

NOTICE WARNING CONCERNING COPYRIGHT RESTRICTIONS

The copyright law of the United States [Title 17, United States Code] governs the making of photocopies or other reproductions of copyrighted material. Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the reproduction is not to be used for any purpose other than private study, scholarship, or research. If a user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use," that use may be liable for copyright infringement. This institution reserves the right to refuse to accept a copying order if, in its judgement, fulfillment of the order would involve violation of copyright law. No further reproduction and distribution of this copy is permitted by transmission or any other means.

Rapid #: -17485486

CROSS REF ID: 977525

LENDER: JBE :: Ejournals

BORROWER: WVU :: Downtown Campus Library

TYPE: Article CC:CCG

JOURNAL TITLE: Computers & fluids

USER JOURNAL TITLE: Computers & Fluids

ARTICLE TITLE: A data-driven surrogate to image-based flow simulations in porous media

ARTICLE AUTHOR: TAKBIRIBORUJENI,

VOLUME: 201C

ISSUE:

MONTH:

YEAR: 2020

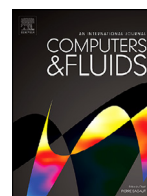
PAGES: 104475-

ISSN: 0045-7930

OCLC #:

Processed by RapidX: 4/22/2021 7:15:09 AM

This material may be protected by copyright law (Title 17 U.S. Code)



A data-driven surrogate to image-based flow simulations in porous media

Ali Takhiri-Borujeni*, Hadi Kazemi, Nasser Nasrabadi

West Virginia University, Morgantown, WV 26506, United States

ARTICLE INFO

Article history:

Received 5 April 2019

Revised 2 December 2019

Accepted 18 February 2020

Available online 22 February 2020

Keywords:

Lattice Boltzmann simulations

Porous media

Machine learning

Data-driven models

Deep learning

Convolutional neural networks

ABSTRACT

The objective for this work is to develop a data-driven surrogate to high-fidelity numerical flow simulations using digital images of porous media. The proposed model can capture the pixel-scale velocity vectors in a large variety of digital porous media created by random two-dimensional (2D) circle packs. To develop the model, images of the 2D media (binary images of solid grains and void spaces) along with their corresponding velocity vectors at the pixel level computed using lattice Boltzmann simulation runs are used to train and to predict the solutions with a high accuracy in much less computational time. The velocity vector predictions made by the surrogate models are used to compute the permeability tensor for samples that have not been used in the training. The results show high accuracy in the prediction of both velocity vectors and permeability tensors. The proposed methodology harnesses the enormous amount of generated data from high-fidelity flow simulations to decode the often under-utilized patterns in simulations and to accurately predict solutions to new cases. The developed model can truly capture the physics of the problem and enhance the prediction capabilities of the simulations at a much lower cost. These predictive models, in essence, do not spatially reduce the order of the problem. They, however, possess the same numerical resolutions as their Lattice Boltzmann simulations equivalents do with the great advantage that their solutions can be achieved by a significant reduction in computational costs (speed and memory).

© 2020 Elsevier Ltd. All rights reserved.

1. Introduction

1.1. Image-based flow simulations

Darcy's principles [7] describe the fluid flow of single-phase fluids in porous media at low Reynolds numbers, which is of significant importance in earth sciences, hydrology, and petroleum engineering. According to the Darcy equation, pressure gradients are linearly proportional to the fluid rate; the proportionality constant is permeability, which is merely a function of pore space topology of porous media irrespective of the fluid type. In numerical flow simulators for porous media, permeability values are obtained based on the data collected from the field and experiments. An accurate quantification of permeability is difficult due to the variations in pore space morphology characteristics. Permeability has been obtained from experiments and also from analytical and empirical expressions that relate permeability to some attributes of the porous media, such as porosity and pore size distribution. The analytical expressions are, however, only approximations for ideal

cases while the empirical expressions have utility only in media similar to scenarios for which they were obtained and thus, are inaccurate when applied to a wide range of other media. Experimental approaches are generally preferred when it is not possible to account for all relevant physics by an equation or model; however, they tend to be time consuming and expensive. Furthermore, they do not capture the effect of pore space morphology characteristics on the flow field and thus, on permeability.

For certain properties, such as permeability, hydraulic tortuosity, and inertial factors of the porous media, high-fidelity numerical simulations using digital images have become a credible alternative, enabled by improvements of imaging techniques, numerical methods, and computing power [38,51]. Appealing aspects of this approach include the ability to probe pore-scale physics at a level not possible with traditional experiments and the ability to perform an endless set of numerical tests without degrading or altering the sample. There are considerations that can limit this digital approach including whether the imaging technique can resolve all relevant characteristic scales in the pore space and whether numerical algorithms can accurately model the physical processes. Higher resolution, however, mandates higher computation power. In high-fidelity numerical simulation

* Corresponding author.

E-mail address: altakhiri@mix.wvu.edu (A. Takhiri-Borujeni).

models, the expensive computational costs, the intensive memory requirements, and the poor scaling performances have traditionally prevented their applications beyond toy or small-scale problems, even using the modern high-performance computing systems.

In image-based pore-scale modeling, the domain is discretized into nodes, voxels, or volume elements, and the resulting grid is used to numerically approximate the relevant partial differential equations for flow, namely computational fluid dynamics (CFD). There is a group of numerical modeling techniques that can utilize the voxel data from X-ray tomography or similar methods as the numerical grid. This gridding approach has become widely used in porous media studies in conjunction with the lattice Boltzmann (LB) simulations and has been proved to be highly effective for simulating fluid flow through porous media [51].

LB simulations have been applied to flow simulations of realistic porous media to compute permeability [9,18,45] with the advantage of being flexible in the specification of variables on the complex boundaries in terms of simple particle bounce back and reflection. This flexibility has opened up the potential in its use for modeling and simulating flow in complex media, such as porous rocks. Challenges for applying LB to real problems include finite-size effects and relaxation time dependence of no-flow boundaries. In image-based simulations, the accuracy of the calculated macroscopic properties depends on the spatial resolution of the rock image [11,28]. However, there is always a trade-off between image resolution and computational power. Furthermore, in all digital samples, there is a resolution threshold, below which certain flow characteristics, such as re-circulation, are not resolved [25].

An extensive research has been performed to study LB modeling of fluid flow in the porous media [1,2,10,27–29,32]. Additionally, Pan et al. [30,31] and Stewart et al. [43] studied the effect of sphere size, spatial discretization, and fluid viscosity (relaxation parameter) on the computed permeability of random-sphere packs and Maier et al. [26] investigated flow of single-phase fluid through a column of glass beads. Takkiri Borujeni [48] studied the applicability of LB simulations in porous media for a wide range of Re and verified the results against experimental and other CFD methods. Their computed permeability tensor and non-Darcy factors were validated experimental flow measurements. They also showed that for $Re < 1$ permeability is not a function of topology of porous media (not a function of pressure gradient and flow velocity), i.e., can be described by the Darcy equation.

The main advantage of pore-scale flow simulations is that explicit influence of each impacting factor can be studied by isolating the effect of other parameters. Attempts of this tabulation of all these impacts have not been manageable yet since such a multi-dimensional parametric study requires comprehensive efforts and time. In this respect, this work aspires to change the status quo and make a transformative leap by combining pore-scale modeling with physics-based ML [34–36,49,50] to develop surrogate models, which can be used to determine the flow fields at very little additional cost. It is also important to note that, using a trained and validated data-driven surrogate model will give us a luxury of performing pore-scale flow simulations, in which computational expenses are not restrictive.

Recently, there have been numerous studies of the application of ML in CFD, most of which are limited to building interpretable reduced-order models (ROMs) [14,15,47,54]. In ROMs, where the number of variables is reduced to simplify the governing equations and the relationships between inputs and outputs, some details are inevitably overlooked. On the other hand, the widespread success of ML-based predictive modeling in other disciplines, such as autonomous cars, suggests a great opportunity to advances in the state-of-the-art by combining conventional CFD simulation techniques with predictive capabilities of data-driven surrogate models to truly capture the physics of the problem and enhance prediction

capabilities of the simulations at a much lower cost. They, however, possess the same numerical resolutions as their CFD equivalents do with the great advantage that their solutions can be achieved by a significant reduction in computational costs (speed and memory). Essentially, the predictive models learn the nature of communications among grid cells and decode the spatial correlations between them (auto- and cross-correlations) in the entire computational domain and can accurately predict solutions to completely new sets of simulation runs, from beginning to end.

Recently, Convolutional Neural Networks (CNNs) with hierarchical feature learning capability has outperformed the state of the art in many computer vision tasks, including image classification [41], segmentation [24], and synthesis [13]. Despite in classification tasks, where the network predicts a single class label for an input image, in many visual tasks, the desired output could be a class label, or a continuous value, assigned to each pixel of the input image [59].

Ciresan et al. [6] predicted the class label of each pixel by training a network in a sliding-window fashion which takes a patch around each pixel. This network, then, is able to localize and also is more robust to overfitting the training data, i.e., generated patches, is much larger than the number of training images. However, this framework is quite slow due to the separate processing of each patch, which results in a lot of redundancy on overlapping patches. Moreover, such networks should deal with the trade-off between the localization and context. Large patches need many pooling layers that can reduce the localization performance, while small patches only incorporate little context information in the final decision. More recent studies [24,39] proposed to fuse the fine to coarse features from multiple layers in different depth. This enables the network to achieve an accurate localization while having a large receptive field (context) at the same time. In the work performed by Ronneberger et al. [37], the authors introduced U-Net which employed contracting path in its Auto-Encoder architecture to capture context and enable precise localization. Furthermore, training a very deep neural network is quite a challenging task. More specifically, it is hard for a deep network to find an optimal solution compared to shallower counterparts. One of the main issues in training a deep network is the vanishing gradient problem, making it difficult to tune the parameters of the early layers in the network [12]. In the past couple of years, multiple training strategies have been proposed to train a deep neural network effectively, including deep supervision in hidden layers [23], initialization scheme [12], and batch normalization [17]. He et al. [16] introduced residual connections in which they employ additive merging of signals to improve the training speed, and gradient flow through the networks.

For clarification, the terminology used in the remainder of the paper is the following. The term *input* is used to denote the binary (zeros and ones) images of porous media, where 0 denotes the void spaces and 1 denotes the solid grains. The term *pixel* and *numerical grids* are used interchangeably due to the fact that the numerical method, LB, use image pixels as the numerical grid. The term *output* refers to the velocity vectors computed at each pixel of each input using LB simulations.

1.2. Application of data-driven modeling in engineering problems

Applications of ML have gained lots of popularity in the past few years throughout various industries. The application of ML in CFD has gained considerable interest recently, mostly to build ROMs. However, in such applications of ML in CFD, it is inevitable to overlook some details. On the other hand, predictive ML techniques suggest a greater opportunity, when the conventional CFD simulation techniques are combined with predictive capabilities of data-driven models. Such approaches can truly capture the physics

of the problem and enhance the prediction capabilities of the simulations at a much lower cost.

Unlike the automotive industry, the application of Artificial Intelligence (AI) in CFD has been limited to interpretable models from data [21,40,53], and predictive models are yet to be employed. The widespread success of predictive modeling in complex problems suggests a great opportunity to advances in the state-of-the-art by combining conventional CFD simulation techniques with ML predictive modeling to truly capture the physics of the problem and enhance prediction capabilities of the simulations at a much lower cost. This can be achieved by developing physically interpretable spatio-temporal simulations of complex CFD problems and introducing a significant reduction in computational cost (speed and memory).

2. Lattice Boltzmann Method

The Boltzmann equation is

$$\frac{\partial f_\alpha}{\partial t} + \mathbf{e}_\alpha \cdot \nabla f_\alpha = \Omega_\alpha, \quad (1)$$

where $f_\alpha(\mathbf{x}, t)$ is the fraction of fluid particles that have traveled in the α -direction in the phase space directions, $\mathbf{e}_\alpha$ is the particle velocity in the α -direction, and Ω_α is the collision operator [3]. The LB simulation method is a discrete form of the continuous Boltzmann equation in which time and space are discretized with velocity limited to a finite set of admissible directions in which the particles can travel [5,46]. The basic LB algorithm consists of two steps; particle streaming and collision. In the streaming step, the transfer of the particles between nodes along a particular velocity direction occurs. In the collisions step, momentum exchange between the particles takes place due to collision with each other at a particular node. Bhatnagar-Gross-Krook (BGK) [33,44] and Multiple Relaxation Time (MRT) [8] models have been proposed to approximate the collision term. In the BGK approximation, particle distribution evolves due to collision tending toward an equilibrium distribution function which is defined by the macroscopic velocity at that particular point. The collision term in the BGK model is given by

$$\Omega_\alpha = \frac{f_\alpha(\mathbf{x}, t) - f_\alpha^{eq}(\mathbf{x}, t)}{\tau}, \quad (2)$$

in which τ is the relaxation time that measures the rate at which the distribution functions tend towards equilibrium and also is a tuning parameter that controls the fluid kinematic viscosity,

$$\nu = \frac{2\tau - 1}{6}, \quad (3)$$

and f_α^{eq} are the equilibrium distributions calculated in terms of $\mathbf{u}^{eq}$ that incorporates external forces, F_b , [46]

$$f_\alpha^{eq} = w_\alpha \rho \left[1 + \frac{(\mathbf{e}_\alpha \cdot \mathbf{u}^{eq})}{c_s^2} + \frac{(\mathbf{e}_\alpha \cdot \mathbf{u}^{eq})^2}{2c_s^4} - \frac{(\mathbf{u}^{eq})^2}{2c_s^2} \right], \quad (4)$$

where,

$$\mathbf{u}^{eq} = \mathbf{u} + \left(\frac{\tau}{\rho} \right) F_b,$$

$$w_\alpha = \begin{cases} 4/9, & \alpha = 0 \\ 1/9, & \alpha = 1, 2, 3, 4 \\ 1/36 & \alpha = 5, 6, 7, 8 \end{cases}.$$

where w_α are weight factors specific to different directions, $c_s = 1/\sqrt{3}$ is the speed of sound in the fluid in the lattice units [5], $\mathbf{u}^{eq}$ and $\mathbf{u}$ are equilibrium flow velocity and fluid velocity, respectively, and ρ is the fluid density. In LB simulations, parameterized values of the lattice constants and fluid in lattice units are used in simulation while correspondence between the physical systems being

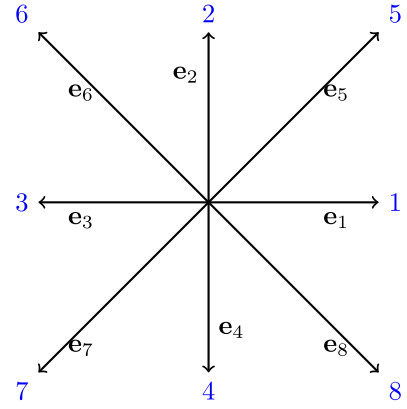


Fig. 1. Depiction of the D_2Q_9 model.

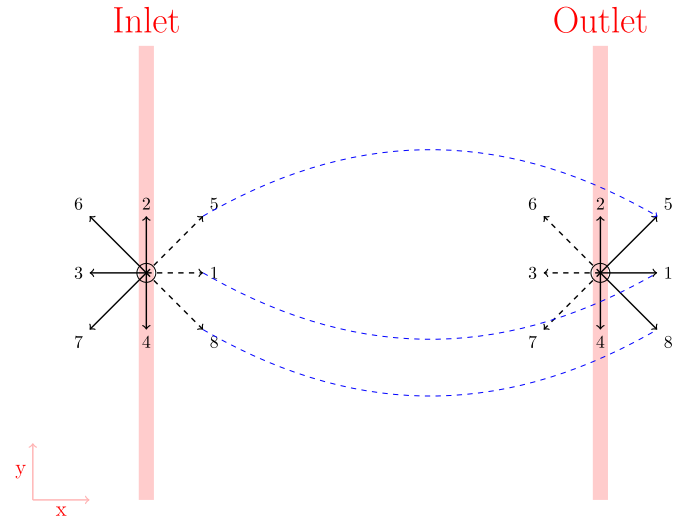


Fig. 2. Depiction of the periodic boundary conditions.

simulated and the parameterized simulation is achieved through Reynold's number [4].

The macroscopic quantities, such as density and momentum density, are defined as velocity moments of the distribution function as follows,

$$\begin{aligned} \rho &= \sum_{\alpha} f_{\alpha}, \\ \rho \mathbf{u} &= \sum_{\alpha} f_{\alpha} \mathbf{e}_{\alpha}. \end{aligned} \quad (5)$$

In this work, the D_2Q_9 model (two dimensions and nine directions of fluid movement) is used (Fig. 1). Velocity vectors for this model are described below,

$$\begin{aligned} \mathbf{e}_{\alpha} &= \left[\cos \left(\frac{2\pi(\alpha-1)}{8} \right), \sin \left(\frac{2\pi(\alpha-1)}{8} \right) \right], \\ \alpha &= 1, 2, \dots, 8. \end{aligned} \quad (6)$$

Periodic boundary conditions are applied to all the external faces in x- and y-direction (Fig. 2). For nodes on a boundary, neighboring points are on the opposite boundary. For instance, components of the particle distribution functions coming out of one boundary (directions 1, 5, and 8 at the outlet on the right side of Fig. 2) enter into the opposite boundary (at the inlet on the left side of Fig. 2).

The bounce-back boundary scheme is used to implement the no-flow boundary conditions at the void-solid interfaces [5,48]. In this scheme, the distribution function traveling from a fluid node

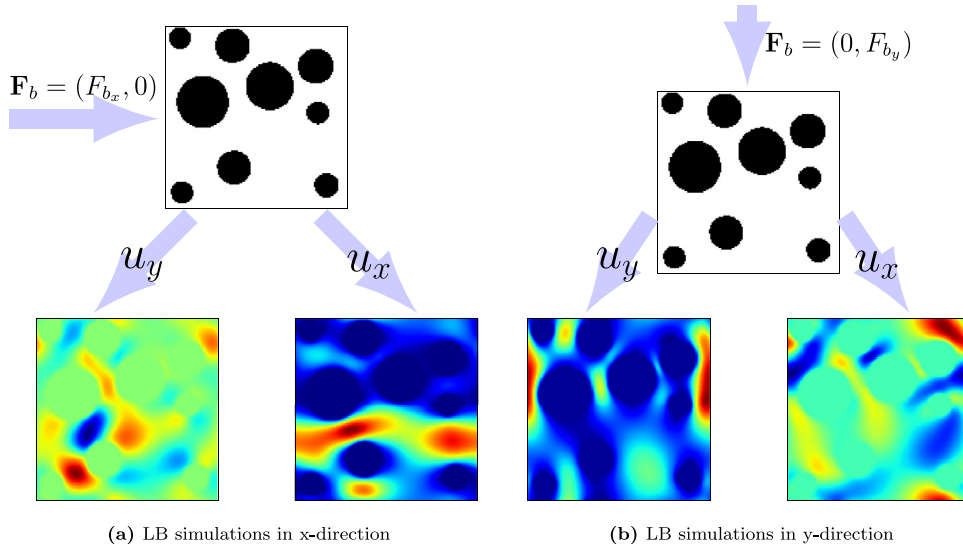


Fig. 3. Schematic of flow simulations in (a) x-direction and (b) y-direction.

to a neighboring solid node is bounced back along the same link. This ensures a zero velocity vector on the bounce back node as obtained in real fluid flows. LB simulations in this study are performed using the Parallel Lattice Boltzmann Solver (PALABOS) [22].

Permeability is calculated from the Darcy equation,

$$\langle \mathbf{u} \rangle = \frac{K}{\mu} \mathbf{F}_b, \quad (7)$$

or,

$$\begin{pmatrix} \langle u_x \rangle \\ \langle u_y \rangle \end{pmatrix} = \frac{1}{\mu} \begin{pmatrix} K_{xx} & K_{xy} \\ K_{xy} & K_{yy} \end{pmatrix} \begin{pmatrix} F_{bx} \\ F_{by} \end{pmatrix}$$

where,

$$\langle \mathbf{u} \rangle = \frac{1}{V} \int_{V_p} \mathbf{u} dV$$

in which K is the permeability tensor of the porous medium, $\langle \mathbf{u} \rangle$ is the average velocity vector of the fluid in the domain, μ is the viscosity of the fluid, and F_b is the body force exerted. The relationship between the intrinsic average velocity ($\langle \mathbf{u} \rangle^p = \frac{1}{V_p} \int_{V_p} \mathbf{u} dV$) and average velocity is $\langle \mathbf{u} \rangle = \epsilon \langle \mathbf{u} \rangle^p$, where $\epsilon = V_p/V$ is the porosity of the media and p denotes the pore [55]. Velocity values in each grid are computed in all directions using the LB simulations to determine the permeability tensor.

All the simulations are tested to verify that they have reached steady-state conditions, where the kinetic energy of the system becomes constant. A body force approach, which is an alternative to specifying pressure values at the inlet and outlet of the domain, is used [48]. A body force of $1.0e^{-7}$ in lattice units [5] is used in x-direction ($\mathbf{F}_b = (F_{bx}, 0)$) and y-directions ($\mathbf{F}_b = (0, F_{by})$) for simulating fluid flow in x- and y-directions, respectively (Fig. 3).

The relaxation time is set to be 1.0 for all the simulations, resulting in a kinematic viscosity of 1/6 according to Eq. 3. Having reasonably large pore sizes in the input images (more than 10 grids in general), calculation of the permeability is done without substantial numerical errors (finite-size errors and relaxation-time dependence of the no-flow boundaries) [51,52]. Reynolds number resulting from this choice of the body force ranges from 0.002 to 0.017 with an average of 0.007, which shows that the flow is in the Darcy flow regime ($Re < 1$) [48].

3. Methodology

In LB simulations, the solutions, $\mathbf{u}(\mathbf{s}, x(\mathbf{s}))$, are obtained at spatial locations $\mathbf{s}$, where the pixels of the binary input image $x(\mathbf{s})$ and $S = \{s_1, \dots, s_{n_s}\}$ are the index set for the spatial grid locations, $\mathbf{s} \in S \subset \mathbb{R}^{d_s}$ ($d_s = 1, 2, 3$) are the spatial locations. The simulations can be considered as a mapping of $x \in \{0, 1\} \subset \mathbb{R}^{d_x n_s}$ to its corresponding solution $\mathbf{u} \in \mathcal{U} \subset \mathbb{R}^{d_u n_s}$,

$$\eta : \{0, 1\} \rightarrow \mathcal{U}, \quad (8)$$

where $\mathbf{u} = \eta(x)$. The purpose for building the surrogate model is to develop a new mapping function, $\hat{\mathbf{u}} = \mathcal{F}(x, \theta)$, to be trained using a limited number of simulation data, $\mathcal{D} = \{x^i, \mathbf{u}^i\}_{i=1}^N$ with θ as the model parameters and N as the number of simulation runs in the training, to approximate the predictions made by η mapping.

LB simulations are performed over computer-generated 2D porous media consisting of random circle packs, which provide a number of advantages for testing pore-scale modeling algorithms. The most intuitive advantage is the ability to fully control the pore structure. Another advantage related to image-based modeling is that the geometric-based data, e.g., locations and sizes of solid grains in a random packing can be converted to voxel data at any desired image resolution without segmentation error. Computer-generated packings have been widely used to simulate granular materials. In some cases, unconsolidated sphere packs have been modified using procedures that mimic diagenetic processes, thus producing consolidated materials [2,19,58].

We generated two-dimensional random circle pack images of size 128×128 pixels ($n_s = H \times W$, where $H = W = 128$), consisting of 5 to 10 grains (circles) with 15 to 36 pixels diameter with random positions. A total of 2000 images are generated for LB simulation runs to determine the permeability.

4. Deep convolutional neural network

Neural networks and specially CNNs, are known for being a powerful tool with the ability to process high dimensional data and vast data sets. The universal approximation theorem indicates that NNs can approximate any arbitrary functions on compact subspaces. NNs comprise a set of vector-valued functions known as layers of neurons. Each layer learns a linear transformation of the input vector, $x(\mathbf{s})$, through its matrix of weights, θ , and vector of biases, b . A non-linear activation function, $\mathcal{F}$, is then applied to the

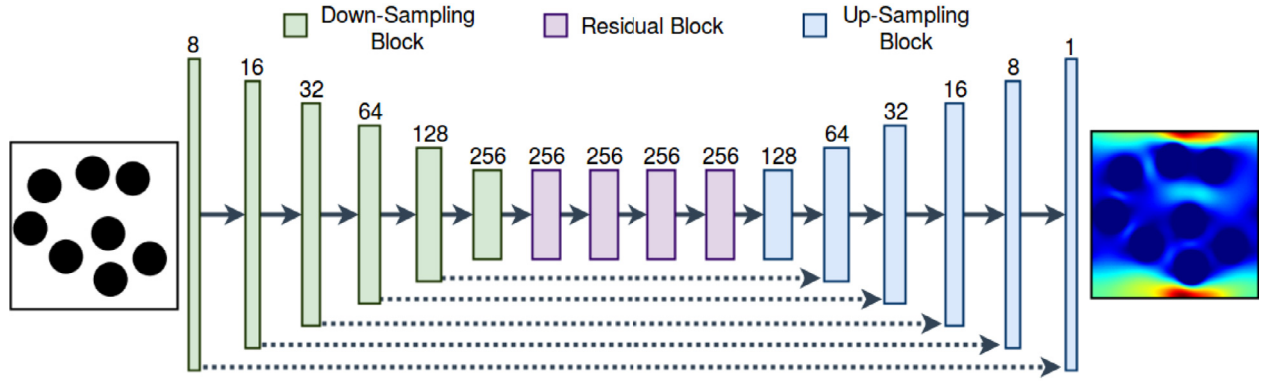


Fig. 4. U-ResNet architecture.

result and yields the velocity vector,

$$\mathbf{u} = \mathcal{F}(\theta\mathbf{x} + b). \quad (9)$$

The rectified linear unit (ReLU) activation function was used for all the layers but the last layer which is tanh to limit the output between -1 and 1. The weights and biases of the network are trained by minimizing an error metric between the predicted output by the network and the corresponding ground truth in the training set.

CNNs, on the other hand, can be considered as a locally constrained form of neural networks, inspired by the structure of the visual cortex. They are highly employed for data like images which have hierarchical local structure. Their superiority in such data comes from eliminating the redundant parameters in the neural networks model. Each convolutional layer comprises a certain number of filters (aka kernels), which are locally connected neurons with fewer sizes than input. Then the filters can be convolved with the input to produce output. Since the weight matrices are shared across the whole input, the number of parameters in a convolutional model is significantly reduced compared to a simple feed-forward neural network. Generally, several filters are employed in a single layer, which form the depth of the output, where each filter extract different types of feature from their inputs. Consequently, the output of a convolutional layer is a 3D tensor (aka feature maps). Convolutional layers are able to distill structure and improve the robustness of the neural networks.

In order to achieve an accurate and efficient model, we employ a deep CNN (DCNN) based on contracting paths and residual blocks. CNNs have been proven successful in geometry representation learning and per-pixel prediction in images. The other motivation for adopting CNNs is its memory efficiency. Memory requirement is a bottleneck to build whole velocity field surrogate models for large geometry shapes. The sparse connectivity and weight-sharing property of CNNs reduce the GPU memory cost greatly.

Since the network consists of only convolutional layers, it can take any arbitrary-sized image as input and generate an output of a similar size. For down-sampling, we use convolutional layers with increased stride instead of pooling layers. After a series of successive strided convolution, the spatial size of feature maps becomes much smaller than that of the input image. To increase the computational capacity of the network, the generated feature maps by the last strided convolution is followed by multiple residual blocks before upscaling to the same size as the input image. The residual connections improve the gradients flow at the training time. Finally, to rescale the feature maps to the size of the input image, we exploit Nearest Neighbor (NN) up-sampling followed by a convolutional layer, instead of deconvolutional layers [57] to prevent checker-board artifacts. Generally, as we go deeper into a DCNN, the size of receptive field increases, which means the

learned feature maps represent more abstract and global contextual features. However, the information about the exact local structure of the image may be lost. On the other hand, the feature maps in early layers, which have smaller the receptive fields, preserve the local structure information. This information is critical for effective velocity field predictions. Consequently, to preserve the local structure information, high resolution features from the contracting path (down-sampling) are combined with the output of the NN up-sampling layer. Then, the subsequent convolution learns to produce a more precise output based on this information. Exploiting the learned discriminative features by the proposed DCNN, we can produce an accurate prediction of velocity fields.

4.1. Architecture set-up

To infer the velocity maps, we use a CNN architecture, whose input is the 2D images of the porous media geometry, where solid circles represent the solid grains. The CNN model is parameterized by its weights and biases. Fig. 4 shows the architecture of the proposed network. It consists of 6 strided convolutions which reduce the size of input by a factor of 64, followed by four residual blocks. At each layer, features are generated from local interactions of inputs and these local interactions combine to capture higher-level global behavior in the deeper layers of the network. However, a single resolution network would have limited context which limits the network's ability to model long-range dynamics resulting from the periodic boundary conditions chosen for this problem. As such, we add multi-resolution features to enable modeling long range physical phenomenon by downsampling the first hidden layer six times, processing resolutions in parallel then upsampling the resultant low-resolution features before accumulating them.

Employing strided convolutions can significantly reduce the amount of computation that has to be done by the network in the subsequent layers. It compresses multiple 3×3 convolution into one 7×7 convolution, to make sure that it has exactly the same receptive field. Each residual block consists of two 3×3 convolutional layers. Finally, In order to obtain the final prediction map, we add six subsequent up-sampling blocks on top of the residual blocks. Neural nets typically use multiple layers of deconvolution when creating images, iteratively building a larger image out of a series of lower resolution descriptions. While it is possible for these stacked deconvolutions to cancel out artifacts, they often compound, creating artifacts on a variety of scales. To avoid these artifacts, we separate out upsampling to a higher resolution from convolution to compute features, i.e., we resize the image (using nearest-neighbor (NN) interpolation) and then apply a convolutional layer. The input to each up-sampling block is the feature maps of the previous layer concatenated in depth with those of the contracting down-sampling path. As mentioned earlier each

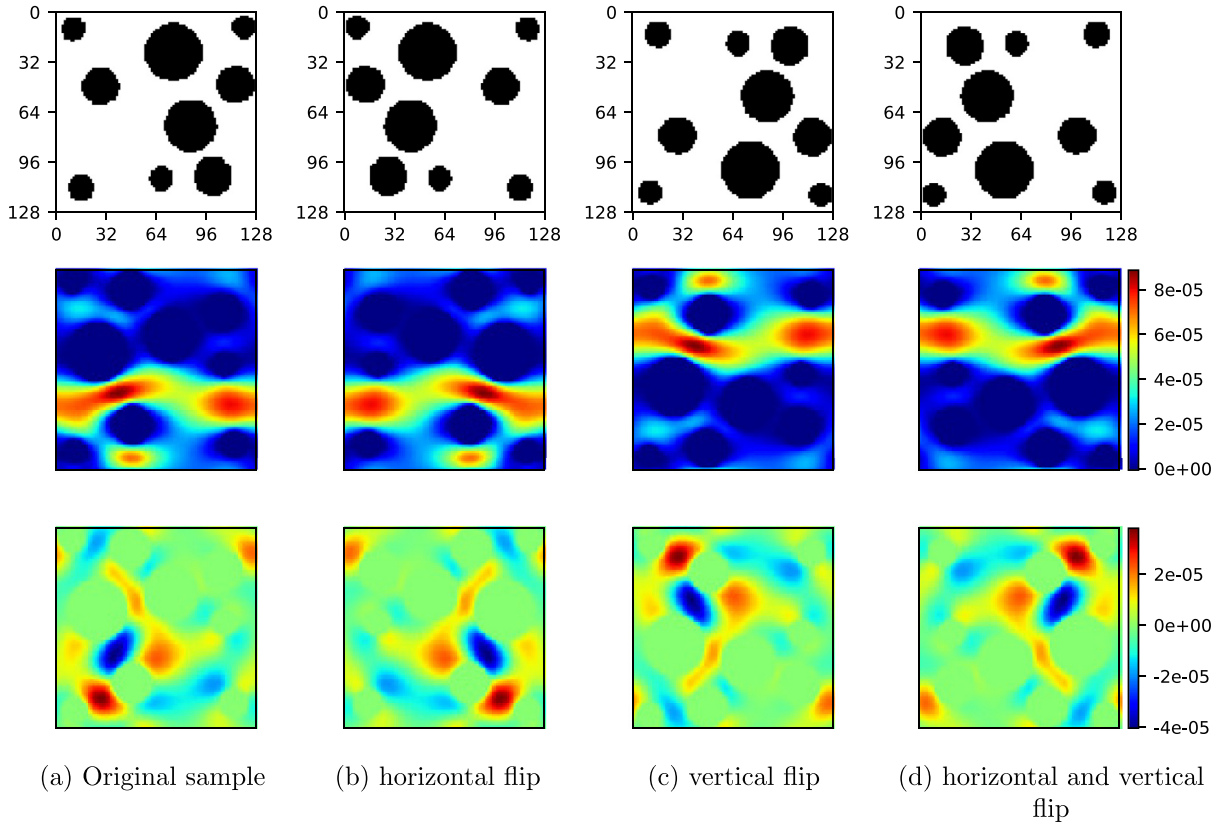


Fig. 5. Sample augmentation. Original samples in (a) are flipped (b) horizontally, (c) vertically, and (d) horizontally and vertically.

up-sampling block comprises successive NN-upsampling and 3x3 convolution with unit stride. Note that all the convolutions are followed by a Batch Normalization and rectified linear unit (ReLU) activation function. Since our input (geometry input image) is the simulated velocity fields we employ reflection 1x1 padding for all the convolutions. Note that since our network is fully-convolutional, the size of the domain can be modified at inference time.

Usually, a deep learning model learns the mapping from an input x to an output u . When the network depth increases, its accuracy starts getting saturated and then degrades very quickly. Unexpectedly, overfitting is not the reason for such degradation, and adding more layers to a suitably deep model leads to higher training error. In fact, this problem appears as a result of gradient vanishing. One solution to this problem was proposed by Zagoruyko and Komodakis [56] to use Resnet blocks, which connect the output of one layer with the input of an earlier layer. Instead of learning a direct mapping, the residual function $R(i)$ learns the difference between the applied mapping to i and the original input, i . Residual blocks are basically a special case of highway networks [42] without any gates in their skip connections. Essentially, residual blocks allow the flow of memory (or information) from initial layers to the last layers. Despite the absence of gates in their skip connections, residual networks perform as good as any other highway network in practice. The idea of skipping connections between the layers was first introduced in Highway Networks. Highway networks had skip connections with gates that controlled how much information is passed through them and these gates can be trained to open selectively.

To train the network, we first normalize the velocity maps. To train the surrogate model, sample augmentation (increasing the number of data samples by performing transformations on the existing ones) was performed by flipping the maps horizontally

and/or vertically (Fig. 5). As can be seen, any single input-output data sample pair can be transformed to generate three new samples, which restricts the need for using new input-output sample pairs.

We have considered $\mathcal{L}_1$ regularized mean absolute error (MAE) training loss function. The input and the corresponding velocity vector maps are used to update the parameters of the network minimizing the L_1 norm error,

$$L_1(\hat{\mathbf{u}}(\mathbf{x}, \theta), \mathbf{u}) = \frac{1}{n} \sum_{i=1}^n \|\hat{\mathbf{u}}_i - \mathbf{u}_i\|_1 + \lambda \Omega(\theta), \quad (10)$$

where $\Omega(\theta) = \|\theta\|_1$ for $\mathcal{L}_1$ regularization, n is the number of samples in the training data in $\mathcal{N}$ simulation runs, and λ is the regularization strength. The $\mathcal{L}_1$ regularization has the intriguing property that it leads the weight vectors to become sparse during optimization (i.e. very close to exactly zero). In other words, neurons with $\mathcal{L}_1$ regularization end up using only a sparse subset of their most important inputs and become nearly invariant to the noisy inputs. Adam optimization technique [20] is used with a learning rate of $1e-3$, and an $\mathcal{L}_1$ regularization parameter of $2e-5$. The network was implemented in Pytorch running on an NVIDIA TITAN Xp GPU. The network is trained for 500 epochs and the model with the minimum error on validation is selected.

4.2. Evaluation metrics

We also used several metrics to quantitatively evaluate the trained models on the test data. In particular, we consider the following:

Pearson Correlation coefficient

The similarity of the predicted velocity maps and their corresponding ground truth maps are quantified using the Pearson correlation coefficient (PCC) averaged over all the pixels of test data

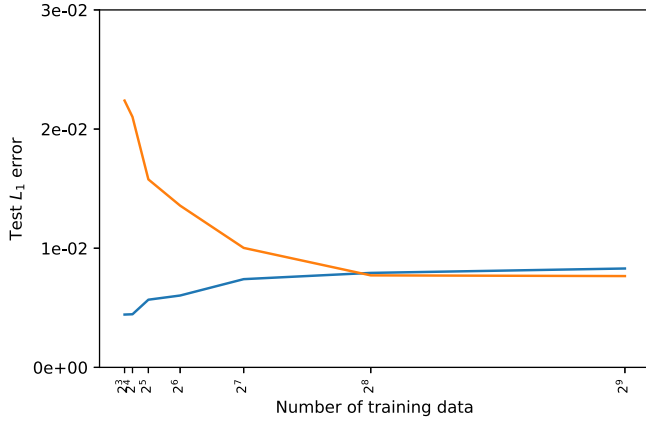


Fig. 6. Test L_1 error calculated for different numbers of training data used.

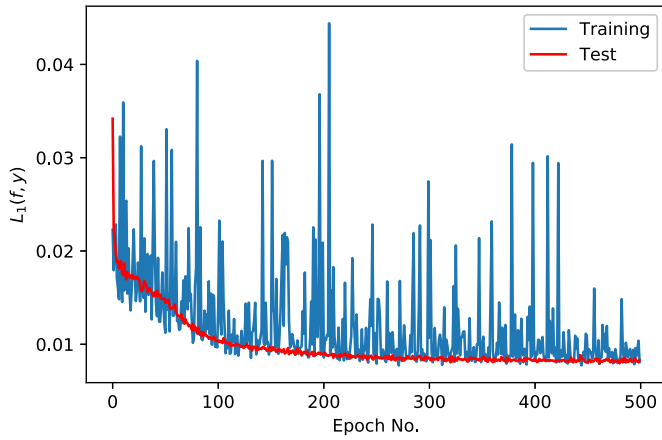


Fig. 7. Training and validation loss curve.

samples:

$$PCC = \frac{\text{cov}(\hat{\mathbf{u}}, \mathbf{u})}{\sigma_{\hat{\mathbf{u}}} \cdot \sigma_{\mathbf{u}}} \quad (11)$$

where $\sigma_{\hat{y}}$ and σ_y are the standard deviations of f and y . PCC is a measure of how the normalized variables tend to have the same sign and magnitude. A value of -1 indicates total disagreement and +1 total agreement. The correlation coefficient is 0 for completely random predictions.

Normalized Mean Squared Error (NMSE)

We calculate per-pixel NMSE which is an estimator of the overall deviations between predicted and measured values. The normalization of the MSE by the product of the observed and predicted means assures that the NMSE will not be biased towards models that tend to over predict or under predict. Note that, smaller values of NMSE denote better model performance. The NMSE is defined as:

$$NMSE = \frac{1}{T} \sum_{i=1}^T \frac{\|\hat{\mathbf{u}}^i - \mathbf{u}^i\|_2^2}{\hat{\mathbf{u}} \cdot \bar{\mathbf{u}}} \quad (12)$$

where $\bar{\mathbf{u}} = \sum_{i=1}^T \hat{\mathbf{u}}^i / T$ and $\bar{\mathbf{u}} = \sum_{i=1}^T \mathbf{u}^i / T$.

Fractional Bias

This fractional bias (FB) is normalized mean error and is dimensionless. It can vary between +2 and -2 and has an ideal value of zero for an ideal model. It is defined as:

$$FB = 2 \frac{\bar{\mathbf{u}} - \hat{\bar{\mathbf{u}}}}{\bar{\mathbf{u}} + \hat{\bar{\mathbf{u}}}} \quad (13)$$

5. Results

To develop the model, velocity values for the entire output set are normalized between zero and one (the minimum value of the velocity values is transformed linearly into zero, the maximum value is transformed into one, and every other value is transformed into a decimal between 0 and 1). All the simulation cases are divided into two sections; the first section with 12.8% of the data is used to train the model while the remaining data are used as test data. For the training, only x-direction flow, $\mathbf{F}_b = (F_{bx}, 0)$, is used. The test portion of the data, which is not used in the training process, is only used to examine the predictive capabilities and the robustness of the model. All the data samples for y-direction flow, $\mathbf{F}_b = (0, F_{by})$, are used as test data.

The surrogate model was trained with different number of training data (Fig. 6). By increasing the number of training samples from 8 to 512, L_1 loss decreases from $2.23e^{-2}$ to $7.31e^{-3}$. As the number of training data increases, the training error increases and validation error decreases; the gap between the training and validation error losses decrease, which signifies the reduction in the model variance. There are negligible differences (gap) between the training and validation errors, which show that the model benefits from both low bias and low variance errors.

For the remainder of the paper, the model trained by 12.8% of the data (256 out of 2000 total images) is used to evaluate the

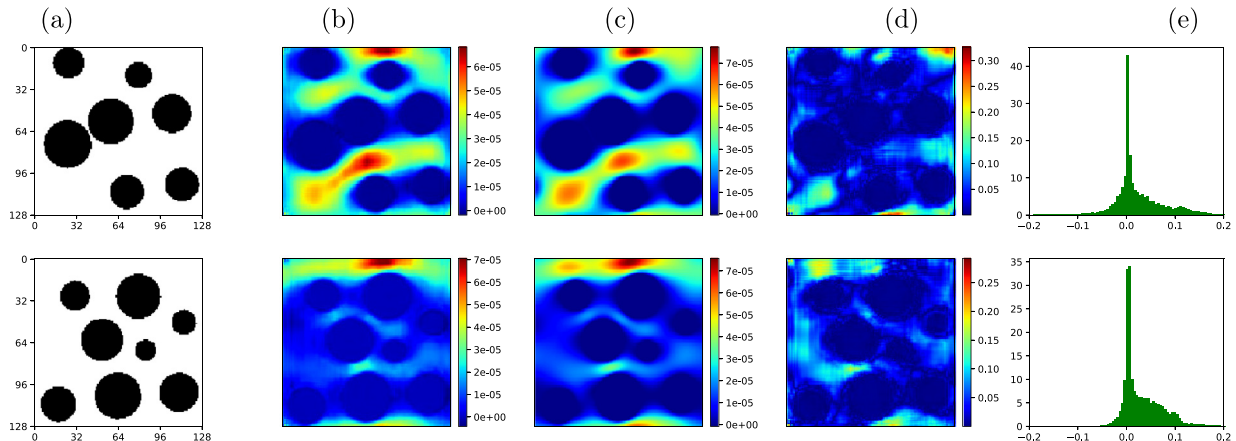


Fig. 8. Side-by-side comparison of the x-direction velocity values in x-direction flow predicted by the surrogate model vs. LB simulations results for two the training cases. (a) input images used for simulations; (b) model predictions; (c) LB simulation results; (d) absolute error percentage between (b) and (c); and (e) distribution of error percentage. Velocities are in lattice units, $[lu]$.

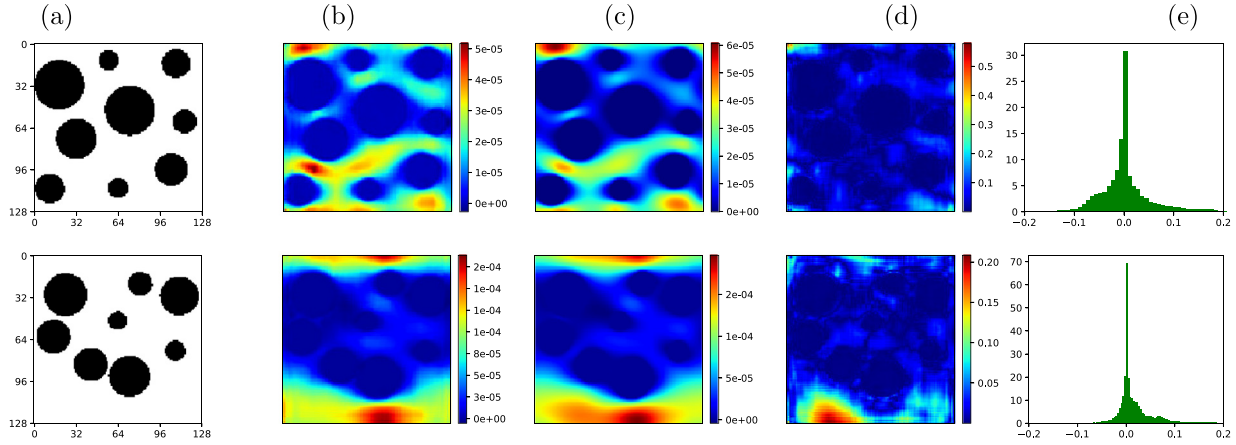


Fig. 9. Side-by-side comparison of the y-direction velocity values in y-direction flow predicted by the surrogate model vs. LB simulations results for two test samples. (a) input images used for simulations; (b) model predictions; (c) LB simulation results; (d) absolute error percentage between (b) and (c); and (e) distribution of error percentage. Velocities are in lattice units, [lu].

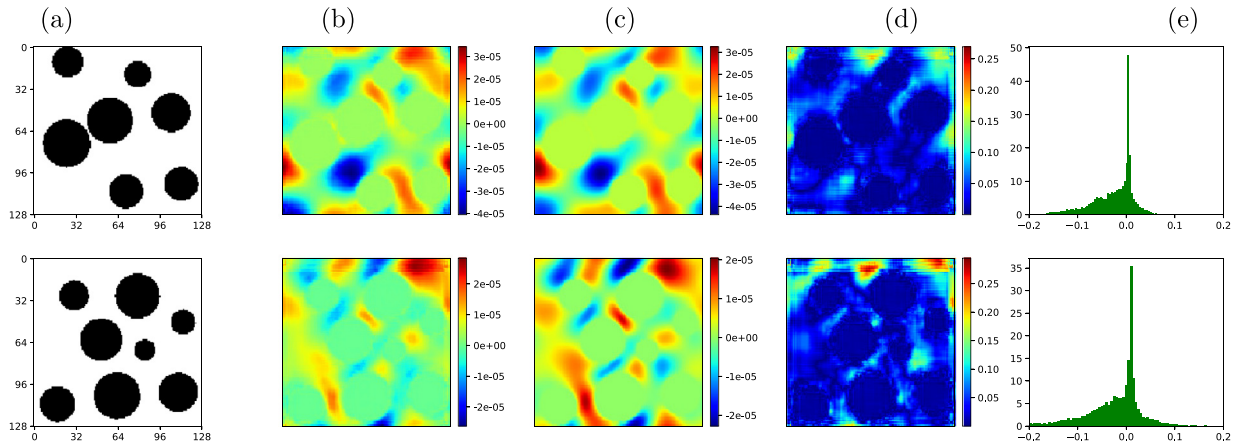


Fig. 10. Side-by-side comparison of the y-direction velocity values in x-direction flow predicted by the surrogate model vs. LB simulations results for two test samples. (a) input images used for simulations; (b) model predictions; (c) LB simulation results; (d) absolute error percentage between (b) and (c); and (e) distribution of error percentage. Velocities are in lattice units, [lu].

model robustness in predicting the velocity values for the test data. The training and validation loss curve for the training process is depicted in Fig. 7.

The performance of the surrogate model in predicting the grid-level velocity in the leading directions (x-direction velocity in x-direction flow and y-direction velocity in y-direction flow) is demonstrated in Figs. 8 and 9, respectively. The binary images are used for the LB simulations are shown in Figs. 8a and 9a. The regions away from the solid-pore interfaces exhibit higher velocity values (both in x- and y-directions) compared to the ones adjacent to the interfaces. Contour plots of the velocities for the developed model (Figs. 8b and 9b, respectively) and those computed using the LB simulations (Figs. 8c and 9c, respectively) show similar behavior. The velocity values within the solid circles (zero-velocity valued grids in Figs. 8c and 9c) are accurately predicted (Figs. 8b and 9b). The absolute values of the error, $\frac{|\hat{u}^i - u^i|}{u_{\max}^i - u_{\min}^i}$, for i^{th} test sample are

shown in Figs. 8d and 9d. Distributions of the error, $\frac{|\hat{u}^i - u^i|}{u_{\max}^i - u_{\min}^i}$, predicted are bound within 20% for almost all the cases, confirming the plausibility of the approach to accurately replicate numerical simulations (Figs. 8e and 9e).

The performance of the surrogate model in predicting the grid-level velocity in the secondary directions (y-direction velocity in x-direction flow and x-direction velocity in y-direction flow) is demon-

strated in Figs. 10 and 11, respectively. The binary images are used for the LB simulations are shown in Figs. 10a and 11a. Contour plots of the velocities for the developed model (Figs. 10b and 11b, respectively) and those computed using the LB simulations (Figs. 10c and 11c, respectively) show similar behavior. The velocity values within the solid circles (zero-velocity valued grids in Figs. 10c and 11c) are accurately predicted (Figs. 10b and 11b). The absolute values of the error in the samples are shown in Figs. 10d and 11d. The error distribution predicted is bound within 20% for almost all the cases. (Figs. 10e and 11e).

A grid-by-grid comparison of the predicted velocity values in the leading direction (x-direction velocity in x-direction flow) using the surrogate model and the LB simulations is performed (Fig. 12). Velocity profiles in a vertical (Fig. 12b) and horizontal (Fig. 12c) cross-sections are depicted for the simulation domain for one of the test cases. By inspection of these plots, one can see that the surrogate model mimics the LB simulation results with negligible errors (less than 15%).

A grid-by-grid comparison of the predicted velocity values in the secondary direction (y-direction velocity in x-direction flow) using the surrogate model and the LB simulations is performed (Fig. 13). Velocity profiles in a vertical (Fig. 13b) and horizontal (Fig. 13c) cross-sections are depicted for the simulation domain for one of the test cases. By inspection of these plots, one can see that

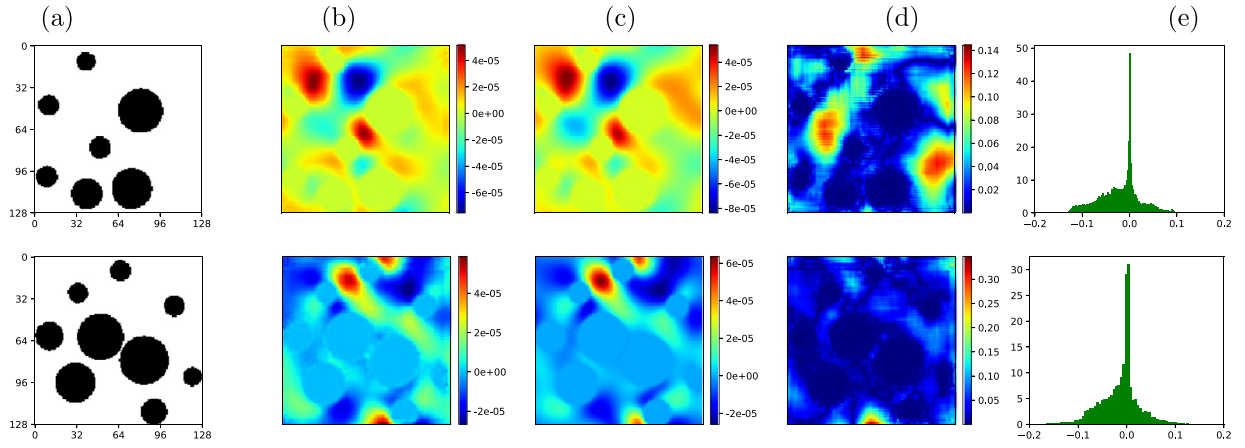


Fig. 11. Side-by-side comparison of the x-direction velocity values in y-direction flow predicted by the surrogate model vs. LB simulations results for two test samples. (a) input images used for simulations; (b) model predictions; (c) LB simulation results; (d) absolute error percentage between (b) and (c); and (e) distribution of error percentage. Velocities are in lattice units, $[lu]$.

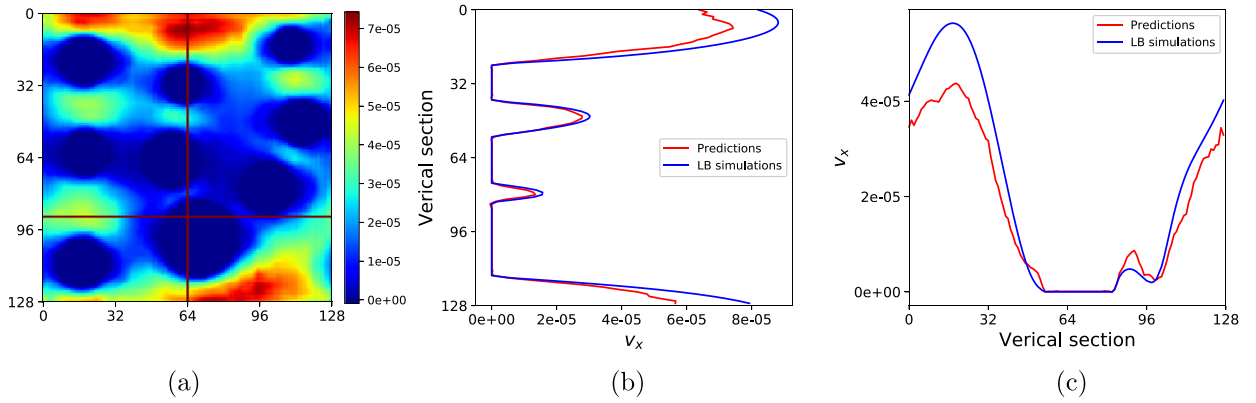


Fig. 12. Grid-by-grid comparison of the x-direction velocity in x-direction flow predicted by the surrogate model vs. LB simulations results for a test sample. (a) depiction of the velocity contour plots and the vertical and horizontal cross-sections; (b) velocity profiles along the vertical cross-section of the simulation domain; and (c) velocity profiles along the horizontal cross-section of the simulation domain. Velocities are in lattice units, $[lu]$.

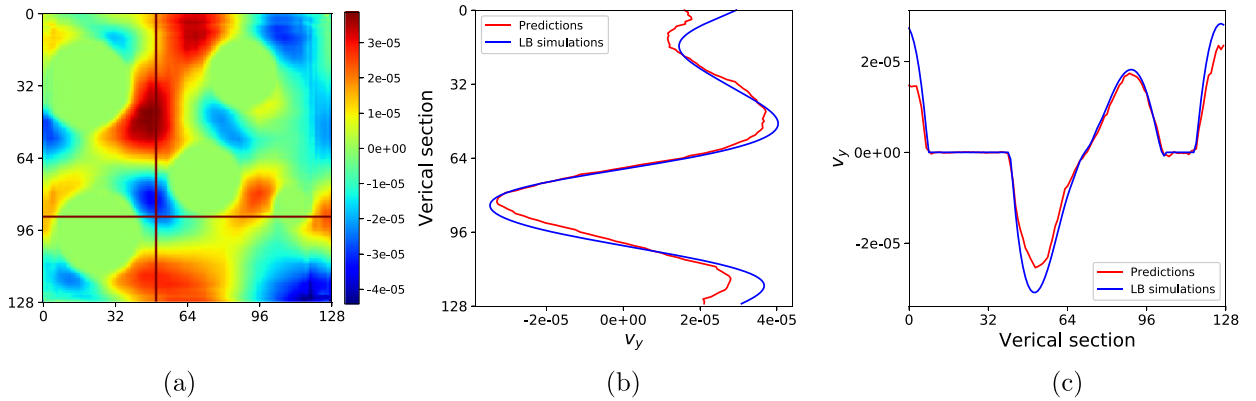


Fig. 13. Grid-by-grid comparison of the y-direction velocity in x-direction flow predicted by the surrogate model vs. LB simulations results for a test sample. (a) depiction of the velocity contour plots and the vertical and horizontal cross-sections; (b) velocity profiles along the vertical cross-section of the simulation domain; and (c) velocity profiles along the horizontal cross-section of the simulation domain. Velocities are in lattice units, $[lu]$.

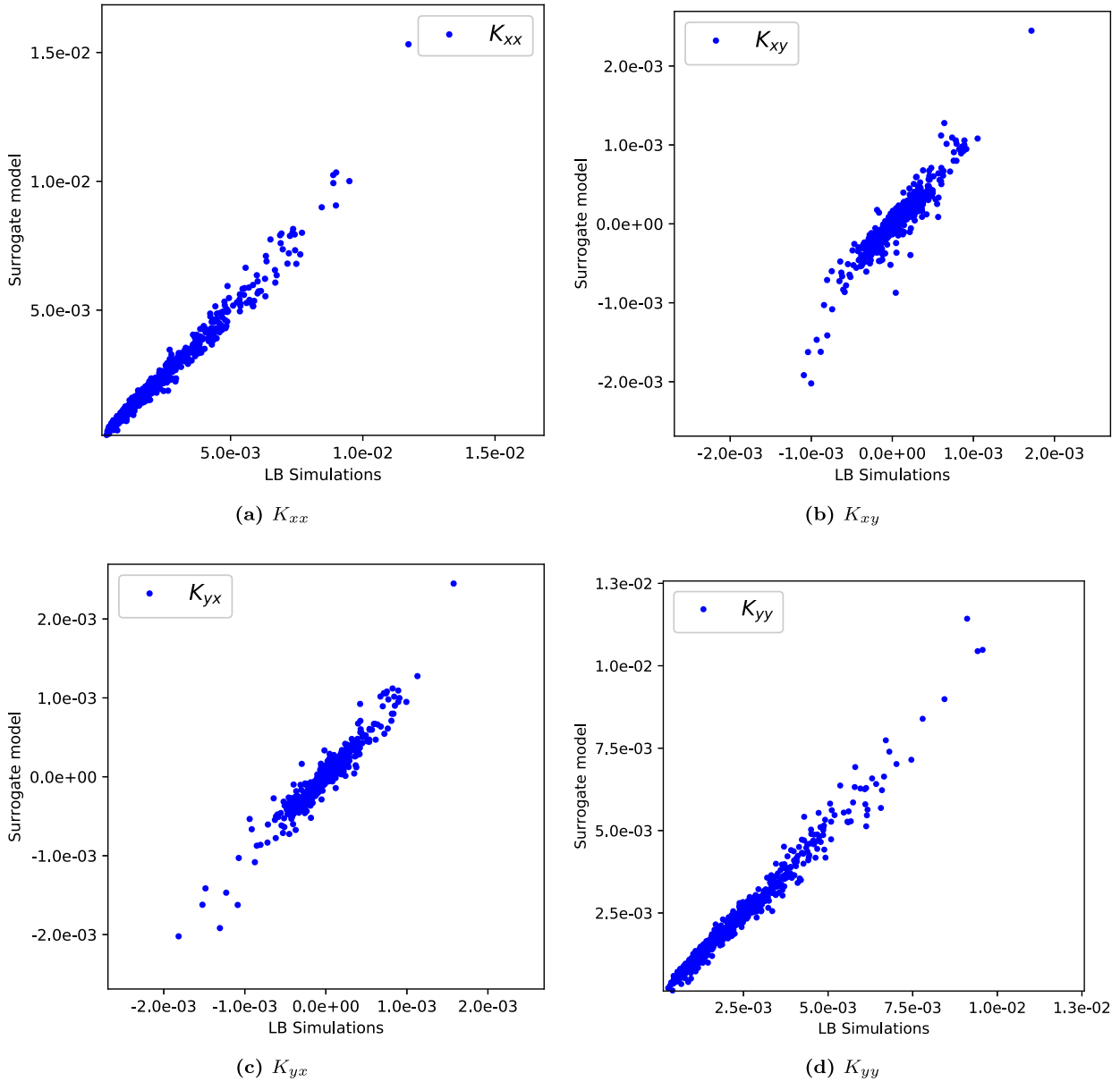


Fig. 14. Cross-plot of the predicted (a) K_{xx} , (b) K_{xy} , and (c) K_{yx} , and (d) K_{yy} , for test images vs. ground truth in $[lu^2]$.

the surrogate model mimics the LB simulation results with negligible errors (less than 15%).

The predicted elements of the permeability tensor values for all test cases are shown in Fig. 14. All the points are along the unit-slope line, which shows that predicted values are fairly close to the LB simulation results.

As was expected, based on the cross-plots of the predicted permeability values for test images vs. ground truth, the surrogate model has high PCC of 0.986 and 0.949 for u_x and u_y , respectively. The NMSE for u_x and u_y are 0.019 and 0.026, respectively. The FB for u_x and u_y are 0.022 and 0.28, respectively. Based on the results, the predicted permeability results for the 2D domains are predicted using the surrogate model with high accuracy.

The LB simulations of 2000 samples takes approximately 19,856 seconds (330.93 mins) on 4 processors. The implemented algorithm runs on a single NVIDIA Xp GPU, which requires about 8 minutes for training 500 epochs with 256 training samples. The training mini-batch size is 16. The predictions made by the surro-

gate model take less 2 seconds on a single GPU, which results in a significant amount of speed-up.

6. Performance of surrogate models trained with less data

In this section, the performance of the approach presented is evaluated using the models trained with fewer number of data samples. X-direction velocity profile for flow in the x-direction is plotted along a vertical line is depicted in Fig. 15. It can be seen that as the number of the training data increases, the velocity profiles tend to become closer to the LB simulation results (shown by the dashed red line in the right figure in Fig. 15). It should be pointed out that even for the smallest number of training data used (32), the x-direction velocity values are within %20 of the LB velocity values and their prediction is very close to those of LB simulations in the solid grains (zero velocity).

Cross-plot of the predicted K_{xx} for surrogate models developed by 32, 64, 128, and 256 data samples and LB simulation results are

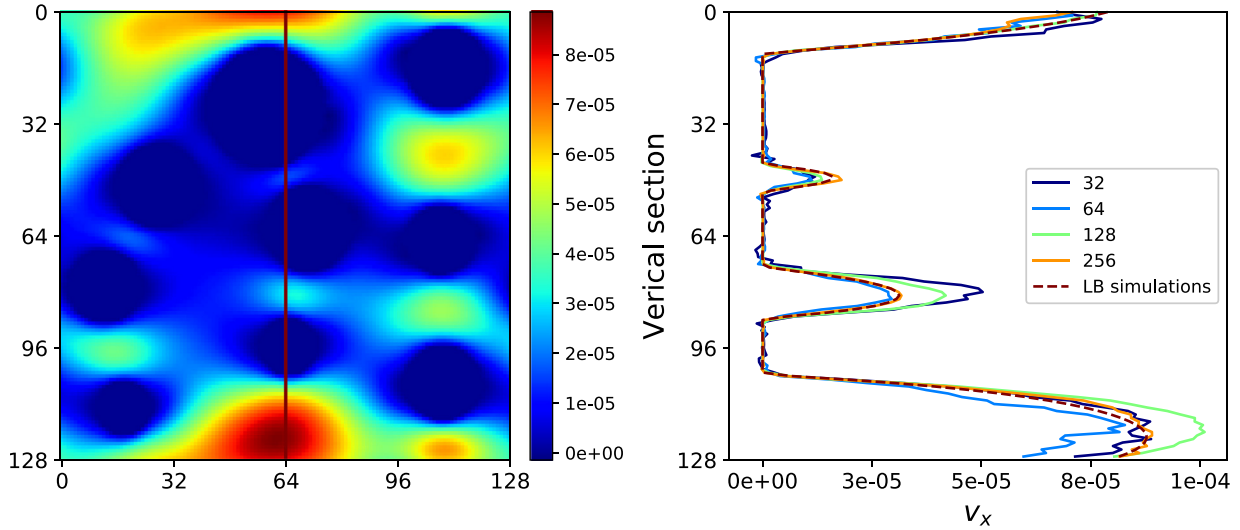


Fig. 15. Predicted velocity profiles along the vertical line (depicted on the contour maps of the left figure) for surrogate models trained using 32, 64, 128, and 256 image pairs.

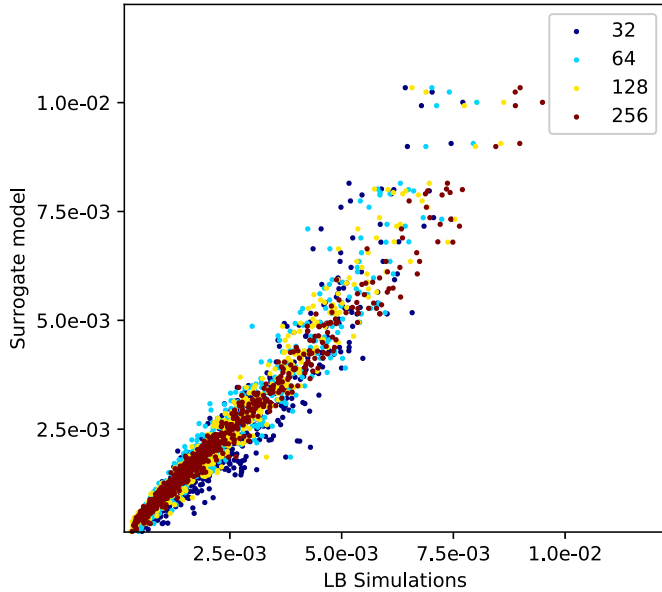


Fig. 16. Cross-plots of the predicted K_{xx} for surrogate models trained using 32, 64, 128, and 256 image pairs.

depicted in Fig. 16. The predicted permeability values using a surrogate model developed by 256 data samples are more condensed along the unit-slope line, showing that the predicted values are close to the LB predicted permeability values. For the models developed by less number of training data, the predictions are less accurate, but they show a fairly accurate permeability values for practical applications (within %10 of the ground truth).

7. Conclusions

A data-driven surrogate to high-fidelity numerical flow simulations is presented by employing a deep convolutional neural network based on contracting paths and residual blocks. The network consists of only convolutional layers and can take any arbitrary-sized image as input and generate an output of a similar size. The developed model captures the flow fields at the grid level for samples that had not been used in the development of the model. Permeability tensor for the samples of porous media can be de-

termined with a high accuracy at much lower computational costs using the presented approach. This work aspires to make a transformative leap by combining fluid flow modeling with ML to develop surrogate models, which can be used to determine the flow fields at very little additional cost.

Our work serves as a proof-of-concept study for determining microscopic (pore-scale) flow fields in porous media for determining macroscopic (the scales at which physical flow experiments are performed, usually larger than Representative Elemental Volumes (REV)s, at which the properties of porous media, such as porosity and permeability, are defined (orders of inches, feet, and tens of feet)) hydrodynamic properties, e.g., permeability and hydraulic tortuosity, using lattice Boltzmann simulations.

Author contribution

All authors have contributed sufficiently to the research work to be included as authors. To the best of our knowledge, no conflict of interest, financial or other, exists.

Declaration of Competing Interest

To the best of our knowledge, no conflict of interest, financial or other, exists.

Acknowledgments

The authors gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

References

- [1] Augier F, Idoux F, Delenne J-Y. Numerical simulations of transfer and transport properties inside packed beds of spherical particles. *Chem Eng Sci* 2010;65(3):1055–64.
- [2] Bosl WJ, Dvorkin J, Nur A. A study of porosity and permeability using a lattice Boltzmann simulation. *Geophys Res Lett* 1998;25(9):1475–8.
- [3] Chen S, Martinez D, Mei R. On boundary conditions in lattice Boltzmann methods. *Phys. fluids* 1996;8(9):2527–36.
- [4] Chukwudozie C, Tyagi M. Pore scale inertial flow simulations in 3-D smooth and rough sphere packs using lattice Boltzmann method. *AIChE J* 2013;59(12):4858–70.
- [5] Chukwudozie CP. Pore-scale lattice Boltzmann simulations of inertial flows in realistic porous media: a first principle analysis of the Forchheimer relationship. Louisiana State University; 2011. Master's thesis.

- [6] Giresan D, Giusti A, Gambardella LM, Schmidhuber J. Deep neural networks segment neuronal membranes in electron microscopy images. In: *Advances in neural information processing systems*; 2012. p. 2843–51.
- [7] Darcy H. Les fontaines publiques de la ville de Dijon, Dalmont. Paris: Dalmont; 1856.
- [8] d'Humieres D. Multiple-relaxation-time lattice Boltzmann models in three dimensions. *Philos Trans R Soc Lond Ser A* 2002;360(1792):437–51.
- [9] Ferreol B, Rothman DH. Lattice-Boltzmann simulations of flow through Fontainebleau sandstone. In: *Multiphase flow in porous media*. Springer; 1995. p. 3–20.
- [10] Ghassemi A, Pak A. Pore scale study of permeability and tortuosity for flow through particulate media using lattice Boltzmann method. *Int J Numer Anal Methods Geomech* 2011;35(8):886–901.
- [11] Ginzbourg I, Adler P. Boundary flow condition analysis for the three-dimensional lattice Boltzmann model. *J Phys II* 1994;4(2):191–214.
- [12] Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks. In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*; 2010. p. 249–56.
- [13] Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, et al. Generative adversarial nets. In: *Advances in neural information processing systems*; 2014. p. 2672–80.
- [14] Grigo C, Koutsourelakis P-S. Bayesian model and dimension reduction for uncertainty propagation: applications in random media. *SIAM/ASA J Uncertain Quantif* 2019;7(1):292–323.
- [15] Grigo C, Koutsourelakis P-S. A physics-aware, probabilistic machine learning framework for coarse-graining high-dimensional systems in the small data regime. *arXiv:190203968* 2019a.
- [16] He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*; 2016. p. 770–8.
- [17] Ioffe S, Szegedy C. Batch normalization: accelerating deep network training by reducing internal covariate shift. *arXiv:150203167* 2015.
- [18] Jin G, Patzek TW, Silin DB, et al. Direct prediction of the absolute permeability of unconsolidated and consolidated reservoir rock. In: *SPE annual technical conference and exhibition. Society of Petroleum Engineers*; 2004.
- [19] Kameda A. Permeability evolution in sandstone: Digital rock approach. Stanford University; 2004. Ph.D. Thesis.
- [20] Kingma D.P., Ba J. Adam: a method for stochastic optimization. *arXiv:1412.6980* 2014.
- [21] Kutz JN. Deep learning in fluid dynamics. *J Fluid Mech* 2017;814:1–4.
- [22] Latt J. Palabos, parallel lattice Boltzmann solver. FlowKit, Lausanne, Switzerland; 2009.
- [23] Lee C-Y, Xie S, Gallagher P, Zhang Z, Tu Z. Deeply-supervised nets. In: *Artificial intelligence and statistics*; 2015. p. 562–70.
- [24] Long J, Shelhamer E, Darrell T. Fully convolutional networks for semantic segmentation. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*; 2015. p. 3431–40.
- [25] Maier R, Bernard R. Lattice-Boltzmann accuracy in pore-scale flow simulation. *J Comput Phys* 2010;229(2):233–55.
- [26] Maier RS, Kroll D, Kutsovsky Y, Davis H, Bernard RS. Simulation of flow through bead packs using the lattice Boltzmann method. *Phys Fluids* 1998;10(1):60–74.
- [27] Maier RS, Kroll DM, Bernard RS, Howington SE, Peters JF, Davis HT. Pore-scale simulation of dispersion. *Phys Fluids* 2000;12(8):2065–79.
- [28] Manwart C, Aaltosalmi U, Koponen A, Hilfer R, Timonen J. Lattice-Boltzmann and finite-difference simulations for the permeability for three-dimensional porous media. *Phys Rev E* 2002;66(1):016702.
- [29] Marty NS, Chen H. Simulation of multicomponent fluids in complex three-dimensional geometries by the lattice Boltzmann method. *Phys Rev E* 1996;53(1):743.
- [30] Pan C, Hilpert M, Miller CT. Pore-scale modeling of saturated permeabilities in random sphere packings. *Phys Rev E* 2001;64(6):066702.
- [31] Pan C, Luo L-S, Miller CT. An evaluation of lattice Boltzmann schemes for porous medium flow simulation. *Comput fluids* 2006;35(8–9):898–909.
- [32] Psihogios J, Kainourgiakis M, Yiotis A, Papaioannou AT, Stubos A. A lattice Boltzmann study of non-newtonian flow in digitally reconstructed porous domains. *Transp Porous Media* 2007;70(2):279–92.
- [33] Qian Y-H, d'Humieres D, Lallemand P. Lattice BGK models for Navier-Stokes equation. *EPL (Europhys Lett)* 1992;17(6):479.
- [34] Raissi M. Deep hidden physics models: deep learning of nonlinear partial differential equations. *J Mach Learn Res* 2018;19(1):932–55.
- [35] Raissi M., Perdikaris P., Karniadakis G.E. Physics informed deep learning (Part I): data-driven solutions of nonlinear partial differential equations. *arXiv:1711.0561* 2017.
- [36] Raissi M., Yazdani A., Karniadakis G.E. Hidden fluid mechanics: a Navier-Stokes informed deep learning framework for assimilating flow visualization data. *arXiv:1808.04327* 2018.
- [37] Ronneberger O, Fischer P, Brox T. U-net: convolutional networks for biomedical image segmentation. In: *International conference on medical image computing and computer-assisted intervention*. Springer; 2015. p. 234–41.
- [38] Sanematsu P, Shen Y, Thompson K, Yu T, Wang Y, Chang D-L, et al. Image-based stokes flow modeling in bulk proppant packs and propped fractures under high loading stresses. *J Pet Sci Eng* 2015;135:391–402.
- [39] Seyedhosseini M, Sajjadi M, Tasdizen T. Image segmentation with cascaded hierarchical models and logistic disjunctive normal networks. In: *Proceedings of the IEEE international conference on computer vision*; 2013. p. 2168–75.
- [40] Shelton M, Gregory B, Lamson S, Moses H, Doughty R, Kiss T. Optimization of a transonic turbine airfoil using artificial intelligence, CFD and cascade testing. *ASME 1993 international gas turbine and aeroengine congress and exposition. American Society of Mechanical Engineers*; 1993. V03AT15A012–V03AT15A012.
- [41] Simonyan K., Zisserman A. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556* 2014.
- [42] Srivastava R.K., Greff K., Schmidhuber J.. Highway networks. *arXiv:1505.00387* 2015.
- [43] Stewart ML, Ward AL, Rector DR. A study of pore geometry effects on anisotropy in hydraulic permeability using the lattice-Boltzmann method. *Adv Water Resour* 2006;29(9):1328–40.
- [44] Struchtrup H. Macroscopic transport equations for rarefied gas flows. In: *Macroscopic transport equations for rarefied gas flows*. Springer; 2005. p. 145–60.
- [45] Succi S, Foti E, Higuera F. Three-dimensional flows in complex geometries with the lattice Boltzmann method. *EPL (Europhys Lett)* 1989;10(5):433.
- [46] Sukop M, Thorne DJ. Lattice Boltzmann modeling. Springer; 2006.
- [47] Swischuk R, Mainini L, Peherstorfer B, Willcox K. Projection-based model reduction: formulations for physics-based machine learning. *Comput Fluids* 2019;179:704–17. doi:10.1016/j.compfluid.2018.07.021.
- [48] Takkbiri-Borujeni A. Multi-scale modeling of inertial flows through propped fractures. Louisiana State University; 2013. Ph.D. Thesis.
- [49] Takkbiri-Borujeni, A., & Ayoobi, M. (2019). Application of Physics-Based Machine Learning in Combustion Modeling, 11th US National Combustion Meeting, Pasadena, California.
- [50] Takkbiri-Borujeni A., Kazemi H., Nasrabadi N. A data-driven proxy to stoke's flow in porous media. *arXiv:1905.06327* 2019.
- [51] Takkbiri-Borujeni A, Lane N, Thompson K, Tyagi M. Effects of image resolution and numerical resolution on computed permeability of consolidated packing using lb and fem pore-scale simulations. *Comput Fluids* 2013;88:753–63.
- [52] Takkbiri-Borujeni A, Tyagi M. Multiscale modeling of permeability and non-darcy factor in propped fractures. *Hydraul Fract J* 2016;3(2):64–74.
- [53] Tracey B, Duraisamy K, Alonso J. Application of supervised learning to quantify uncertainties in turbulence and combustion modeling. In: *51st AIAA aerospace sciences meeting including the new horizons forum and aerospace exposition*; 2013. p. 259.
- [54] Wang Q, Hesthaven JS, Ray D. Non-intrusive reduced order modeling of unsteady flows using artificial neural networks with application to a combustion problem. *J Comput Phys* 2019;384:289–307.
- [55] Whitaker S. Flow in porous media I: a theoretical derivation of Darcy's law. *Transp Porous Media* 1986;1(1):3–25.
- [56] Zagoruyko S., Komodakis N. Wide residual networks. *arXiv:1605.07146* 2016.
- [57] Zeiler M.D., Krishnan D., Taylor G.W., Fergus R. Deconvolutional networks 2010.
- [58] Zhan X, Schwartz LM, Toksöz MN, Smith WC, Morgan FD. Pore-scale modeling of electrical and fluid transport in Berea sandstone. *Geophysics* 2010;75(5):F135–42.
- [59] Zhu Y, Zabarans N. Bayesian deep convolutional encoder-decoder networks for surrogate modeling and uncertainty quantification. *J Comput Phys* 2018;366:415–47.