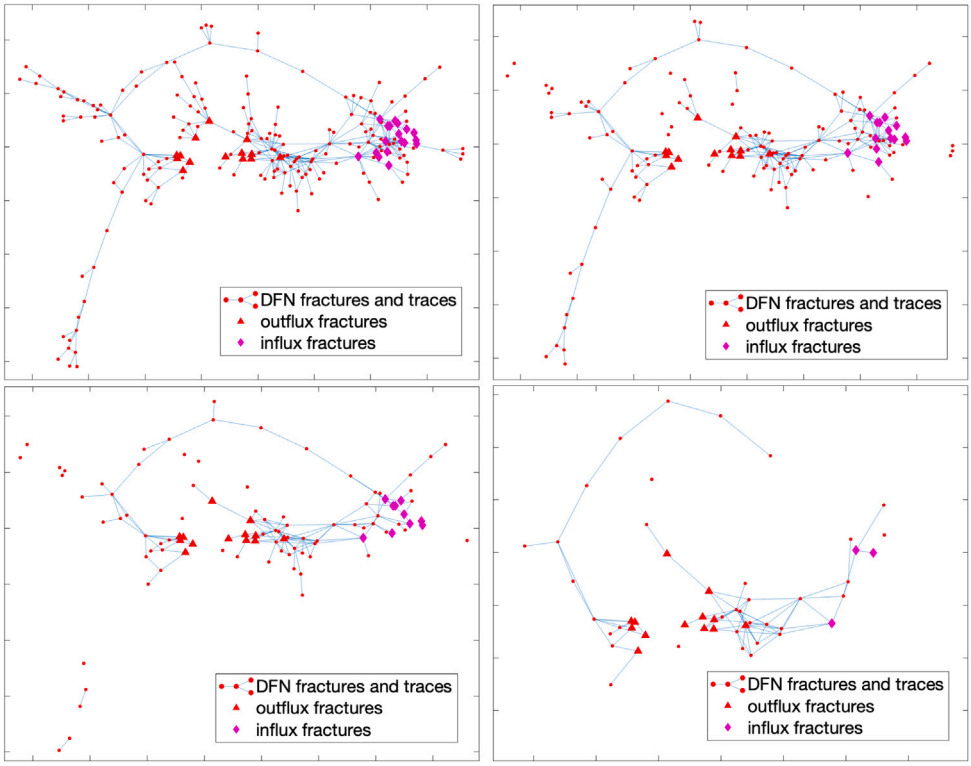

Fig. 8. Graphs representing networks of fractures. From left to right, from top to bottom: DFN202, DFN202$|_{75\%}$, DFN202$|_{50\%}$, DFN202$|_{25\%}$. Triangles/diamonds stars are fractures with exiting/entering fluxes.

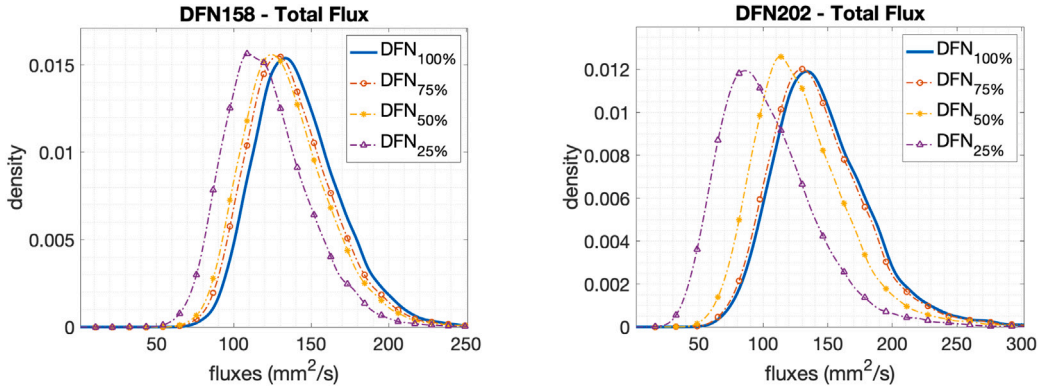

Fig. 9. Total flux comparisons, for both DFN158 (left) and DFN202 (right), with respect to the probability density functions of their sub-DFNs.

(QoI conserved by backbones in [4,5,27,28,30]), since this quantity depends on the fluxes characterizing the DFN. However, further studies should be done to confirm this hypothesis, but they are not part of the purpose of this work.

5.2. Refinement of the direct method: LRP and physically based strategy

In Section 5.1 we observed that, computing the expected relevance scores with the LRP method for $\mathcal{N}^*_{158}$ and $\mathcal{N}^*_{202}$, we are able to identify backbones for DFN158 and DFN202 characterized by a good approximation of the total exiting flux of the corresponding full DFNs. However, sorting the DFN fractures with respect to the scores, we observed that not important fractures (e.g. the ones belonging to dead-ends branches of the network) sometimes have a higher score than other fractures, especially in lower scores cases (see Fig. 10, see item 4, Section 5.1). Therefore, we can refine the backbone identification method described in Section 5.1 taking into account the fractures of the dead-end branches that, for a better evaluation of the abilities of the new LRP-based feature selection method, were intentionally not removed from the network.

The main idea behind the refinement of the method is the following: chosen an arbitrary "percentage step" p, we create iteratively a sequence of sub-DFNs alternating two main steps until a minimum percentage threshold τ of the starting total fractures is reached. The two steps are the following:

- a graph-pruning step based on removing dead-end branches from the graph of the current sub-DFN. Moreover, the sorted sequence of expected relevance scores is "updated", removing from it the same fractures removed from the graph;
- a graph-pruning step based on the "updated" expected relevance scores keeping a specific percentage of fractures (as described in Section 5.1);

More specifically, we can describe the whole refinement procedure with the Algorithm 1.

Table 8
Summary of outcome of Algorithm 1 ($\rho = \tau = 0.25$) showing the number of nodes in G at each step (percentages are w.r.t. the initial number of nodes). Bold values characterize the returned sub-DFNs.

		while-step 1		while-step 2	
	remove dead-ends	$[\rho_1 N_0]$ most rel. fractures	remove dead-ends	$[\rho_2 N_0]$ most rel. fractures	remove dead-ends
DFN158	87 (55.06%)	79 (50%)	**77 (48.73%)**	40 (25.32%)	**39 (24.68%)**
DFN202	108 (53.47%)	101 (50%)	**99 (49.01%)**	51 (25.25%)	**46 (22.77%)**

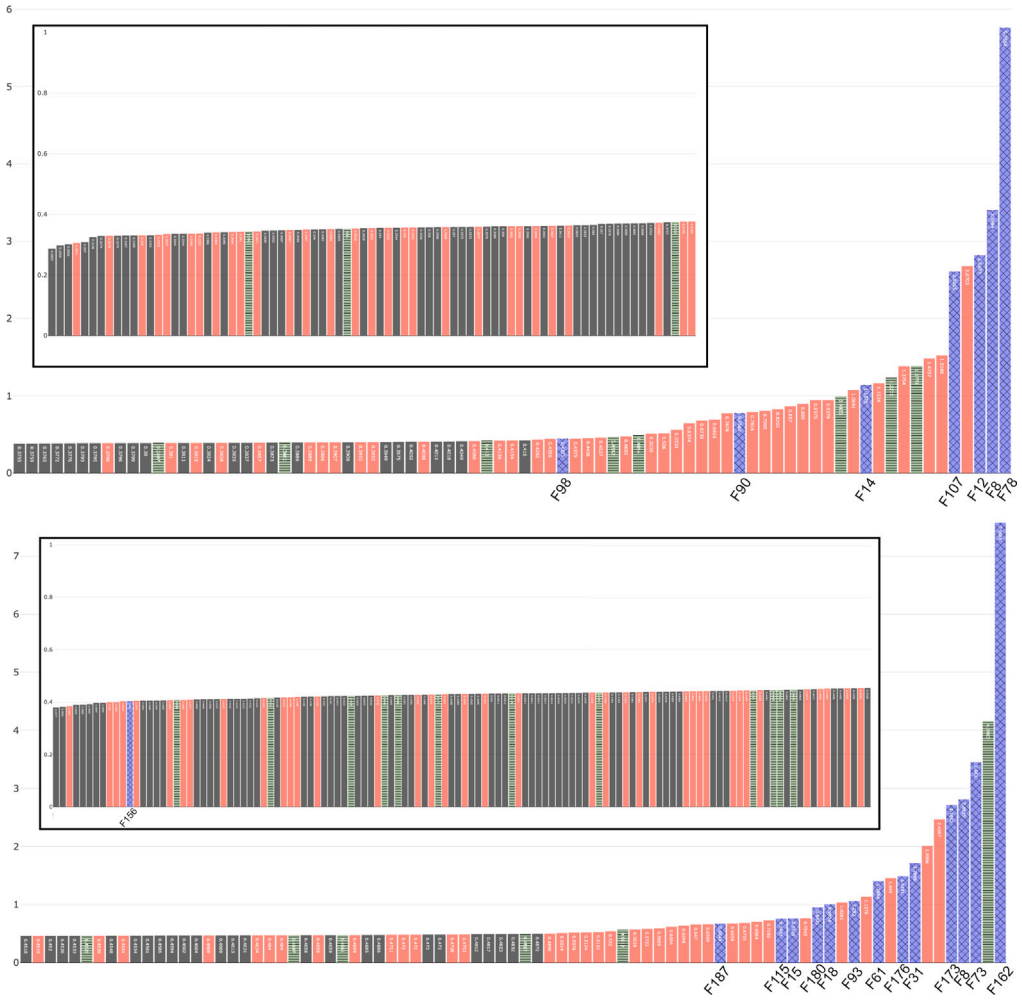

Fig. 10. Mean relevance scores $(\bar{r}_n)_i$ (y axis) and corresponding fractures $\mathcal{F}_i$ (x axis), sorted in ascending order w.r.t. the score values. Top: DFN158; bottom: DFN202. The box in the top-left corner contains the first half of the sorted mean relevance scores. Boundary fractures of the DFN have been highlighted with a crossing-lines texture (exiting flux) and a horizontal-line texture (entering flux). Fractures belonging to dead-end branches of the network are the darkest ones. Only outflowing fractures are labeled.

Algorithm 1 (*Refined Backbone Identification*).

Data: G (graph of the full DFN), $p \in (0, 1)$ (percentage step), L (fracture/node sequence, sorted in increasing order with respect to the relevance score value), $\tau \in (0, 1)$ (minimum percentage threshold).

Outputs: $\mathcal{L}$ (sequence containing the subs-sequences of L and characterizing the sub-DFNs).

Procedure Refined Backbones(G, L, p, τ):

1. $\mathcal{L} \leftarrow$ empty sequence;
2. $N_0 \leftarrow$ number of nodes in G; *(num. of fractures of the full DFN)*
3. $\rho \leftarrow 1$; *(percentage "counter")*
4. $R \leftarrow$ subset of G's nodes s.t. they belong to dead-end branches;
5. $L \leftarrow$ remove all the $v \in R$ from L; *(L still sorted w.r.t. relevance scores)*
6. G $\leftarrow$ sub-graph of G given by nodes in L;
7. $N \leftarrow$ number of nodes in G;
8. **while** $N/N_0 > \tau$ **do:**
9. **while** $\rho \geq N/N_0$ **do:** *(in case $(|R|/N_0) > p$)*
10. $\rho \leftarrow \rho - p$;
11. **end while**
12. **if** $\rho < \tau$ **do** break;
13. $L \leftarrow$ last `round`(ρN_0) elements of L; *L (still sorted w.r.t. relevance scores)*
14. $G \leftarrow$ sub-graph of G given by nodes in L;
15. repeat lines 4–7;
16. add G to $\mathcal{L}$;
17. **end while**

In Algorithm 1, the step corresponding to the removal of the dead-ends, both from the graph and the sequence of relevance scores, is described in lines 4–7; the step that keeps only a particular number of fractures (i.e. $[\rho N_0]$) with higher relevance score (as in Section 5.1)

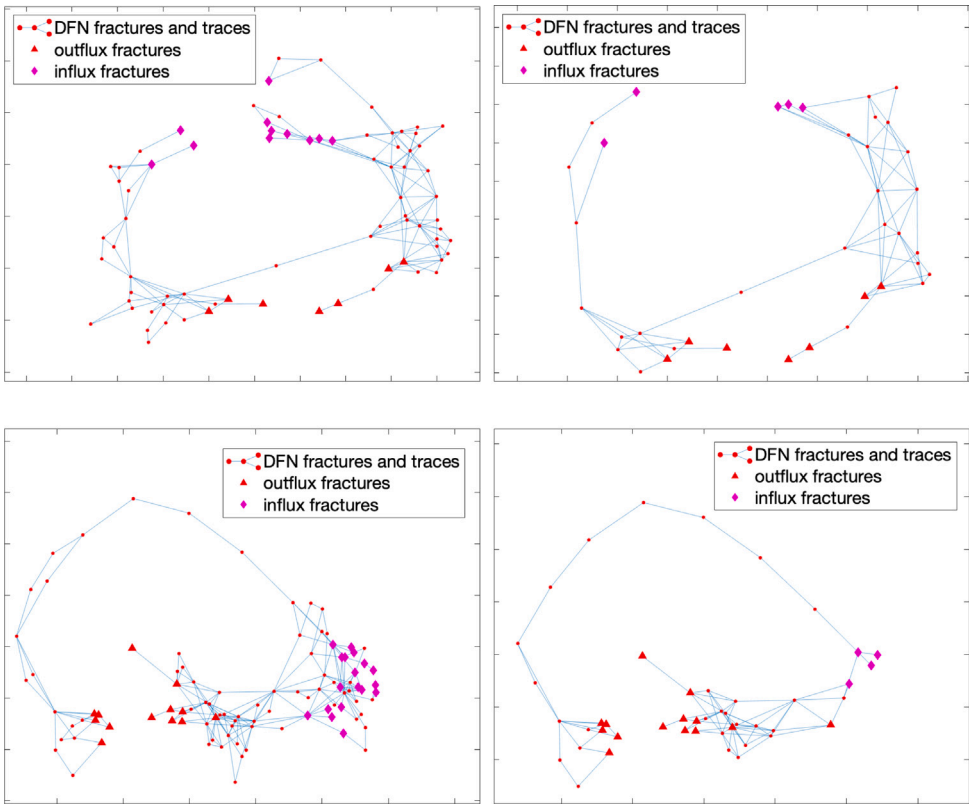

Fig. 11. Graphs representing the network of fractures of: $\text{DFN158}|_{48.73\%}$ (top-left), $\text{DFN158}|_{24.68\%}$ (top-right), $\text{DFN202}|_{49.01\%}$ (bottom-left), $\text{DFN202}|_{22.77\%}$ (bottom-right). Triangles/diamonds are fractures with exiting/entering fluxes.

is described in lines 13–14. In general, the algorithm returns, for each DFN, a sequence of sub-DFNs

$$\left(\text{DFN}|_{(\rho_i-\epsilon_i)100\%}\right)_{i=1,\dots,s}, \tag{19}$$

where s is the number of main iterations executed by the algorithm, ρ_i is the value of ρ at the ith iteration and ϵ_i is the percentage of fracture removed at line List 15 at the ith iteration. The stopping criteria of the algorithm are such that ρ_s is always greater than the threshold τ but $\tau > \rho_s - p$; on the other hand, $(\rho_s - \epsilon_s)$ can be lesser than τ, if the dead-end fractures in $\text{DFN}|_{\rho_s}$ are more than $(\rho_s - \tau)N_0$. In this last case, the stopping criterion of the main while loop is reached. Concluding the description of the algorithm, the role of the inner while loop (line List 9) is only to skip the values of ρ such that ρN_0 is greater than the current number of nodes in G.

One of the most important observation for Algorithm 1 is that such a kind of refinement actually has a negligible computational cost, since the computation of dead-end branches of a graph is not particularly expensive; all the remaining operations are the same ones illustrated in Section 5.1.

Now, we apply the refined method to DFN158 and DFN202. Then, we compare the results with the previous ones. For both DFN158 and DFN202, we run Algorithm 1 with $p = \tau = 0.25$, such that it should return a sequence of sub-DFNs

$$\left(\text{DFN}|_{(0.75-\epsilon_1)100\%}, \text{DFN}|_{(0.50-\epsilon_2)100\%}, \text{DFN}|_{(0.25-\epsilon_3)100\%}\right) \tag{20}$$

comparable with those computed in Section 5.1. However, since both in DFN158 and in DFN202 the number of fractures belonging to dead-end branches is greater than 25% of the total number of fractures, Algorithm 1 does not return the first element of sequence (20). A brief summary of the operations performed by the refined method is described in Table 8.

Looking at the sub-DFNs graphs obtained with the refined method (see Fig. 11), we observe that from all the influx fractures exist a path that ends in an outflux fracture. This simple fact is actually very important; indeed it is an evidence of the efficiency of the refined method introduced, since the original method was not able to preserve a source–sink path for each influx fracture of the sub-DFNs (e.g., we recall the observations made for $\text{DFN202}|_{25\%}$).

We conclude the comparison analyzing the total fluxes exiting from the new sub-DFNs. Reading the values in Tables 9 and 10 we observe that the sub-DFNs obtained with the refined method are characterized by a better flux approximation of the total flux exiting from the full DFN (see also Figs. 12 and 13 for a visual comparison). Therefore, in the end, we can assert that the method characterized by Algorithm 1 is a good refinement of the method proposed in Section 5.1, returning subnetworks of the DFN that can be considered its backbones.

Table 9
DFN158. Mean value and standard deviation of the total flux for the DFN and its sub-DFNs computed with Algorithm 1 (percentages are computed w.r.t. the DFN values). Last row shows $D_{\text{KL}}/\mathcal{E}$ for the flux distributions of sub-DFNs with respect to the one of the DFN.

	DFN158	$\text{DFN158}\|_{48.73\%}$	$\text{DFN158}\|_{24.68\%}$
$\mathbb{E}[\sum\varphi]$	141.6738	138.2734 (97.60%)	124.8932 (88.15%)
$\sigma[\sum\varphi]$	27.5962	27.5544 (99.85%)	26.9265 (97.57%)
$D_{\text{KL}}/\mathcal{E}$	–	0.0014	0.0314

Table 10
DFN202. Mean value and standard deviation of the total flux for the DFN and its sub-DFNs computed with Algorithm 1 (percentages are computed w.r.t. the DFN values). Last row shows $D_{\text{KL}}/\mathcal{E}$ for the flux distributions of sub-DFNs with respect to the one of the DFN.

	DFN202	$\text{DFN202}\|_{49.01\%}$	$\text{DFN202}\|_{22.77\%}$
$\mathbb{E}[\sum\varphi]$	146.0742	142.9952 (97.89%)	112.4792 (77.00%)
$\sigma[\sum\varphi]$	37.4519	37.5844 (100.35%)	38.0651 (101.64%)
$D_{\text{KL}}/\mathcal{E}$	–	0.0008	0.0695

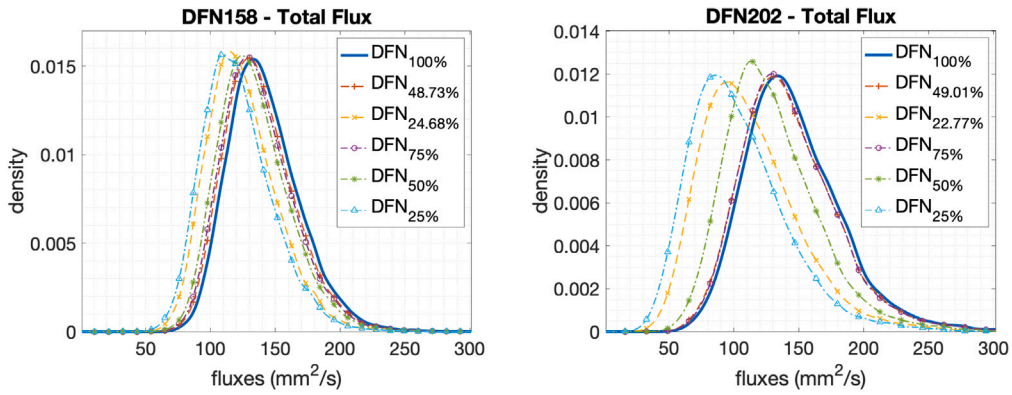

Fig. 12. Total flux comparisons, for both DFN158 (top) and DFN202 (bottom), with respect to the probability density functions of all their sub-DFNs (computed with both two methods described in this work).

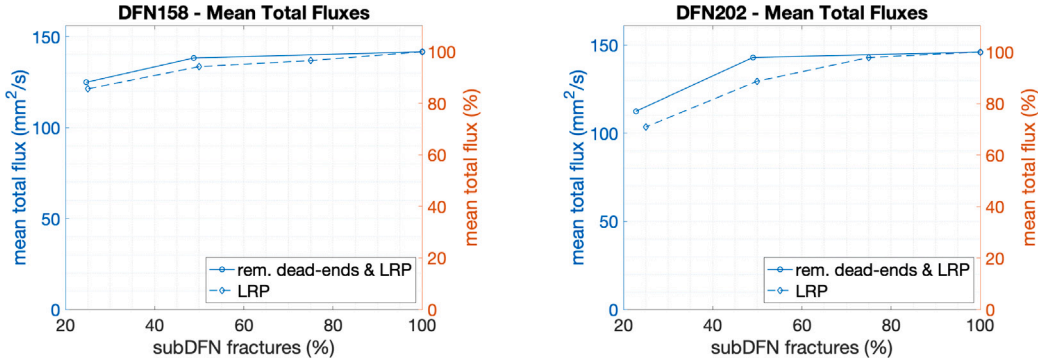

Fig. 13. Mean total flux comparison for both DFN158 (left) and DFN202 (right), with respect to their sub-DFNs. Continuous line characterizes the values corresponding to the sub-DFNs obtained with the refined method. Dotted lines characterizes the values corresponding to the sub-DFNs obtained with the not refined method.

6. Conclusions

In this work, we presented a novel backbone identification method with target preserved QoI the total flux exiting from the DFN. The new method is based on a novel application of the LRP algorithm to a NN trained to predict the fluxes of an arbitrary DFN.

Given two test case DFNs with a total number of fractures equal to 158 and 202, respectively, we trained a set of flux regression multi-task NNs for both of them; then, we selected the two NNs with best regression performance, i.e. the networks $\mathcal{N}^*_{158}$ and $\mathcal{N}^*_{202}$, and we applied the LRP algorithm as a feature selection method, computing the expected relevance of the fractures of both the DFNs.

The simpler method proposed for identifying the backbone consists of selecting an arbitrary percentage of the most relevant fractures, looking only at the expected relevances. This method showed a very interesting characteristic of the NNs: looking both at the graph of the sub-DFNs and at their flux distributions, the backbones obtained proved to be physically consistent and able to preserve well the total flux of the DFN; these facts underline a not obvious NN ability of understanding the implicit and hidden physical relationships between fractures. Indeed, we want to recall that the NNs have been trained looking only at the pairs $(\tilde{\boldsymbol{\kappa}}, \boldsymbol{\varphi})$ of the training sets, having no other information of the problem, both geometrical and hydrogeological.

At the end of this work, we proposed also a refinement of the first method. Since flux regression NNs approximate the numerical simulation results for computing the total flux of the DFNs, the understanding of the real relationships between fractures is good but cannot be perfect, especially for the less relevant fractures. Then, a new iterative method has been proposed; the main idea behind this refined method is based on alternating the removal of dead-end fractures and the removal of the less relevant fractures. The result is an algorithm able to return backbones even better than the ones obtained with the first method, where the connectivity of the fracture network and the preservation of the statistical properties of the total flux is robust.

In conclusion, the results illustrated in this work confirm the possibility of identifying flux-preserving backbones through a novel approach, based on Deep Learning and Layer-wise Relevance Propagation, that can be a new useful instrument for many engineering applications.

CRediT authorship contribution statement

Stefano Berrone: Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Supervision, Validation, Roles/Writing – original draft, Writing – review and editing. **Francesco Della Santa:** Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Project administration, Resources, Software, Validation, Visualization, Roles/Writing – original draft, Writing – review and editing. **Antonio Mastropietro:** Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Project administration, Resources, Software, Validation, Visualization, Roles/Writing – original draft, Writing – review and editing. **Sandra Pieraccini:** Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Supervision, Validation, Roles/Writing – original draft, Writing – review and editing. **Francesco Vaccarino:** Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Supervision, Validation, Roles/Writing – original draft, Writing – review and editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

Research performed in the framework of the Italian MIUR Award "Dipartimento di Eccellenza 2018-2022" granted to the Department of Mathematical Sciences, Politecnico di Torino, CUP: E11G18000350001. F.D. and S.P. also acknowledges support from Italian MIUR PRIN project 201752HKH8_003. A.M. gratefully acknowledges the support from Addfor Industriale. The research leading to these results has also been partially funded by INdAM-GNCS and by the SmartData@PoliTO center for Big Data and Machine Learning technologies.

Appendix A. Details about layer-wise relevance propagation

In this appendix, we propose a novel general and formal definition of the LRP messages introduced in Section 4.1. We define the message $R_{i\leftarrow j}^{(\ell,\ell+1)}$, from unit $u_j \in U^{(\ell+1)}$ to unit $u_i \in U^{(\ell)}$ for the LRP method, as

$$R_{i\leftarrow j}^{(\ell,\ell+1)} := \rho\left(c(x_i^{(\ell)}, w_{ij}^{(\ell+1)}),\ R_j^{(\ell+1)}\right), \tag{A.1}$$

where $x_i^{(\ell)}$ is the output of unit $u_i \in U^{(\ell)}$ and $w_{ij}^{(\ell+1)}$ is the weight of the edge (u_i, u_j) in the FNN. The functions $\rho : \mathbb{R}^2 \to \mathbb{R}$ and $c : \mathbb{R}^2 \to \mathbb{R}$ are arbitrary functions, that we denote as *relevance function* and *contribution function*, respectively.

The relevance function ρ has the role to decide "how much" of the score $R_j^{(\ell+1)}$ has to be propagated backward to unit u_i; the higher the output of ρ (i.e. the message $R_{i\leftarrow j}^{(\ell,\ell+1)}$), the more relevant is u_i with respect to u_j. The partial relevance of u_i obviously depends on the score $R_j^{(\ell+1)}$, which is the relevance of u_j, but also on how much u_i "contributed" to unit u_j in the forward passage. This contribution is measured by the function c, taking into account both $x_i^{(\ell)}$ and $w_{ij}^{(\ell+1)}$. In literature [32,33], the most used choice of ρ is the multiplication, such that the role of c is reduced to the computation of an appropriate factor for rescaling $R_j^{(\ell+1)}$. Moreover, the function c can be different for each message. For example c can be characterized by different parameter values varying the unit u_j considered (see parameters $z_j^{\pm}$ in α-β rule later).

The propagation rule, that the LRP method defines to compute the message $R_{i\leftarrow j}^{(\ell,\ell+1)}$ through ρ and c, is assumed to satisfy the following properties:

1. *Conservation* [33]: the sum of the scores propagated from each layer to the preceding ones remains equal, that is:

$$R^{(0)} = \cdots = R^{(L-1)} = R^{(L)} = R(\boldsymbol{x}), \tag{A.2}$$

where, for each $\ell = 0, \ldots, L$, we define $R^{(\ell)}$ as

$$R^{(\ell)} = \sum_{u_i \in U^{(\ell)}} R_i^{(\ell)}. \tag{A.3}$$

The introduction of the conservation property is important because it highlights that LRP actually compute a decomposition of the outputs (assuming (11)) in terms of the input variables [32, 33].

2. *Coherence*: with "coherence" we intend the general criterion behind the user choices of the functions ρ and c for the propagation rule. Definitions of ρ and c are not formally considered or argued in literature due to the empirical and heuristic nature of both the propagation rules addressed and the problems related to the typical LRP applications.
Therefore, trying to introduce a more detailed formalization, we decide to define as "coherent" a propagation rule such that:
   - exists a measure or signed measure μ_ρ with respect to the Borel σ-algebra on $\mathbb{R}^2$ (here denoted as $\mathcal{B}(\mathbb{R}^2)$) such that $\rho(\xi_1, \xi_2) = \mu_\rho((0, \xi_1) \times (0, \xi_2))$, for each $(\xi_1, \xi_2) \in \mathbb{R}^2$;
   - exists a measure or signed measure μ_c with respect to the σ-algebra $\mathcal{B}(\mathbb{R}^2)$ such that $c(\xi_1, \xi_2) = \mu_c((0, \xi_1) \times (0, \xi_2))$, for each $(\xi_1, \xi_2) \in \mathbb{R}^2$.

   The coherence property, introduced by the authors of this work, has the purpose of providing an outline on how to build a "good" propagation rule for LRP. Indeed, not every pair of arbitrary functions ρ and c return scores that describe correctly the relevance of the inputs on the outputs, even if the conservation property is guaranteed. Defining ρ and c as functions characterized by the measures μ_ρ, μ_c, respectively, we think that the relevance scores obtained with LRP better characterize the relationship between input and outputs in the NN.

Once a propagation rule is defined, recalling the purpose of the LRP method, the total starting score $R^{(L)}$ propagates from the output layer to the input units $u_i \in U^{(0)}$ and values $R_i^{(0)}$ let the user to understand the relevance of the components of the given input $\boldsymbol{x}$ behind the FNN prediction $\widehat{F}(\boldsymbol{x})$.

The propagation rule, defined by the two properties of conservation and coherence is not uniquely identified. Therefore different propagation rules have been proposed in literature [33]. In this work, the rule applied in Section 5 belongs to the class of rules named α-β rule. For the ease of notation we denote as z_{ij} the product

$$z_{ij} = x_i w_{ij}, \tag{A.4}$$

where for simplicity we decide to drop the layer dependencies previously denoted with superscripts like (ℓ) and $(\ell+1)$.

Then, α-β rule defines the quantity

$$z_{ij}^{\pm} := \max(\pm z_{ij}, 0) \tag{A.5}$$

as signed local contribution of u_i with respect to u_j and the quantity

$$z_j^{\pm} := b_j^{\pm} + \sum_{u_i \in U^{(\ell)}} z_{ij}^{\pm}, \tag{A.6}$$

as signed pre-activation of u_j, where b_j is the bias of u_j and $b_j^{\pm} = \max(\pm b_j, 0)$.

Therefore the definition of the message from $u_j \in U^{(\ell+1)}$ to $u_i \in U^{(\ell)}$ in the α-β rule [33, section 5.1] is characterized by

$$R_{i\leftarrow j}^{(\ell,\ell+1)} = \rho\left(c_j(x_i, w_{ij}),\ R_j^{(\ell+1)}\right) = c_j(x_i, w_{ij}) \cdot R_j^{(\ell+1)} = \left(\alpha \frac{z_{ij}^{+}}{z_j^{+}} - \beta \frac{z_{ij}^{-}}{z_j^{-}}\right) R_j^{(\ell+1)}, \tag{A.7}$$

where, for each fixed $\alpha, \beta \in \mathbb{R}^+$, one should have $\alpha - \beta = 1$ in order to satisfy the conservation property. To avoid numerical instability, a small number (e.g. $\epsilon = 10^{-9}$) is added to the denominators of (A.7). We also observe that the α-β rule is characterized by a coherence property, where μ_ρ and μ_{c_j} are such that:

$$\mu_\rho((0, \xi_1) \times (0, \xi_2)) = \xi_1\, \xi_2 \tag{A.8}$$

and

$$\mu_{c_j}((0,\xi_1) \times (0,\xi_2)) = \begin{cases} \frac{\alpha}{z_j^{+}}\, \xi_1\, \xi_2, & \text{if } \xi_1\, \xi_2 \geq 0 \\ -\frac{\beta}{z_j^{-}}\, \xi_1\, \xi_2, & \text{if } \xi_1\, \xi_2 < 0 \end{cases}. \tag{A.9}$$

Specifically, in Section 5 of this work the parameter values are fixed to $\alpha = 1$ and $\beta = 0$. This setting implies that, in the propagation rule, the message is defined as

$$R_{i\leftarrow j}^{(\ell,\ell+1)} = \begin{cases} (x_i w_{ij}/z_j^{+}) R_j^{(\ell+1)}, & \text{if } x_i w_{ij} \geq 0 \\ 0, & \text{if } x_i w_{ij} < 0 \end{cases}. \tag{A.10}$$

A.1. LRP generalization to multitask neural networks

In this subsection of the appendix we generalize the LRP method to FNNs having a multitask architecture of the type described in Section 3.1. In principle, the application of LRP to multitask FNNs can be easily performed, since any multitask FNN can be "translated" into an equivalent non-multitask FNN characterized by some null weights. However, this sort of translation is not advisable for at least two reasons: firstly, it violates the principle of parsimony in terms of coding and computational load; secondly, one hinders the model interpretability by losing its tights with topology of the underlying DFN. It is, therefore, worthwhile generalizing the LRP method to multi-task architectures.

Let us consider a multitask FNN $\mathcal{N}$ with the same architecture as the one introduced in Section 3.1, characterized by n inputs, m branches and outputs, and depth $2d$. Let us introduce the following notation for the scores involving the layers of the branches.

- Let L be such that $L = 2d + 1$. For each $h = 1, \dots, m$, the starting score assigned to the hth output is denoted as

$$R_1^{(L;h)} = \widehat{\varphi}_h = \left(\widehat{\boldsymbol{F}}(\widetilde{\boldsymbol{\kappa}})\right)_h , \tag{A.11}$$

  where $\widehat{\boldsymbol{F}}$ is the function associated to $\mathcal{N}$ and $(\widetilde{\boldsymbol{\kappa}}, \widehat{\boldsymbol{\varphi}})$ is a given input–output pair defined as in Section 3.3. Then, the total starting score $R(\widetilde{\boldsymbol{\kappa}}) = \sum_{h=1}^{m} \widehat{\varphi}_h$ is re-defined as

$$R(\widetilde{\boldsymbol{\kappa}}) = \sum_{h=1}^{m} R_1^{(L;h)} =: R^{(L)} . \tag{A.12}$$

- For each $\ell = d + 1, \dots, 2d$, the scores computed with respect to units of the layer $U_h^{(\ell)}$ (belonging to the hth branch of $\mathcal{N}$) are denoted as $R_{i\leftarrow j}^{(\ell,\ell+1;h)}$. Then, analogously to (13) and (A.3), $R_i^{(\ell;h)}$, $R^{(\ell;h)}$ and $R^{(\ell)}$ are defined, respectively, as

$$R_i^{(\ell;h)} = \sum_{u_j \in U_h^{(\ell+1)}} R_{i\leftarrow j}^{(\ell,\ell+1;h)} , \tag{A.13}$$

$$R^{(\ell;h)} = \sum_{u_i \in U_h^{(\ell)}} R_i^{(\ell;h)} \tag{A.14}$$

  and

$$R^{(\ell)} = \sum_{h=1}^{m} R^{(\ell;h)} . \tag{A.15}$$

  In particular, the above equation is used also for the case $\ell = L = 2d + 1$, even if the result is $R^{(L;h)} = R_1^{(L;h)}$.
- For $\ell = d$ and for each $h = 1, \dots, m$, the score propagated from $u_j \in U_h^{(d+1)}$ to $u_i \in U^{(d)}$ is denoted as $R_{i\leftarrow j}^{(d,d+1;h)}$. Then the total score $R_i^{(d)}$ propagated to u_i is given by

$$R_i^{(d)} = \sum_{h=1}^{m} \sum_{u_j \in U_h^{(d+1)}} R_{i\leftarrow j}^{(d,d+1;h)} \tag{A.16}$$

Generalizing now the α-β rule (A.7) to define the scores $R_{i\leftarrow j}^{(\ell,\ell+1;h)}$, for each $\ell = d, \dots, L-1$ and for each $h = 1, \dots, m$, we observe that the conservation and coherence properties are satisfied for the multitask NN $\mathcal{N}$. Then, for the generality of parameters n, m, and d considered, we can assert that the LRP method characterized by the α-β rule can be applied to multitask NNs characterized by the architecture described in Section 3.1.

Appendix B. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.jocs.2021.101458.

References

[1] P.M. Adler, Fractures and Fracture Networks, Kluwer Academic, Dordrecht, 1999.
[2] G. Cammarata, C. Fidelibus, M. Cravero, G. Barla, The hydro-mechanically coupled response of rock fractures, Rock Mech. Rock Eng. 40 (1) (2007) 41–61.
[3] C. Fidelibus, G. Cammarata, M. Cravero, Hydraulic characterization of fractured rocks, in: M. Abbie, J.S. Bedford (Eds.), Rock Mechanics: New Research, Nova Science Publishers Inc., New York, 2009.
[4] S. Srinivasan, S. Karra, J. Hyman, H. Viswanathan, G. Srinivasan, Model reduction for fractured porous media: A machine learning approach for identifying main flow pathways, Comput. Geosci. (2019) http://dx.doi.org/10.1007/s10596-019-9811-7.
[5] G. Aldrich, J.D. Hyman, S. Karra, C.W. Gable, N. Makedonska, H. Viswanathan, J. Woodring, B. Hamann, Analysis and visualization of discrete fracture networks using a flow topology graph, IEEE Trans. Vis. Comput. Graphics 23 (8) (2017) 1896–1909.
[6] B. Nœtinger, N. Jarrige, A quasi steady state method for solving transient Darcy flow in complex 3D fractured networks, J. Comput. Phys. 231 (1) (2012) 23–38.
[7] B. Nœtinger, A quasi steady state method for solving transient Darcy flow in complex 3D fractured networks accounting for matrix to fracture flow, J. Comput. Phys. 283 (2015) 205–223.
[8] W. Dershowitz, C. Fidelibus, Derivation of equivalent pipe networks analogues for three-dimensional discrete fracture networks by the boundary element method, Water Resour. Res. 35 (1999) 2685–2691.
[9] G. Pichot, J. Erhel, J. de Dreuzy, A mixed hybrid mortar method for solving flow in discrete fracture networks, Appl. Anal. 89 (2010) 1629–1643.
[10] G. Pichot, J. Erhel, J. de Dreuzy, A generalized mixed hybrid mortar method for solving flow in stochastic discrete fracture networks, SIAM J. Sci. Comput. 34 (2012) B86–B105.
[11] J.-R. de Dreuzy, G. Pichot, B. Poirriez, J. Erhel, Synthetic benchmark for modeling flow in 3D fractured media, Comput. Geosci. 50 (2013) 59–71.
[12] G. Pichot, B. Poirriez, J. Erhel, J.-R. de Dreuzy, A mortar BDD method for solving flow in stochastic discrete fracture networks, in: Domain Decomposition Methods in Science and Engineering XXI, in: Lecture Notes in Computational Science and Engineering, Springer, 2014, pp. 99–112.
[13] L. Formaggia, A. Fumagalli, A. Scotti, P. Ruffo, A reduced model for Darcy's problem in networks of fractures, ESAIM Math. Model. Numer. Anal. 48 (2014) 1089–1116.
[14] L. Formaggia, P. Antonietti, P. Panfili, A. Scotti, L. Turconi, M. Verani, A. Cominelli, Optimal techniques to simulate flow in fractured reservoir, in: ECMOR XIV-14th European Conference on the Mathematics of Oil Recovery, 2014.
[15] A. Fumagalli, A. Scotti, A numerical method for two-phase flow in fractured porous media with non-matching grids, Adv. Water Resour. 62 (2013) 454–464.
[16] J. Jaffré, J.E. Roberts, Modeling flow in porous media with fractures; Discrete fracture models with matrix-fracture exchange, Numer. Anal. Appl. 5 (2) (2012) 162–167.
[17] M. Karimi-Fard, L.J. Durlofsky, Unstructured adaptive mesh refinement for flow in heterogeneous porous media, in: ECMOR XIV-14th European Conference on the Mathematics of Oil Recovery, 2014.
[18] S. Berrone, A. Borio, A. D'Auria, Refinement strategies for polygonal meshes applied to adaptive VEM discretization, Finite Elem. Anal. Des. 186 (2021) 103502.
[19] S. Berrone, S. Pieraccini, S. Scialò, A PDE-constrained optimization formulation for discrete fracture network flows, SIAM J. Sci. Comput. 35 (2) (2013) B487–B510.
[20] S. Berrone, S. Pieraccini, S. Scialò, On simulations of discrete fracture network flows with an optimization-based extended finite element method, SIAM J. Sci. Comput. 35 (2) (2013) A908–A935.
[21] S. Berrone, S. Pieraccini, S. Scialò, An optimization approach for large scale simulations of discrete fracture network flows, J. Comput. Phys. 256 (2014) 838–853.
[22] S. Berrone, A. Borio, C. Fidelibus, S. Pieraccini, S. Scialò, F. Vicini, Advanced computation of steady-state fluid flow in discrete fracture-matrix models: FEM–BEM and VEM–VEM fracture-block coupling, GEM Int. J. Geomath. 9 (2) (2018) 377–399.
[23] S. Berrone, S. Pieraccini, S. Scialò, F. Vicini, A parallel solver for large scale DFN flow simulations, SIAM J. Sci. Comput. 37 (3) (2015) C285–C306.
[24] S. Berrone, S. Scialò, F. Vicini, Parallel meshing, discretization, and computation of flow in massive discrete fracture networks, SIAM J. Sci. Comput. 41 (4) (2019) C317–C338.
[25] S. Berrone, S. Pieraccini, S. Scialò, Towards effective flow simulations in realistic discrete fracture networks, J. Comput. Phys. 310 (2016) 181–201.
[26] M. Benedetto, S. Berrone, S. Pieraccini, S. Scialò, The virtual element method for discrete fracture network simulations, Comput. Methods Appl. Mech. Engrg. 280 (2014) 135–156.
[27] J.D. Hyman, A. Hagberg, G. Srinivasan, J. Mohd-Yusof, H. Viswanathan, Predictions of first passage times in sparse discrete fracture networks using graph-based reductions, Phys. Rev. E 96 (1) (2017) 1–10.
[28] J.D. Hyman, A. Hagberg, D. Osthus, S. Srinivasan, H. Viswanathan, G. Srinivasan, Identifying backbones in three-dimensional discrete fracture networks: A bipartite graph-based approach, Multiscale Model. Simul. 16 (4) (2018) 1948–1968.

[29] M. Valera, Z. Guo, P. Kelly, S. Matz, V.A. Cantu, A.G. Percus, J.D. Hyman, G. Srinivasan, H.S. Viswanathan, Machine learning for graph-based representations of three-dimensional discrete fracture networks, Comput. Geosci. 22 (3) (2018) 695–710.

[30] S. Srinivasan, E. Cawi, J. Hyman, D. Osthus, A. Hagberg, H. Viswanathan, G. Srinivasan, Physics-informed machine learning for backbone identification in discrete fracture networks, Comput. Geosci. (Mc) (2020).

[31] S. Srinivasan, J.D. Hyman, D. O'Malley, S. Karra, H.S. Viswanathan, G. Srinivasan, Chapter three - machine learning techniques for fractured media, in: B. Moseley, L. Krischer (Eds.), Machine Learning in Geosciences, in: Advances in Geophysics, vol. 61, Elsevier, 2020, pp. 109–150.

[32] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, W. Samek, On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation, PLoS One 10 (7) (2015) e0130140.

[33] G. Montavon, W. Samek, K.-R. Müller, Methods for interpreting and understanding deep neural networks, Digit. Signal Process. 73 (2018) 1–15.

[34] S. Berrone, F. Della Santa, S. Pieraccini, F. Vaccarino, Machine learning for flux regression in discrete fracture networks, GEM Int. J. Geomath. 12 (1) (2021) 9.

[35] S. Berrone, F. Della Santa, Performance analysis of multi-task deep learning models for flux regression in discrete fracture networks, Geosciences 11 (3) (2021) 131.

[36] F.K. Dosilovic, M. Brcic, N. Hlupic, Explainable artificial intelligence: A survey, in: 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO, IEEE, 2018, pp. 0210–0215.

[37] M. Böhle, F. Eitel, M. Weygandt, K. Ritter, Layer-wise relevance propagation for explaining deep neural network decisions in MRI-based alzheimer's disease classification, Front. Aging. Neurosci. 11 (2019) 194.

[38] X. Chen, J. Zhao, L. Chen, Experimental and numerical investigation of preferential flow in fractured network with clogging process, Math. Probl. Eng. 2014 (2014).

[39] S. Vialle, J.L. Druhan, K. Maher, Multi-phase flow simulation of CO2 leakage through a fractured caprock in response to mitigation strategies, Int. J. Greenh. Gas. Control. 44 (2016) 11–25.

[40] H. Deng, P.H. Stauffer, Z. Dai, Z. Jiao, R.C. Surdam, Simulation of industrial-scale CO2 storage: Multi-scale heterogeneity and its impacts on storage capacity, injectivity and leakage, Int. J. Greenh. Gas. Control. 10 (2012) 397–418.

[41] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016, http://www.deeplearningbook.org.

[42] X. Sanchez-Vila, A. Guadagnini, J. Carrera, Representative hydraulic conductivities in saturated groundwater flow, Rev. Geophys. 44 (3) (2006) 1–46.

[43] Svensk Kärnbränslehantering AB, Data Report for the Safety Assessment, SR-site, Tech. Rep. TR-10-52, SKB, Stockholm, Sweden, 2010.

[44] J.D. Hyman, G. Aldrich, H. Viswanathan, N. Makedonska, S. Karra, 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. (2016).

[45] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: Y. Bengio, Y. LeCun (Eds.), 3rd International Conference on Learning Representations, ICLR2015, San Diego, CA, USA, May 7–9, 2015, Conference Track Proceedings, 2015.

[46] N.M. Nawi, W.H. Atomi, M. Rehman, The effect of data pre-processing on optimized training of artificial neural networks, Procedia Technol. 11 (2013) 32–39, 4th International Conference on Electrical Engineering and Informatics, ICEEI 2013.

[47] A. Shrikumar, P. Greenside, A. Kundaje, Learning important features through propagating activation differences, in: Proceedings of the 34th International Conference on Machine Learning-Volume 70, JMLR.org, 2017, pp. 3145–3153.

[48] M. Sundararajan, A. Taly, Q. Yan, Axiomatic attribution for deep networks, in: Proceedings of the 34th International Conference on Machine Learning-Volume 70, JMLR.org, 2017, pp. 3319–3328.

[49] S.M. Lundberg, S.-I. Lee, A unified approach to interpreting model predictions, in: I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems 30, Curran Associates, Inc., 2017, pp. 4765–4774.

[50] W. Samek, G. Montavon, S. Lapuschkin, C.J. Anders, K.-R. Müller, Explaining deep neural networks and beyond: A review of methods and applications, Proc. IEEE 109 (3) (2021) 247–278.

Stefano Berrone: Full Professor in Numerical Analysis. He got a Master Degree in Aeronautical Engineering and a Ph.D. in Fluid Dynamics. His main interests concern the efficient and reliable solution of partial differential equations in complex domains. In the recent years the use of Machine Learning and A.I. tools started to take a relevant part of his interests, Variational Physically informed Machine Learning and its mathematical properties is one of his research subjects. He is author of more than 50 scientific papers on international journals and referee for the most important journal of Numerical Analysis, Applied Mathematics and Computational Engineering.

Francesco Della Santa: Research fellow at Politecnico di Torino. He got a Master Degree in Mathematics at University of Florence in 2017 and a Ph.D. in Pure and Applied Mathematics at Politecnico di Torino in 2021. His main research topic is data-driven deep learning methods for physically-based simulations. In December 2020 he won the "young researcher presenter" award at the CMWR conference.

Antonio Mastropietro: Ph.D. student in Pure and Applied Mathematics at Politecnico di Torino and Data Science Researcher at Addfor Industriale. He graduated in Mathematical Engineering at Politecnico di Torino. His research focuses on Machine Learning, in particular eXplainable AI, Deep Reinforcement Learning, and Deep Learning applied to computer vision.

Sandra Pieraccini: Associate Professor in Numerical Analysis at Politecnico di Torino. She got a Master Degree in Mathematics and a Ph.D. in Applied Mathematics and Operation Research. Her main scientific interests concern Numerical Optimization and Uncertainty Quantification; recently these interests have been combined by investigations in the field of deep learning. She is author of approximately 50 scientific publications on international journals and conference proceedings. She is referee for several important journal of Numerical Analysis and Applied Mathematics.

Francesco Vaccarino: Associate Professor in Geometry at Politecnico di Torino. He is a cofounder and member of the Scientific Board of the SmartData@PoliTO center. He is currently working on several aspects of topological data analysis and computational topology. He gave talks at the International Congress of Mathematicians ICM 2006, the AMS Summer School in Algebraic Geometry 2005, Netsci 2014, 2015, 2016, JMM AMS MAA 2017, and ATMCS 2018. He is author of several scientific papers. As a scientific journalist, he is regularly publishing on major italian newspapers and magazines.