

École polytechnique de Louvain

Porous media reconstruction using deep texture synthesis

Author: **Mattéo COUPLET**

Supervisors: **Laurent DEMANET, Laurent JACQUES**

Reader: **Jean-François REMACLE**

Academic year 2019–2020

Master [120] in Mathematical Engineering

Remerciements

Je souhaite adresser mes sincères remerciements au Professeur Laurent Demanet pour m'avoir proposé ce fascinant sujet de recherche, et m'avoir accueilli l'été dernier au MIT pour son démarrage. Son grand intérêt pour mon travail ainsi que ses conseils avisés m'ont donné la motivation pour avancer. De manière égale, ces remerciements vont au Professeur Laurent Jacques pour avoir accepté de superviser ce mémoire. Son regard critique et ses conseils ont grandement contribué à ce travail.

Merci au Professeur Jean-François Remacle d'avoir accepté de lire et évaluer ce mémoire.

Enfin, merci à mes parents et à mes frères pour leur soutien inconditionnel, et plus particulièrement pour m'avoir supporté durant ce confinement. Merci à Diane et à sa famille pour leurs précieux encouragements.

Abstract

The field of Digital Rock Physics relies on three-dimensional representations of porous media, acquired using X-ray microtomography (μ CT), for extracting macroscopic properties of rocks. The resolution of such representations is limited by the resolution of μ CT; higher-resolution information is available, but only with two-dimensional imaging techniques. *Porous media reconstruction* aims at reconstructing three-dimensional samples of a rock, given only a two-dimensional image, while preserving the rock’s macroscopic properties. This work proposes a deep learning solution to this problem that leverages *texture networks*, which have been recently acclaimed for their impressive results in texture synthesis. The solution learns an implicit representation of the pore structure, which allows it to generate multiple realizations of arbitrary size very rapidly. To validate the reconstruction procedure, we introduce a set of metrics that compare the reconstructed samples with the ground truth. Our results show that the solution successfully preserves the relevant rock properties, both in terms of statistical quantities and physical macroscopic properties such as the rock’s permeability. We conclude that texture networks are an effective approach to porous media reconstruction, and they open up interesting research directions in the domain.

Contents

1	Introduction	2
1.1	Motivation and context	2
1.2	Aim of the work	6
1.3	Structure of the work	6
2	State of the Art	7
2.1	Definition of a porous medium and its properties	7
2.2	A historical overview of porous media reconstruction	9
2.3	Related work	15
3	Deep Texture Synthesis for Porous Media Reconstruction	16
3.1	Introduction	16
3.2	Description of the method	18
3.3	Results	25
4	Validation of the Reconstruction	32
4.1	Validation methodology	32
4.2	Validation metrics and their computation	32
4.3	Results	43
4.4	Discussion	43
5	Conclusion and Future Work	51

Chapter 1

Introduction

1.1 Motivation and context

One of the greatest challenges of the twenty-first century is how to provide sufficient energy for a growing and (hopefully) more prosperous population, while coping with the threat of climate change. In 2018, fossil fuel represented 85% of the world's total primary energy consumption [Pet19], and there is no reason to believe this number will significantly go down in the years to come. On the other hand, the Intergovernmental Panel on Climate Change (IPCC) recommends to drastically reduce the greenhouse gas emissions to reach net zero around 2050, to stay below 1.5 °C of global warming and mitigate the unprecedented consequences on both humans and ecosystems [Cli18]. These challenges have a common scientific underpinning: flow in porous media. Fossil fuel is extracted from porous rocks; and a promising method to reduce atmospheric emissions of carbon dioxide is to collect it from major sources, such as fossil-fuel burning power stations, and inject it in deep underground reservoirs. Oil and gas recovery and carbon storage both rely on quantifying how fluids flow through porous rocks.

In combination with the huge practical applications that drive the science, two recent advances have transformed our understanding of how fluids move at the pore scale: the development of methods to image rocks, the pore space and the fluids within them, in three dimensions at a resolution from nanometers to centimeters; and the availability of good scientific computing software for solving flow problems.

The present work focuses on the former, and seeks to contribute to the recent effort to enhance porous media models by leveraging new generation computational approaches, and in this way form a better understanding of the physical processes at play inside these media.

Digital rock physics The aim of *rock physics* is to discover, understand and model the relations between remotely-sensed geophysical observables and local rock properties. Conventional rock physics models are based either on empirical relations obtained from laboratory experiments, or theoretical models based on idealized microstructures, calibrated with measurements. These models have provided important insights; however, they are oversimplified with respect to the geometry they represent and there is a lack of universality because different rock properties are extracted from different models for the microstructure. Recently, *digital rock physics* has rapidly emerged as a potential source of both valuable rock property determination and better understanding of pore-scale processes governing these properties. The principle of digital rock physics is "image-and-compute": imaging the 3D geometry of a rock, then computationally simulating the physical process we are interested in: fluid flow for permeability, electrical current flow for resistivity, elastic deformation for elastic moduli, and so on. With the advent of fine-resolution 3D imaging hardware and software, digital rock physics is set to be a game

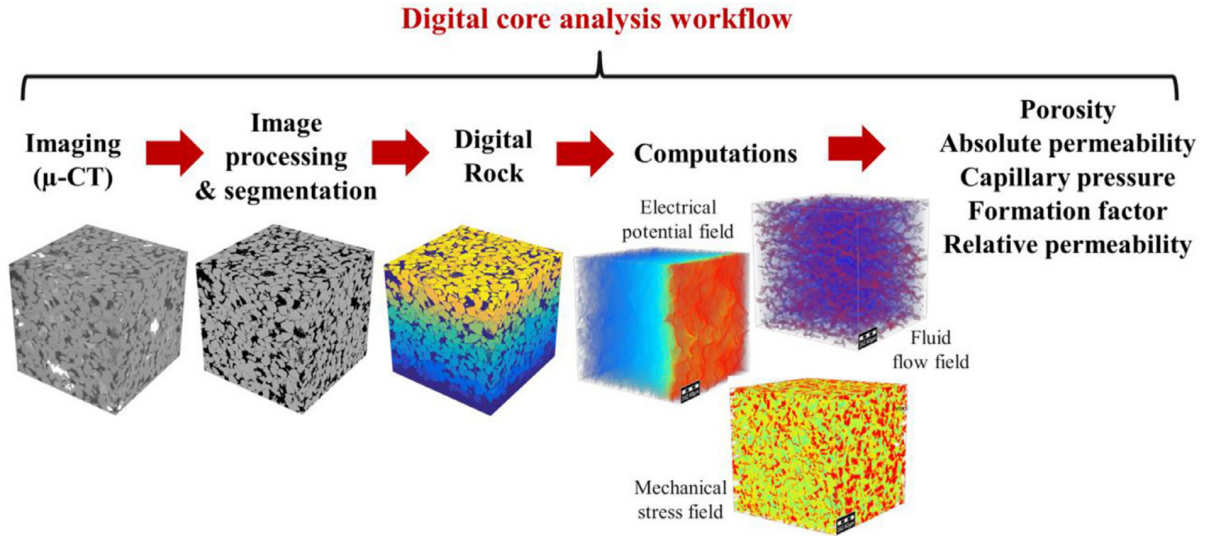


Figure 1.1: A conventional Digital Rock Physics workflow. From [Sax+17]

changer.

The typical Digital Rock Physics workflow is illustrated in fig. 1.1 and goes as follows:

1. A 3D image of the rock is acquired. A modern method to acquire images of pore geometries is the micro-scale x-ray computed tomography (micro-CT). This method allows to measure the local absorption of x-ray with a small rock sample with a typical size of a few millimeters. Three-dimensional images are reconstructed from a large number of projections of the imaged object, obtained at different projection angles. The result is a grayscale image where the brightness depends on the imaged material properties.
2. Segmentation is performed on the scanned rock, that is, pore and mineral phases are identified and labeled within the image. Due to the size of the 3D images, manual segmentation is usually not tractable and image processing algorithms are required to automatically perform this task. Common 3D segmentation techniques are spatial filtering, noise and artifact removal, thresholding, morphological operators and cluster analysis. The reader can refer to [IGT09] for a review of image segmentation methods applied to porous media.
3. Physical processes are simulated in the segmented pore-space and volume-averaged physical property estimates are obtained. Examples of properties of interest in rock physics include hydraulic permeability and electrical resistivity. Over the last decades, various algorithms have been proposed for simulating these micro-scale physical processes and estimating the corresponding macroscopic properties [And+13].

Current micro-CT scanners can produce images of around $1000^3 - 2000^3$ voxels (the three-dimensional extension of pixels) with rock samples of a few mm across, which allows to achieve resolutions of a few microns. An example of a micro-CT scan applied to a specific rock is illustrated in Fig. 1.2. Sub-micron resolution can be achieved with *nano-CT* using a specially designed instrument and smaller samples. The best nanotomography scanners can have spatial resolutions down to 20 nm [De +16], but the commercial availability of these scanners is still limited and very expensive.

On the other hand, *scanning electron microscopy* (SEM) techniques can be used to image samples in two dimensions. Here, voxel sizes as small as 5 nm can be achieved; this resolution

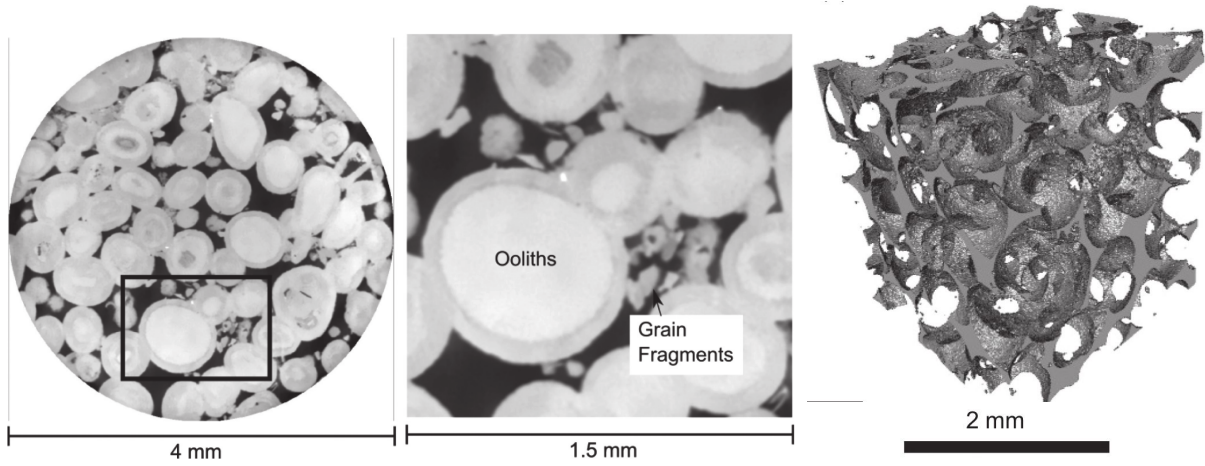


Figure 1.2: Micro-CT scans of a Ketton limestone. Left: Two-dimensional cross-section of three-dimensional micro-CT images of a Ketton limestone sample (grains are bright, pores are dark). Center: Close-up view showing the structure of the grains. Right: Three-dimensional micro-CT image, binarized into pore and grain (pore space is shown here). From [Blu17].

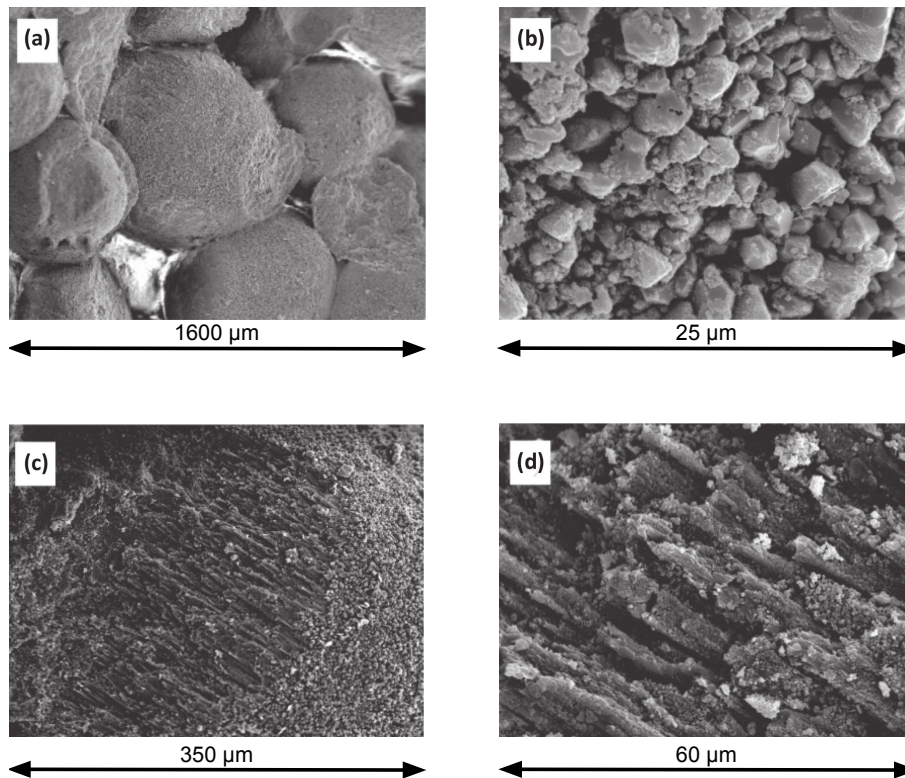


Figure 1.3: Scanning electron microscope images of the Ketton limestone from fig. 1.2, showing that the grains are micro-porous with different textures. From [Blu17].

is necessary to observe the full range of micro-porosity in rocks such as carbonates and shales. Electron microscopy, however, destroys the sample and therefore cannot be used in conjunction with dynamic flow experiments. Figure 1.3 shows further detail of the rock from fig. 1.2 using scanning electron microscopy showing that the individual grains are micro-porous at the sub-micron scale with different textures in different parts of the rock.

Up to now, rock physicists have relied heavily on micro-CT scans to upscale macroscopic properties, since it can produce convenient three-dimensional images in a nondestructive fashion. Which means that the finest scale we have taken into account is the micron-scale. However, for most rocks it is really the nanometer-scale that controls the flow properties. Being able to incorporate nano-scale information into the available three-dimensional images at the micron-scale could help create porous media models that combine multiple scales and are therefore more representative of the reality, thereby allowing to upscale macroscopic properties more precisely.

While two-dimensional SEM images offer great potential in studying microporosity, their direct usage provides poor insight into the pore structure: pores appearing to be isolated in 2D can be connected via the third dimension, and running direct numerical simulations of physical processes on these images is hence worthless. This observation leads to the overarching question that motivates the *porous media reconstruction* task:

Can we indirectly exploit two-dimensional images to gain insights about the macroscopic properties of porous media?

Porous media reconstruction Since direct upscaling of physical properties is impossible to do on 2D images, an indirect approach consists in learning a 3D representation of the porous medium that is *representative* of the original porous medium from which the image was taken.

More precisely, let f denote a *random field* on $\mathbb{R}^3$, i.e. a collection of random variables $\{f(x) : x \in \mathbb{R}^3\}$, that represents our ground truth porous medium. Let g be the restriction of f to some plane S , which represents the "image" of a slice through the porous medium. Define some metrics on f : $F_1(f), \dots, F_M(f)$, that represent the physical properties we are interested in¹. The reconstruction problem is to find, given g only, a random field $\hat{f}$ on $\mathbb{R}^3$ that preserves the chosen metrics: $F_m(\hat{f}) = F_m(f)$ for every m . There are several difficulties related to this problem:

- In practice, we are only provided a finite sample of g , and we can only produce finite samples of $\hat{f}$,
- The metrics $F_1, \dots, F_M$ are usually not available in closed form,
- It is not even guaranteed that the slice g contains enough information to preserve the metrics.

The ambition of the reconstruction task is not to revolutionize digital rock physics by providing a universal upscaling methodology, but rather to demonstrate that the information available in nanoscale 2D images is exploitable, and thereby advance in the quest for better porous media models.

¹ Of course there are assumptions that need to be made on f for the metrics to be well-defined; we ignore this aspect for now.

1.2 Aim of the work

The porous media reconstruction literature dates back to the 1980s, and a detailed survey is conducted in chapter 2. In recent years, the development of artificial neural networks and deep learning have transformed our way of approaching many engineering problems. This is particularly true for problems involving two- or three-dimensional images. The present work seeks to join the effort of developing the next generation of computational methods in the broad field of porous media modelling. More precisely, our goal is to show that deep learning is a powerful tool for solving the reconstruction problem described before.

Texture synthesis is a problem that arises in the computer graphics community, which consists in constructing a large image from a small texture example, taking advantage of its structural content. Connections between porous media modelling and texture synthesis are not new: see, e.g., [ML14] for a survey on the subject. A related task, *solid texture synthesis*, consists in constructing a three-dimensional volume such that any cross-section of this volume is visually similar to a small texture example. This task is very similar to the porous media reconstruction task; only the objective is slightly different.

Recently, major advances have been made in texture synthesis by the development of deep learning solutions (hence *deep texture synthesis*). In particular, the *texture networks* developed by [Uly+16] produce spectacular results. Even more recently, [Gut+20] have applied texture networks to the *solid* texture synthesis problem and obtained very good results.

A breakthrough in solid texture synthesis may translate in a breakthrough in porous media reconstruction. Therefore, the aim of this work is to assess how well solid texture networks are able to deal with the porous media reconstruction task.

1.3 Structure of the work

This master’s thesis is structured as follows:

- In Chapter 2, we introduce the mathematical and physical concepts related to porous media that will be useful throughout the thesis. We also provide an historical overview of the important publications that address the reconstruction task, as well as a selection of works related to the application of deep learning in porous media modelling.
- In Chapter 3, we present the reconstruction procedure in detail. We provide a dataset of porous samples on which the solution is applied, and we show visual results.
- In Chapter 4, we propose a validation methodology as well as a set of validation metrics and numerical schemes that will help us assess the quality of the reconstruction. Results of the validation are exposed exhaustively and discussed.
- In Chapter 5, we conclude the work by providing guidelines for future improvements and research directions.

Chapter 2

State of the Art

2.1 Definition of a porous medium and its properties

The goal of this section is to formally define the representation of porous media in this work, as well as to give an overview of the morphological quantities and macroscopic properties that characterize them.

A *porous medium* is any material that contains both a *solid matrix* and void space: the *pores*. The pores may contain fluid phases (gas or liquid); in the present work, we ignore the possibility of having multiple fluid phases: at any point, a porous medium is either in the pore phase, or the solid phase.

Formally, we idealize a two-phase porous medium as a domain of space $\mathcal{V}(\omega) \subseteq \mathbb{R}^3$ (where the realization ω is taken from some probability space Ω), such that $\mathcal{V}$ is composed of two regions: region $\mathcal{V}_1(\omega)$, corresponding to phase 1 (the *pore* phase), and region $\mathcal{V}_2(\omega)$, corresponding to phase 2 (the *solid* phase). Let $\partial\mathcal{V}(\omega)$ denote the interface between $\mathcal{V}_1(\omega)$ and $\mathcal{V}_2(\omega)$. For a given realization ω , we define the *characteristic function* $I(\mathbf{x}, \omega)$ of phase 1:

$$I(\mathbf{x}, \omega) = \begin{cases} 1, & \text{if } \mathbf{x} \in \mathcal{V}_1(\omega) \text{ (pore phase),} \\ 0, & \text{if } \mathbf{x} \in \mathcal{V}_2(\omega) \text{ (solid phase).} \end{cases} \quad (2.1)$$

We can as well define the characteristic function for the interface $M(\mathbf{x}, \omega)$ as

$$M(\mathbf{x}, \omega) = |\nabla I(\mathbf{x}, \omega)|. \quad (2.2)$$

In the sequel, we drop the mention of the realization ω for better readability.

2.1.1 Stationarity and Isotropy

It is useful to note that a porous medium may also be interpreted as a *random field*, i.e., a collection of random variables

$$\{I(\mathbf{x}) : \mathbf{x} \in \mathbb{R}^3\}. \quad (2.3)$$

In the special case of two-phase porous media, this random field is composed of *binary* random variables. This interpretation allows to formally define the notions of *stationarity* and *isotropy*. A random field is characterized by its mean function

$$m(\mathbf{x}) = \mathbb{E}[I(\mathbf{x})] \quad (2.4)$$

and its covariance function

$$C(\mathbf{x}, \mathbf{y}) = \mathbb{E}[(I(\mathbf{x}) - m(\mathbf{x}))(I(\mathbf{y}) - m(\mathbf{y}))]. \quad (2.5)$$

The random field is then said to be *homogeneous* or *stationary* if the mean $m(\mathbf{x})$ is constant, and the covariance $C(\mathbf{x}, \mathbf{y})$ is a function of the difference $\mathbf{y} - \mathbf{x}$ only. If it is also true that $C(\mathbf{x}, \mathbf{y})$ is a function of the Euclidean distance $|\mathbf{y} - \mathbf{x}|$ only, we say that the random field is also *isotropic*.

As we will see, stationarity and isotropy are important properties that sometimes need to be assumed to properly define higher-order quantities.

2.1.2 Morphological measures

We now define a set of quantities that characterize the morphology of a porous medium. They are present throughout the porous media literature and provide simple but powerful descriptors of the porous geometry.

Volume fraction If a porous medium is stationary we can define, respectively, the pore and solid *volume fractions* ϕ_1 and ϕ_2 :

$$\phi_1 = \langle I(\mathbf{x}) \rangle, \quad \phi_2 = \langle 1 - I(\mathbf{x}) \rangle = 1 - \phi_1, \quad (2.6)$$

where $\langle \cdot \rangle$ denotes an ensemble average. When the subscript is dropped we mean the pore volume fraction (or *porosity*): $\phi = \phi_1$. It may be interpreted as the probability of finding a point in the pore phase.

Two-point correlation function Given two arbitrary points $\mathbf{x}$ and $\mathbf{y}$, the two-point correlation function is defined as

$$S_2(\mathbf{x}, \mathbf{y}) = \langle I(\mathbf{x})I(\mathbf{y}) \rangle. \quad (2.7)$$

It can be interpreted as the probability of finding two points at positions $\mathbf{x}$ and $\mathbf{y}$ both in the pore phase. Notice that $S_2(\mathbf{x}, \mathbf{y}) = C(\mathbf{x}, \mathbf{y}) - m(\mathbf{x})m(\mathbf{y})$ and therefore, for stationary media, S_2 depends only on the difference $\mathbf{r} = \mathbf{y} - \mathbf{x}$ and for isotropic media S_2 depends only on $r = |\mathbf{r}|$. We shall generally abuse notations and write

$$S_2(\mathbf{x}, \mathbf{y}) = S_2(\mathbf{y} - \mathbf{x}) = S_2(|\mathbf{y} - \mathbf{x}|) \quad (2.8)$$

even though these are three different functions. Notice that for all isotropic media we have

$$S_2(0) = \phi, \quad \lim_{r \rightarrow \infty} S_2(r) = \phi^2; \quad (2.9)$$

The first property comes from eq. (2.6), and the second property comes from the fact that two points separated by a long distance are decorrelated.

Lineal-path function An important morphological descriptor of the structure of a medium is the *lineal-path function* $L(\mathbf{x}, \mathbf{x} + \mathbf{r})$, which is defined as the probability that the line segment $[\mathbf{x}, \mathbf{x} + \mathbf{r}]$ lies entirely in phase 1 [LT92]. Contrary to the two-point correlation function, the lineal path contains some information about the *connectedness* between pores, at least along a line segment. In a stationary medium, the lineal-path function depends only on the difference $\mathbf{r}$ between the two endpoints, and for an isotropic medium it only depends on the distance $r = |\mathbf{r}|$ and can therefore be expressed simply as $L(r)$. For an isotropic medium having volume fraction ϕ we have

$$L(0) = S_2(0) = \phi. \quad (2.10)$$

2.1.3 Macroscopic properties

A macroscopic property of a porous medium is a property that characterizes a physical process at a length scale that is much larger than the pore size scale. The physical process is described by a local law, acting at the pore scale, and by averaging over the medium one obtains a global law that describes the process at a macroscopic scale. These quantities are often useful from an engineering point of view. This section briefly introduces two of them; a more detailed treatment is done in chapter 4.

Hydraulic permeability Permeability quantifies the ability of a porous medium to let fluids pass through it, under a prescribed pressure gradient. The slow flow of an incompressible viscous fluid through porous media is often described by *Darcy's law*:

$$\mathbf{v} = -\frac{\kappa}{\mu} \nabla p, \quad (2.11)$$

where $\mathbf{v}$ is the average velocity of the fluid flowing through the medium, κ is the single-phase (or absolute) permeability of the medium, μ is the dynamic viscosity of the fluid, and ∇p is the applied pressure gradient.

Effective electrical conductivity A porous medium may be seen as a resistive material, in the electrical sense. If we consider that the pore phase is conductive with known conductivity σ_1 and the solid phase is insulating, then the current density $\mathbf{J}$ is related to the applied voltage V by Ohm's law:

$$\mathbf{J} = -\sigma_{\text{eff}} \nabla V, \quad (2.12)$$

where σ_{eff} is defined as the *effective electrical conductivity*. This quantity can be made dimensionless by defining the so-called *formation factor* $\mathcal{F} = \sigma_1/\sigma_{\text{eff}}$.

2.2 A historical overview of porous media reconstruction

We have selected a set of important works that tackle the porous media reconstruction problem, in order to provide some perspective about the different methods that have been proposed in the past.

2.2.1 Matching first- and second-order statistics

The first attempts at porous media reconstruction consist in generating 3D models that share the same first- and second-order statistics as the provided 2D image. Before explaining the approach in more details, we need to define *Gaussian random fields* (GRFs):

Gaussian random fields GRFs form a special class of random fields that is remarkable due to the simplicity of their representation. Formally, a real-valued Gaussian random field is a random field F on a set X for which the (finite-dimensional) distributions of $(F(\mathbf{x}_1), \dots, F(\mathbf{x}_n))$ are multivariate Gaussian for any $n \geq 1$ and every $(\mathbf{x}_1, \dots, \mathbf{x}_n) \in X^n$. Since multivariate Gaussian distributions are completely determined by their means and covariances, it is immediate that Gaussian random fields are determined by their mean function $m(\mathbf{x})$ and covariance function $C(\mathbf{x}, \mathbf{y})$. This simple representation by means of two functions is one of the main reasons that Gaussian fields are popular choices of random fields.

It will be useful to be able to generate a realization of a (stationary) GRF with a given mean m and covariance $C(\mathbf{r})$. This task can be done easily and efficiently in $\mathbb{R}^d$ by working in the Fourier

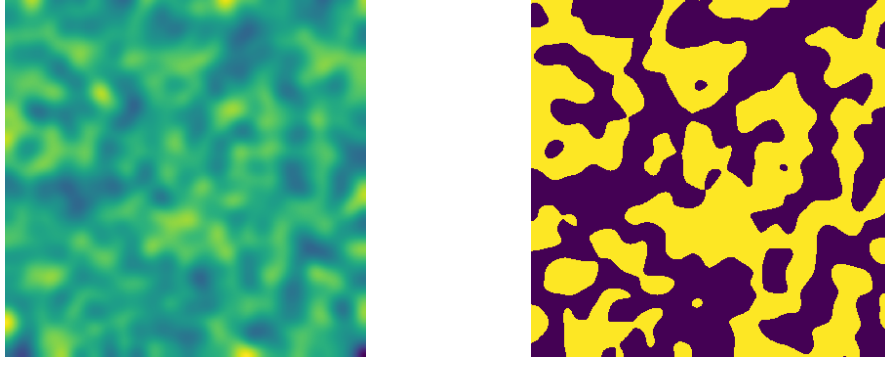


Figure 2.1: Example of a Gaussian random field realization (left) and its binary truncation (right).

domain. Let W be a Gaussian white noise random field on $\mathbb{R}^d$ (i.e., $W(\mathbf{x}) \sim \mathcal{N}(0, 1)$, $\mathbf{x} \in \mathbb{R}^d$) and γ the Fourier transform of the target covariance C (γ is also known as the *power spectral density* of the process, according to the Wiener–Khinchin theorem). Then, the random field [LP11]

$$Y_c(\mathbf{x}) = \left(\mathcal{F}^{-1} \gamma^{1/2} \mathcal{F} W \right)(\mathbf{x}), \quad \mathbf{x} \in \mathbb{R}^d \quad (2.13)$$

is centered Gaussian and has covariance $\mathbb{E}_{\mathbf{x}} [Y_c(\mathbf{x}) Y_c(\mathbf{x} + \mathbf{r})] = C(\mathbf{r})$. The targeted mean m can then be matched by taking $Y(\mathbf{x}) = Y_c(\mathbf{x}) + m$. eq. (2.13) suggests an efficient algorithm for generating Gaussian random fields, that only needs to perform two fast Fourier transform operations, resulting in an $\mathcal{O}(N \log N)$ complexity where N is the number of discretization points.

Reconstruction using truncated Gaussian random fields A convenient way to obtain a two-phase porous model is to define a Gaussian random field $Y(\mathbf{x})$ and to segment it using some threshold α ; the characteristic function I of the binary model is then given by

$$I(\mathbf{x}) = \begin{cases} 1, & \text{if } Y(\mathbf{x}) \geq \alpha \text{ (pore phase)} \\ 0, & \text{otherwise (solid phase)} \end{cases} \quad (2.14)$$

An example of a GRF realization and its truncation are given in fig. 2.1. A reconstruction method was proposed by [Qui84] and relies on the fact that if a GRF is stationary and isotropic, then it shares the same mean m and covariance function $C(r)$ as any of its slices. The approach consists in modeling the porous medium as a truncated GRF that preserves the first- and second order moments of the provided image. This is done in several steps:

1. Given a 2D binary random field $I(\mathbf{x})$, we compute its *volume fraction* $\phi = \langle I(\mathbf{x}) \rangle$ and *two-point correlation function* $S_2(r) = \langle I(\mathbf{x}) I(\mathbf{y}) \rangle$, with $r = \|\mathbf{y} - \mathbf{x}\|$. Estimating these quantities given a sample of I can be done efficiently by working in the Fourier domain.
2. Given ϕ and S_2 , we need to find a GRF $Y(\mathbf{x})$ that correctly reproduces the statistics I and S_2 *after thresholding* at some level α . By convention we set Y to have zero mean and unit variance: $\langle Y(\mathbf{x}) \rangle = 0$ and the covariance $g(r) = \langle Y(\mathbf{x}) Y(\mathbf{y}) \rangle$ is such that $g(0) = 1$. Let $\hat{I}$ denote the 3D binary random field obtained by thresholding Y at level α . Then, since for every $\mathbf{x}$, $Y(\mathbf{x})$ follows a standard Gaussian distribution $\mathcal{N}(0, 1)$, $\hat{I}$ has a volume fraction $\hat{\phi}$ given by

$$\hat{\phi} = \langle \hat{I} \rangle = \Pr[Y(\mathbf{x}) \geq \alpha] = \frac{1}{\sqrt{2\pi}} \int_{\alpha}^{\infty} e^{-t^2/2} dt. \quad (2.15)$$

It can also be shown that the two-point correlation function $\hat{S}_2$ of $\hat{I}$ is given by

$$\hat{S}_2(r) = \hat{\phi}^2 + \frac{1}{2\pi} \int_0^{g(r)} \frac{dt}{\sqrt{1-t^2}} \exp\left(-\frac{\alpha^2}{1+t}\right). \quad (2.16)$$

Therefore, the problem amounts to find a function $g(r)$ and a threshold α , such that $\hat{\phi} = \phi$ and $\hat{S}_2 = S_2$. Inverting (2.15) to find α is straightforward; inverting (2.16) typically requires to make *a priori* assumptions on $g(r)$: a parametrized function is chosen, and its parameters are obtained by minimizing a least-squares error functional that measure the discrepancy between S_2 and $\hat{S}_2$.

3. A 3D reconstruction can be generated by sampling the GRF $Y(\mathbf{x})$ using Equation (2.13) and thresholding it at α . This reconstruction will approximately have the same statistics (up to the second order) as the ground truth image.

This approach implicitly relies on the following key assumptions:

- **The porous medium is stationary and isotropic.** This is often true (at least at the length scales of interest), and can easily be verified empirically or statistically.
- **All the necessary information about the geometry of the medium is contained in the volume fraction and two-point correlation function.** Thresholding GRFs is a simple and convenient way to generate binary random fields, however, real porous media often present higher-order moments. In these cases, the assumption does not hold and the proposed method will likely fail to faithfully generate correct 3D samples.

2.2.2 Extending the class of admissible models

In the previous approach, the best fitting model was computed among the class of level-cut GRFs. As we have observed, this class is quite limited in the sense that its degrees of freedom are limited to the statistics of the underlying GRF. In a subsequent paper, [Rob97] proposes to extend the class of admissible porous media models. Two directions are taken:

Two-level cuts. A GRF $Y(\mathbf{x})$ can be thresholded at two different levels α and β , such that the binary model $I(\mathbf{x})$ is given by

$$I(\mathbf{x}) = \begin{cases} 1, & \text{if } \alpha \leq Y(\mathbf{x}) \leq \beta \text{ (pore phase),} \\ 0, & \text{otherwise (solid phase).} \end{cases} \quad (2.17)$$

Taking $\beta = +\infty$ reduces to the previously defined *one-level cut*. The volume fraction of I is given by $\langle I \rangle = (2\pi)^{-1/2} \int_{\alpha}^{\beta} e^{-t^2/2} dt$, and the two-point correlation function can be derived as an expression similar to Equation (2.16).

Intersection and Union sets. Let I_1 and I_2 denote two independent and statistically identical binary model obtained using the same two-level cut GRF scheme. Three model classes are defined:

Model N: $I_N = I$,

Model I: Intersection set of two statistically identical level-cut GRFs: $I_I = I_1 I_2$.

Model U: Union set of two statistically identical level-cut GRFs: $I_U = I_1 + I_2 - I_1 I_2$.

Obtaining the statistics ϕ and S_2 of these models from the statistics of I_1 and I_2 is straightforward.

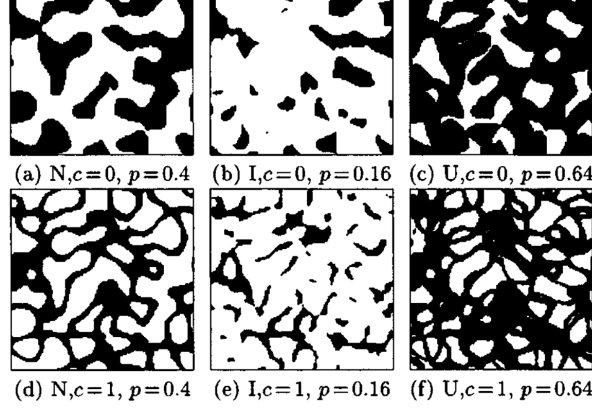


Figure 2.2: Examples of admissible porous media models proposed by [Rob97]. p corresponds to the volume fraction ϕ ; $c = 0$ corresponds to a one-level cut, and $c = 1$ to a symmetric two-level cut ($\alpha = -\beta$).

These extensions allow to generate a rich variety of more complex porous media, illustrated in fig. 2.2. The reconstruction workflow then consists in selecting a two-level cut configuration and one of the three classes N, I and U, and then generating a three-dimensional volume that shares the first- and second-order statistics with the ground truth. The results obtained by changing the initial configuration of the porous model are compared, typically by visual inspection.

2.2.3 Simulated annealing

The simulated annealing method was introduced by [YT98]. They propose to incorporate higher-order information into the reconstruction by optimizing a flexible objective function. First, a set of "reference" correlations functions $f_o^{(k)}(r)$ are computed from the 2D image, where k identifies the different correlation functions that are being used. In their paper, the authors choose to use the two-point correlation function and the lineal-path function:

$$f_o^{(1)}(r) = S_2(r), \quad f_o^{(2)}(r) = L(r). \quad (2.18)$$

Different or additional metrics may be defined, at the expense of a greater computational cost. The reconstruction process starts with an initial guess, taken as a 3D binary random model that has the correct volume fraction ϕ . We can now define $f_s^{(k)}(r)$, the corresponding correlation functions of the reconstructed sample. The sample will be changed iteratively to minimize the energy function

$$E = \sum_k \sum_i \left(f_s^{(k)}(r_i) - f_o^{(k)}(r_i) \right)^2. \quad (2.19)$$

At each iteration, two arbitrarily selected voxels of different phases are swapped. Note that the volume fraction is conveniently preserved. We can calculate the energy E' of the new state and the energy difference $\Delta E = E' - E$. According to the simulated annealing method [LA87], the phase swap is accepted with probability

$$p(\Delta E) = \begin{cases} 1, & \text{if } \Delta E \leq 0, \\ e^{-\Delta E/T}, & \text{if } \Delta E > 0; \end{cases} \quad (2.20)$$

where T is a "temperature". If it causes a decrease in energy, the swap is automatically accepted; otherwise it is accepted with a probability that decreases as $|\Delta E|$ increases. The *cooling schedule* governs the rate of change of T , and is chosen to allow the system to evolve to the desired

state as quickly as possible without getting trapped in a local energy minimum. Computing the correlations functions of the new state at every iteration is very costly; instead, the correlations are estimated by only looking in the three principal directions. By doing so, the new energy E' can be computed from E only by scanning the principal directions that emanate from the two swapped voxels; this results in a computational complexity of $\mathcal{O}(n)$ at every iteration, where n is the maximum number of elements along a dimension.

The simulated annealing method achieves better reconstruction results (in terms of metrics such as macroscopic properties) than the methods based on Gaussian random fields, thanks to the incorporation of more powerful morphological descriptors such as the lineal-path function. However, a drawback of the approach is that the reconstruction provides only a single 3D sample, whereas the previous methods learn a distribution from which one can sample a variety of porous materials.

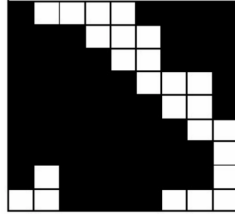
2.2.4 Multiple-point statistics

The methods introduced before, i.e. based on Gaussian random fields and simulated annealing, are limited to the reproduction of low-order statistics inferred from the available 2D images. However, identification of low-order statistics is not sufficient to reproduce connected long-range geometry or complex patterns that typically arise in low-porosity media. A more realistic reconstruction can be obtained by using structural information beyond the two-point correlation. In 2004, Okabe and Blunt [OB04] proposed to generate patterns with shapes observed in a reference image and then distribute these patterns over a synthetic 3D model.

Let us define a *template* T as a set of vectors $\mathbf{r}_1, \dots, \mathbf{r}_n$. The training image is scanned by moving the template and collecting the pattern $p(\mathbf{x})$ defined at each location $\mathbf{x}$, i.e.,

$$p(\mathbf{x}) = \{I(\mathbf{x}); I(\mathbf{x} + \mathbf{r}_j), j = 1, \dots, n\}, \quad (2.21)$$

where $I(\mathbf{x})$ is the phase at location $\mathbf{x}$. An example pattern measured by a 9×9 template:



The set of all scanned patterns results in a *training set*

$$S = \{p(\mathbf{x}_i), i = 1, \dots, N\}, \quad (2.22)$$

where N is the number of central locations at which the template is applied.

Let $c(p)$ denote the number of occurrences of pattern p in the image. The probability of this pattern appearing in the reconstructed porous medium can be inferred as

$$\Pr[p] \approx \frac{c(p)}{N}. \quad (2.23)$$

The algorithm starts from a volume where each voxel is unspecified, and fills it iteratively by visiting a new voxel and sampling its *conditional probability distribution*:

$$\Pr[I(\mathbf{x}) = b \mid p(\mathbf{x})] = \frac{\Pr[I(\mathbf{x}) = b, p(\mathbf{x})]}{\Pr[p(\mathbf{x})]}, \quad b = 0, 1, \quad (2.24)$$

	[Qui84]	[Rob97]	[YT98]	[OB04]	Ours
Learns distribution?	Yes	Yes	No	No	Yes
Slice present?	No	No	No	Yes	No
Multi-scale?	No	No	No	Yes	Yes
Manual intervention?	No	Yes	No	No	No

Table 2.1: Classification of the various reconstruction methods.

where $\Pr[I(\mathbf{x}) = b \mid p(\mathbf{x})]$ is the probability that the new voxel $\mathbf{x}$ has phase b , given that its neighborhood has pattern $p(\mathbf{x})$. Notice that this pattern can be incomplete (some voxels are unspecified): then we simply consider patterns that are equal in terms of their specified voxels. When the conditional probability is obtained, the new voxel is sampled, and its value is used as data for the subsequent voxel assignments. If the pattern is too sparse to infer a correct probability (especially at the beginning), the size of the pattern is reduced.

With this method, it is possible to force the training image to be present as some slice in the reconstructed volume, by simply initializing the algorithm with the image as data in some slice of the volume. This aspect may be useful in some scenarios, e.g. if we dispose of images of several parallel slices, to enforce the consistency of the output.

In order to reproduce large-scale structures, a large template is necessary. In theory, the template size can be expanded to match the largest structure in the training image. In practice, the template size is limited by memory limitations. Therefore the authors propose a *multi-scale* extension that is able to capture long-range correlations. The training image is scanned at several scales by using coarser templates, simply expanded from the small-scaled samples. A reconstruction is first performed on the coarsest grid; once the coarse simulation is finished, the voxel values are placed on the finer grid and used as conditioning data for the next scale. This multi-scale approach allows to capture large-scale multiple-point statistics effectively while requiring little memory and computations.

The multi-scale aspect, as well as the ability to capture complex statistics about the porous medium, has made the multiple-point statistics approach the state of the art in porous media reconstruction for a long time.

2.2.5 Summary

We conclude this historical review by defining categories for the presented methods; the classification is done by answering the following questions:

- Does the method learn a distribution of the three-dimensional porous material?
- Is it possible to force the training image to be present as a slice?
- Does the method follow a multi-scale approach?
- Is there some manual intervention needed?

To help the reader situate the new method presented in this work, we have also classified it. A summary of the classification is given in table 2.1.

2.3 Related work

We want to mention other important works which do not directly tackle the reconstruction from 2D images, but are still connected to our work.

Reconstruction from 3D samples In [MDB17], a 3D representation of a porous sample is provided, and the goal is to generate new 3D samples which are different from the training image but respect the macroscopic properties. The motivation is to assess the *variability* of the material properties, which is difficult to do experimentally. Of course their task is "easier" since all macroscopic properties can be directly inferred from the provided 3D sample. They follow a generative adversarial network (GAN) approach, with a generator that is conceptually similar to ours, and a discriminator that is trained to detect real and fake 3D samples.

3D reconstruction with forced 2D slice In [Vol+19], the work of [MDB17] is extended with the possibility to provide (additionally to the 3D sample) a 2D image that is forced to be present in the reconstructed samples. This is done by encoding the 2D image in a latent space with a trained encoder, feeding the encoded vector to the generator, and training the generator to look like the provided 3D sample, in the same adversarial fashion as [MDB17]. The 2D and 3D training images must have the same resolution, which prevents the use of high-resolution images for which 3D samples cannot be imaged.

Direct estimation of permeability from 2D images In [Ara+20], direct deep learning permeability estimation from 2D images is done. A deep convolutional neural network is trained on a dataset of 2D images labeled with the correct permeability, in order to predict the permeability of new images. The results they obtain show that in most cases, permeability can be correctly inferred from 2D images only, which at least indicates that trying to reproduce 3D volumes that preserve the permeability is not an impossible task.

Chapter 3

Deep Texture Synthesis for Porous Media Reconstruction

3.1 Introduction

This section presents the deep learning reconstruction method, which is a slight adaptation from [Gut+20], with some extensions.

Figure 3.1 outlines the proposed method. Solid texture synthesis is performed by a deep convolutional neural *generator* network $\mathcal{G}$ with learnable parameters θ . The generator takes a set of multi-scale volumetric noise inputs $Z = \{z_0, \dots, z_K\}$ and processes it to produce a *binary* solid texture $v = \mathcal{G}(Z|\theta)$. Note that the noise inputs Z and the solid texture v are, in principle, infinite; the size of Z is therefore arbitrary and proportional to the size of the output v .

To specify the desired appearance of the samples v , a set of $D \in \{1, 2, 3\}$ *example images* $\{u_1, \dots, u_D\}$ is provided, each corresponding to a principal direction of the Cartesian grid (for example, if we wish to synthesize an isotropic solid, we will set $D = 3$ and $u_1 = u_2 = u_3$; in specific scenarios, one may wish to learn only one or two directions, by setting $D = 1$ or 2). To train the generator, the generator volume v is cut into 2D slices and every slice $v_{d,n}$ (n -th slice along the d -th direction) is compared against its corresponding example u_d .

To perform the comparison, we resort to a *pretrained* deep convolutional neural *descriptor* network $\mathcal{D}$. The descriptor is fed with 2D images and outputs their perceptual features which are taken as the activations of the hidden layers of the network. Comparing the features of the slice $v_{d,n}$ against the features of the example u_d allows to formulate a *slice-based loss function* $\mathcal{L}(v|\{u_1, \dots, u_D\})$.

Texture networks The approach of training a feed-forward generator network using a complex and expressive loss function was pioneered by Ulyanov et al. in 2016 [Uly+16]. This important paper was tackling two tasks:

1. Generate multiple samples of the same texture of arbitrary size,
2. Transfer an artistic style from a given image to any other image.

The resulting generator network is light-weight and can generate textures of remarkable quality. Contrary to approaches based on iteratively optimizing the synthesized texture (e.g., [GEB15]), here the generator actually learns a *distribution* of the texture, and after training it can produce new texture samples very quickly.

In 2019, Gutierrez et al. [Gut+20] proposed to extend the solution of Ulyanov et al. to the *solid texture synthesis* task, that seeks to synthesize 3D volumes that are visually similar to the

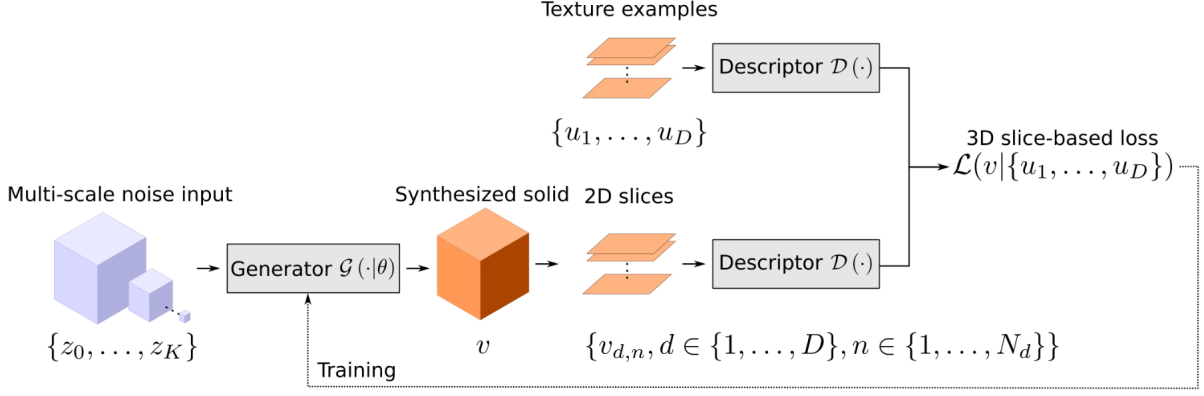


Figure 3.1: Training framework for the Generator network $\mathcal{G}(\cdot|\theta)$ with parameters θ . The Generator is fed with a set of noise inputs $z_0, \dots, z_K$ at different scales and produces a solid texture v . The pretrained Descriptor network $\mathcal{D}$ is fed with two-dimensional images and extracts their perceptual features. The loss $\mathcal{L}$ compares, for every direction d , the features of the example u_d with those of slices taken from the volume in that direction. The loss’ gradient is backpropagated to the network to iteratively update the parameters θ .

provided texture example. The proposed generator network is the three-dimensional extension of the texture network of Ulyanov. The training is done by taking slices from the generated volume and comparing them to the texture example. To the best of our knowledge, Gutierrez et al. are the first and only to have leveraged the (not yet fully understood) power of deep convolutional neural networks for the solid texture synthesis task.

Motivation of texture networks for porous media reconstruction Obviously, the goal of texture networks is not to reconstruct porous media from 2D images and they have never been used for that purpose. As stated in the introduction of this work, the fundamental goal of porous media reconstruction is that the reconstructed medium matches its macroscopic properties with the original medium from which the image was taken. Therefore, using solid texture synthesis for the reconstruction task amounts to make the following assumption:

Given a porous medium f , a slice g of f , and metrics $F_1, \dots, F_M$. If the reconstructed medium $\hat{f}$ is such that every slice of $\hat{f}$ looks like g (according to the provided descriptor), then the metrics are preserved: $F_m(\hat{f}) = F_m(f)$ for every m .

We believe that the descriptor proposed by Ulyanov, VGG-19, is powerful enough for this claim to hold. This belief is supported by the remarkable visual results achieved by texture networks, and transfer learning in general. Moreover, the multi-scale generation and description involved in the method makes it suitable to handle the inherently multi-scale nature of porous media (see, e.g., [Blu17]); it also allows to train the generator using several images at different scales. Finally, as we will see, representing a porous medium in a multi-scale latent space opens up interesting research directions.

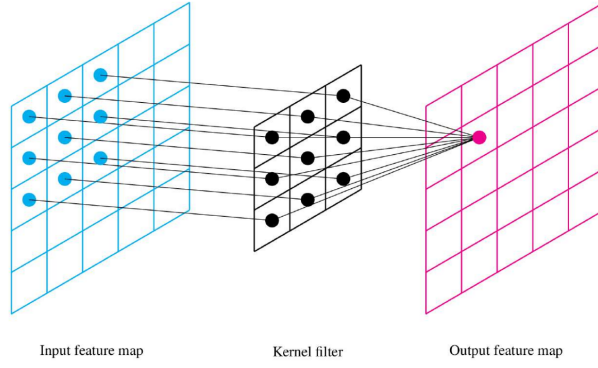


Figure 3.2: Schematic representation of the convolution operation, where a feature map is filtered by a 3×3 kernel.

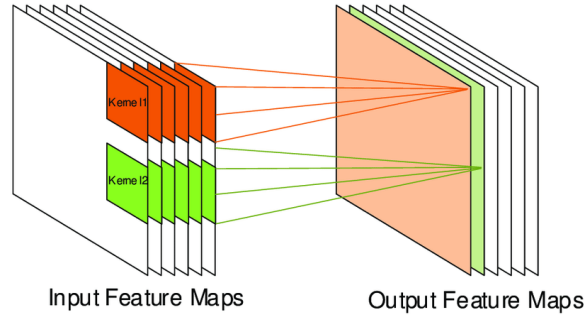


Figure 3.3: Schematic representation of a convolutional layer, where a set of feature maps are filtered by a set of convolution operations. There are $N_{\text{in}} = 5$ input feature maps and $N_{\text{out}} = 5$ output feature maps.

3.2 Description of the method

The goal of this section is to describe in detail how the solution works.

3.2.1 Prerequisites

Convolution operation The fundamental building block of Convolutional Neural Networks (CNNs) is the *convolution* operation, illustrated in fig. 3.2. Given a *kernel* w and an input *feature map* x , the kernel is slid along x and at every step, the scalar product is computed between the kernel w and the corresponding subset of x . This forms the output feature map $y = w * x$. An important property of this operation is that it is *shift-invariant*: the filter performs the same *local* operation on the image. This property makes it particularly suitable for treating images, because individual pixels are, in principle, not correlated over long distances. A convolution is parametrized only by the weights of the kernel, e.g., for a 3×3 kernel, only 9 real numbers need to be provided.

Convolutional layer In order to process different features of an image, it is described by multiple *feature maps* (or *channels*). Processing a set of feature maps into another set of feature maps is done by a *convolutional layer*, which builds on the convolution operator (fig. 3.3). Let x_j denote the j -th input feature map, y_i the i -th output feature map, b_i a scalar *bias* term, and

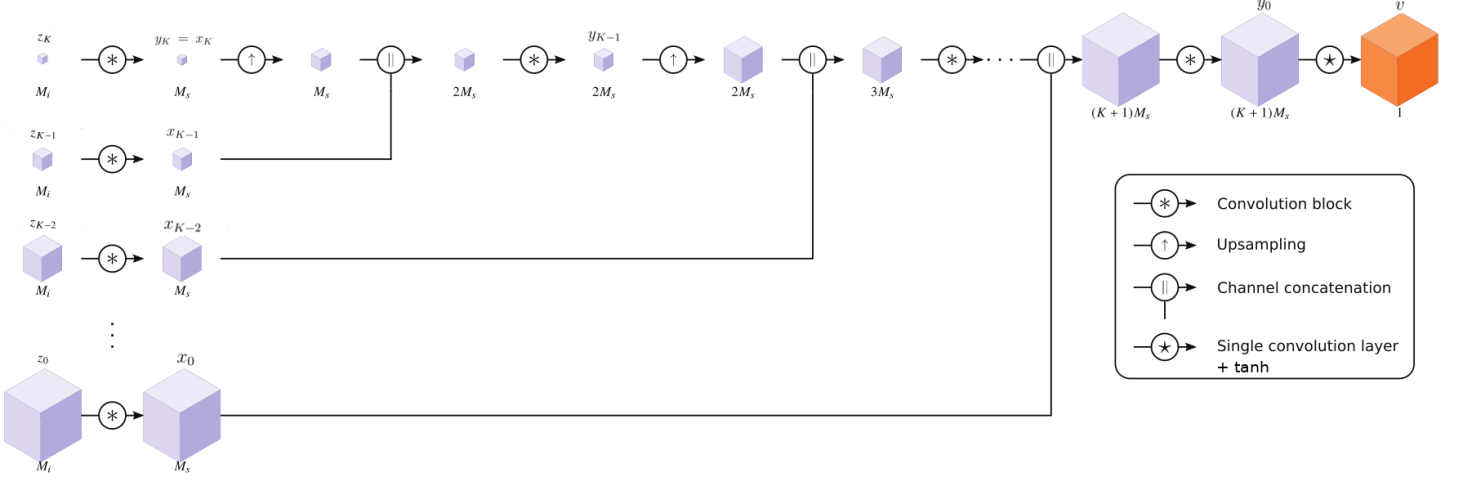


Figure 3.4: Schematic of the network architecture. Noise input $Z = \{z_0, \dots, z_K\}$ is processed at $K + 1$ different scales using convolutions and non-linear activations. After a convolution block, the cube is upsampled and channel-wise concatenated with the next scale. For each cube the number of channels is indicated below. M_i and M_s are tunable parameters controlling the number of input channels and channels at intermediate layers, respectively. Adapted from [Gut+20].

w_{ij} an $n \times n$ kernel. Then the convolutional layer is defined by

$$y_i = b_i + \sum_{j=1}^{N_{\text{in}}} w_{ij} * x_j, \quad i = 1, \dots, N_{\text{out}}, \quad (3.1)$$

where N_{in} and N_{out} are the number of input and output feature maps, respectively. Note that this operation can be seen as a matrix-vector product, where entries of the vector are feature maps, and entries of the matrix are kernels. With this in mind, one easily finds that a convolutional layer has $N_{\text{out}} + n^2 N_{\text{in}} N_{\text{out}}$ parameters, independently of the size of the feature maps, which is typically much larger than the hyperparameters N_{in} , N_{out} or n . The compact representation of convolutional layers is an asset for both the learning phase as well as the generalization property of the convolutional network: indeed, a network with too many parameters, trained for a sufficiently long time, will tend to overfit the data and not generalize well.

3.2.2 The generator network

Now that we have introduced the concept of convolutional layer, we can build a convolutional generator network.

The general idea is to start from a coarse (low-resolution) representation of the texture, and gradually incorporate finer information. An overview of the network architecture is depicted in fig. 3.4. For now, consider that volumes are infinite. We start from a set of volumetric noise inputs $z_0, \dots, z_K$, where 0 is the finest scale and K is the coarsest scale (typically, we work with 6 scales, i.e. $K = 5$). Every noise input is first passed through a *convolution block* which performs a series of operations that will be detailed later. Let $C_k^{\text{init}}(\cdot)$ denote the convolution block at scale k , and let $x_k = C_k^{\text{init}}(z_k)$ be the result of these blocks on the noise inputs. Notice that z_k is composed of M_i feature maps (or channels) and x_k of M_s feature maps, where M_i and M_s are hyperparameters. The feature maps of x_k can be interpreted as a representation of the texture at the k -th scale.

Starting from $y_K = x_K$, the volume is successively refined, first by *upsampling*, then incorporating new information by *channel concatenation*, and passing through a convolution

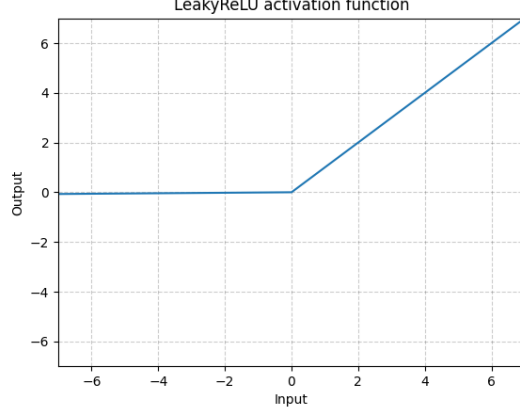


Figure 3.5: Plot of the LeakyReLU activation function.

block:

$$y_k = C_k^{\text{inter}}(\text{concat}(x_k, \text{upsample}(y_{k+1}))). \quad (3.2)$$

The upsampling step doubles the resolution by replicating every voxel 8 times. Channel concatenation simply consists in putting together the feature maps of two volumes; the number of feature maps is thereby increased by M_s at every concatenation. Applying the intermediate convolution block C_k^{inter} may be interpreted as *combining* the coarser information from y_{k+1} with the finer information from x_k , into a new set of feature maps.

After K iterations of eq. (3.2), we have a set of $(K + 1)M_s$ feature maps, which are finally merged into a single volume v using a convolutional layer, and a hyperbolic tangent activation $\tanh(\cdot)$ is applied in order to constrain the network to generate a binary volume, representing the two-phase porous medium:

$$v = \tanh(C^{\text{final}}(y_0)). \quad (3.3)$$

Let us now define the *convolution block* more precisely. This block consists of three convolutional layers; the size of the convolution kernels is $3 \times 3 \times 3$ for the first two convolutions, and $1 \times 1 \times 1$ for the last one. Each of the three layers is followed by a leaky rectified linear unit function (*Leaky ReLU*):

$$\text{LeakyReLU}(x) = \begin{cases} x & \text{if } x \geq 0, \\ 0.01x & \text{otherwise.} \end{cases} \quad (3.4)$$

This nonlinear function, plotted on fig. 3.5, is called an *activation function* and introduces nonlinearity in the network (indeed, if there was no activation at all, the network would merely be a large linear function). This choice has the advantage (over, e.g., ReLU) that its gradient is nonzero regardless of the input value, which is valuable during the training phase.

After every convolutional layer, as well as before every channel concatenation, *batch normalization* is performed for every feature map x_i :

$$y_i = \beta_i + \gamma_i \frac{x_i - \mathbb{E}[x_i]}{\sqrt{\text{Var}[x_i] + \epsilon}}, \quad (3.5)$$

where the mean and variance are computed over the batch and volume; β_i and γ_i are learnable parameters. This operation was shown to significantly improve the stability of the training [IS15].

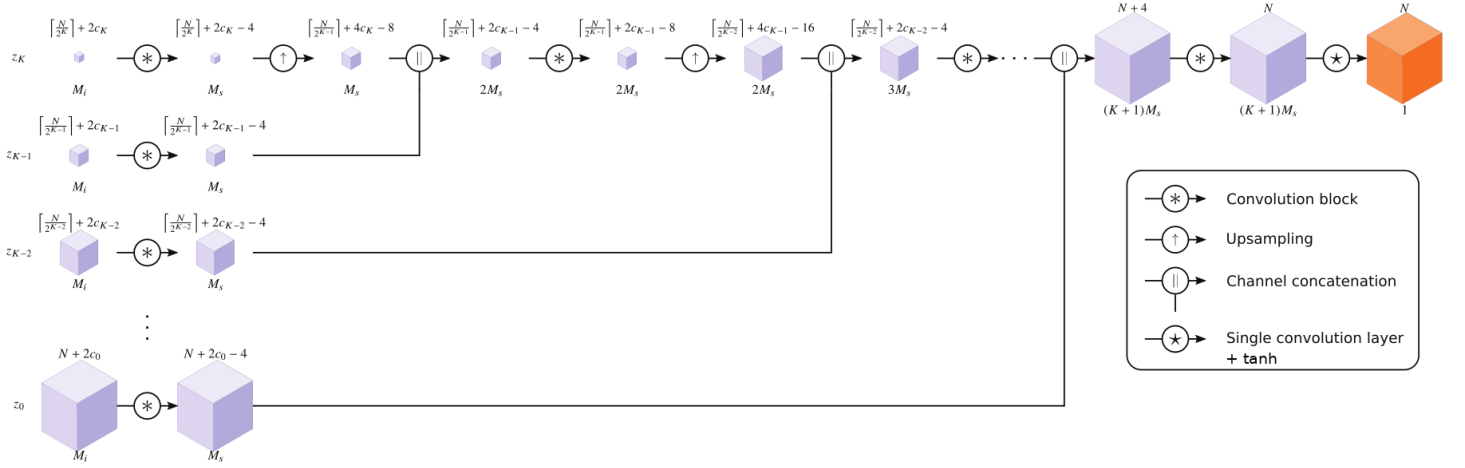


Figure 3.6: Schematic of the network architecture, where the size of each cube is indicated above. It is assumed that all three dimensions have equal sizes. Adapted from [Gut+20]

Treatment of finite volumes Up to now we have assumed that the volumes passing through the network were infinite. In fig. 3.6 we have put the generator network again, but with an indication of the size of each cube. The target size for the output volume v is (N, N, N) . Convolutions are performed without padding; therefore every convolution block reduces each dimension by 4 voxels. An upsampling block doubles the size in every dimension. The additional values c_k allow to compensate the size reduction caused by the convolutions. At scale $k = 0$ we have $c_0 = 4$ and it can be shown that

$$c_k = \left\lceil \frac{c_{k-1} - 2}{2} \right\rceil + 4. \quad (3.6)$$

Notice that the output of the concatenation block always has even sizes, while the volume to be concatenated with, x_k , may have odd sizes; if it is the case we need to reduce the size of the upsampled volume by 1. There are two available choices for this reduction; we will see that this choice is important in the training phase.

Performing convolutions without padding may seem like an unnecessary complication. However, they allow to generate multiple volumes that can be tiled together seamlessly: by generating large noise volumes, and cutting them into several pieces (with appropriate overlap), processing the noise pieces separately produces tileable output pieces. This is particularly useful if we want to generate a very large volume and if the generator memory is limited.

Generating periodic volumes As we will see in chapter 4, it is useful to have a periodic porous sample, i.e. that can be tiled seamlessly with itself along every dimension, in order to represent an infinite medium using only a finite sample. The generator allows to construct periodic textures in a natural way. For simplicity, consider a one-dimensional version of the network that takes as input infinite 1D signals z_k^l (k is the scale and l is the channel), and that outputs an infinite output signal v . We will show that if the input is periodic at each scale (with a period corresponding to the scale), then the output is periodic, too:

$$\text{if } z_k^l[i + 2^{K-k}N] = z_k^l[i] \quad \forall k \in \{0, \dots, K\}, l \in \{1, \dots, M_i\}, i \in \mathbb{Z}, \quad (3.7)$$

$$\text{then } v[i + 2^K N] = v[i]. \quad (3.8)$$

At the coarsest scale K , input signals have an arbitrary period N , and the period doubles at each refinement, until reaching a final period of $2^K N$. Clearly the convolution and batch

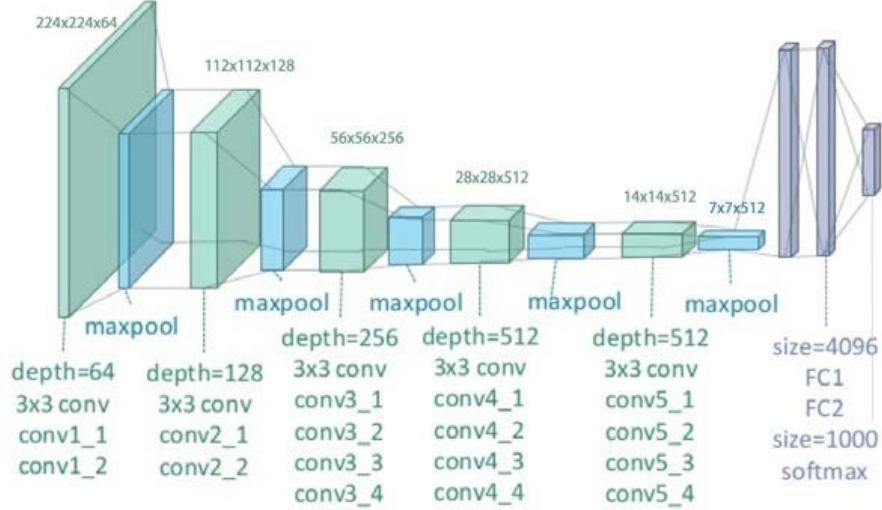


Figure 3.7: Schematic overview of the VGG-19 network for the ILSVRC classification task. The network takes as input a 224^2 color image and outputs a vector assigning a probability to each of the 1000 possible classes. It consists of 17 convolutional layers and 2 fully connected (FC) layers. From [ZYM18].

normalization operations preserve the periodicity as long as every input channel has the same period. Upsampling blocks perform nearest neighbor upsampling with a scaling factor of 2; therefore the period is doubled when signals are upsampled. This explains why we want the period of the inputs to double at each scale: to ensure that all channels that are concatenated have the same period. Notice that due to the upsampling method, it is only possible to generate an output with a period that is a multiple of 2^K .

In practice, inputs are finite and hence we only keep the truncation of signals to their period, that is $z_k^l[i]$ for all $i \in \{1, \dots, 2^{K-k}N\}$. Before each convolution, *circular padding* is performed in order to account for the periodicity. Notice that there is no more reduction in the size of the signals, hence we do not need to compensate: $c_0 = \dots = c_K = 0$.

The argument can easily be extended to three dimensions.

3.2.3 Training

The parameters involved in the convolutional layers as well as the batch-normalization layers need to be optimized so that the generator can synthesize volumes that are perceptually similar to the provided 2D images $\{u_1, \dots, u_D\}$. This optimization requires an objective function (also called *loss function*), denoted $\mathcal{L}(v|u_1, \dots, u_D)$, to be minimized. Before defining this loss function, we first introduce the *descriptor network* $\mathcal{D}$.

Descriptor network The descriptor network $\mathcal{D}$ is a convolutional neural network acting on 2D images. We apply the ideas of *transfer learning*, which consists in exploiting the knowledge gained from solving one problem (the *source task*) to solve another problem (the *target task*) (see, e.g., [Oqu+14] for an example of successful application of transfer learning). Here, the *source task* is the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) and a particular solution that was trained on this task is the VGG-19 network developed by [SZ15], depicted on fig. 3.7. The network is composed of 19 trainable layers in total (hence the name), of which 17 are convolutional layers ($\text{conv}k_l$), and 2 fully connected layers that act on the final feature maps to output a vector assigning a probability to each of the 1000 possible classes. VGG stands for Visual Geometry Group, the research group that has developed the solution. The network

was trained for classification on ImageNet, a large database of millions of annotated images. On the other hand, the *target task* is to extract a compact representation of the perceptual information from a two-dimensional texture (or, more specifically, porous medium). To do so, we first remove the fully connected layers, which removes the constraint on the input image size. Let $x \in \mathbb{R}^{N_1 \times N_2 \times 3}$ denote the (color) image we wish to describe. We select a set L of internal layers of VGG-19 from which the information will be extracted. The feature maps corresponding to these layers form the set

$$\left\{ F^l(x) \in \mathbb{R}^{N^l \times M^l} \right\}_{l \in L}, \quad (3.9)$$

where N^l is the number of spatial values and M^l the number of channels for layer l . Now we wish to derive a representation of these feature maps whose size does not depend on the size of the feature maps. This representation is obtained by computing, for every layer l , its Gram matrix $G^l \in \mathbb{R}^{M^l \times M^l}$:

$$G^l(x) = \frac{1}{N^l} F^l(x)^T F^l(x). \quad (3.10)$$

This matrix encodes first and second order information about the distribution of the features, by averaging their statistics over the pixels. To conclude, the descriptor $\mathcal{D}$ is a sophisticated function that takes an image x and extracts its perceptual features as a set of Gram matrices:

$$\mathcal{D}(x) = \left\{ G^l(x) \in \mathbb{R}^{M^l \times M^l} \right\}_{l \in L}. \quad (3.11)$$

Slice-based loss function Now that we have formally defined the descriptor $\mathcal{D}(\cdot)$, we can build a loss function that will compare the slices of the generated 3D volume v with the provided images $u_1, \dots, u_D$. Since the output of the generator is a single-channel volume, we first need to replicate it 3 times to obtain a "color volume" $v \in \mathbb{R}^{N_1 \times N_2 \times N_3 \times 3}$. Let $v_{d,n}$ be the n -th slice of v taken in direction d . Using the descriptor network, we can quantify the discrepancy between $v_{d,n}$ and the example image along the same dimension, by comparing their Gram matrices:

$$\mathcal{L}_2(v_{d,n}, u_d) = \sum_{l \in L} \frac{1}{(M^l)^2} \left\| G^l(v_{d,n}) - G^l(u_d) \right\|_F^2, \quad (3.12)$$

where $\|\cdot\|_F$ denotes the Frobenius norm. Notice that $v_{d,n}$ and u_d need not have the same dimensions to perform the comparison. Building on this discrepancy measure, we can define the complete loss function $\mathcal{L}$ that compares every slice of the volume to its corresponding example:

$$\mathcal{L}(v|u_1, \dots, u_D) = \sum_{d=1}^D \frac{1}{N_d} \sum_{n=1}^{N_d} \mathcal{L}_2(v_{d,n}, u_d). \quad (3.13)$$

Given parameters θ , the loss $\mathcal{L}(\mathcal{G}(Z|\theta)|u)$ is a stochastic objective since Z is random. Formally, we wish to find parameters that minimize the expectation of this loss, given the examples $u_1, \dots, u_D$:

$$\theta^* = \operatorname{argmin}_{\theta} \mathbb{E}_Z [\mathcal{L}(\mathcal{G}(Z|\theta) | u)], \quad (3.14)$$

where Z is a multi-scale noise composed of voxels that are independent and identically sampled from a uniform distribution $\mathcal{U}(0, 1)$. The optimization is done by a *stochastic gradient descent* algorithm:

$$\theta^{k+1} = \theta^k - \alpha \frac{\partial \mathcal{L}(\mathcal{G}(z|\theta^k))}{\partial \theta^k}, \quad (3.15)$$

where z is a realization of the noise Z , and α is called the *learning rate*. In practice, this learning rate is not constant but is adjusted during the optimization by the Adam algorithm [KB17]. The gradient in eq. (3.15) can be computed efficiently with the well-known backpropagation algorithm [Hec92]; in practice, it has the same computational cost as the forward pass, i.e. computing $\mathcal{G}$ and $\mathcal{L}$.

Training with a single slice Deep learning is done more efficiently on a Graphics Processing Units (GPU). A limitation of these devices is their memory capacity, which is about a few gigabytes. One should therefore be careful with 3D models since they quickly grow up in size. As an illustration, training the network with generated volumes of size 256^3 would require about 24 GB of GPU memory, which is not available on most GPUs. To alleviate this limitation and increase the training resolution, a training scheme is proposed that only generates a single slice at every iteration.

A first simplification relies on the following observation. The coarsest noise input, z_K , is statistically invariant under translation (in other words, it is stationary). This remains true after passing through the convolution block, i.e. for $y_K = x_K = C_K^{\text{init}}(z_K)$. However, due to the upsampling, y_{K-1} is only invariant under translations of multiples of 2. Reusing the argument for the successive scales, we find that the final volume v is invariant under translations of multiples of 2^K . Therefore, for each dimension, generating any subset of 2^K *contiguous* slices is enough to compute a discrepancy measure that is representative of the whole sample. Formally,

$$\mathbb{E}_Z [\mathcal{L}(\mathcal{G}(Z|\theta) \mid u)] = \sum_{d=1}^D \frac{1}{N_d} \sum_{n=1}^{N_d} \mathbb{E}_Z [\mathcal{L}_2(\mathcal{G}(Z|\theta)_{d,n}, u_d)] \quad (3.16)$$

$$= \sum_{d=1}^D \frac{1}{2^K} \sum_{q=1}^{2^K} \mathbb{E}_Z [\mathcal{L}_2(\mathcal{G}(Z|\theta)_{d,q}, u_d)]. \quad (3.17)$$

In practice, $2^K \ll N_d$, hence this trick greatly reduces the memory requirements.

We can go even further and perform the training with a single slice, by stochastically approximating the inner sum in eq. (3.17). Let Q_d be a random variable uniformly drawn from $\{1, \dots, 2^K\}$. Then,

$$\mathbb{E}_{Z, Q_d} [\mathcal{L}_2(\mathcal{G}(Z|\theta)_{d, Q_d}, u_d)] = \frac{1}{2^K} \sum_{q=1}^{2^K} \mathbb{E}_Z [\mathcal{L}_2(\mathcal{G}(Z|\theta)_{d,q}, u_d)], \quad (3.18)$$

and therefore,

$$\mathbb{E}_Z [\mathcal{L}(\mathcal{G}(Z|\theta) \mid u)] = \sum_{d=1}^D \mathbb{E}_{Z, Q_d} [\mathcal{L}_2(\mathcal{G}(Z|\theta)_{d, Q_d}, u_d)], \quad (3.19)$$

which means that we can train the generator with one slice for each principal direction only. Note that the whole network is impacted at every iteration since the convolution weights are shared by all slices. The proposed scheme significantly reduces the amount of GPU memory needed for training the generator network, and allows to increase the resolution up to 1024 voxels per dimension.

When training the network with a single slice, we should be careful to ensure that the coordinate of the output slice (in the direction orthogonal to the slice) is indeed uniformly drawn from $\{1, \dots, 2^K\}$. This coordinate is actually controlled by the dimension reduction during concatenation. As stated before, the dimension of the upsampled volume is reduced by 1 if the volume to be concatenated with has odd dimensions; optionally, a shift $s_k = \{0, 1\}$ may be applied before concatenation. If we choose $s_k = 0$ systematically, we are actually always generating the same slice, and therefore the whole volume is not trained. Instead, if we choose the shift s_k randomly at every concatenation, we are randomly drawing a slice coordinate from $\{1, \dots, 2^K\}$, and the whole volume is correctly trained. Notice that there are 2^K shift combinations, each one corresponding to one of the 2^K slice coordinates.

Training resolution	Training time
128 ²	1.4 h
256 ²	5 h
512 ²	17.5 h

Table 3.1: Training time as a function of the resolution of the generated slices, using a single GPU Nvidia GeForce GTX 1080 Ti.

3.3 Results

In this section we describe the setup for testing the reconstruction solution, and we show the results for a set of three distinct porous materials.

3.3.1 Experimental settings

All results in this section were obtained using the following settings.

Generator network Following [Gut+20], we set the number of scales to 6, i.e. $K = 5$, and we use, for each convolution block, $M_i = 3$ input channels and $M_s = 4$ output channels. The resulting network is quite compact with about 8.5×10^4 parameters, which is usually much smaller than the size of the volumes we wish to generate.

Descriptor network As mentioned before, a truncated VGG-19 is used as a descriptor network. The considered set of layers L for the loss is (referring to fig. 3.7)

$$L = \{\text{conv1_1}, \text{conv2_1}, \text{conv3_1}, \text{conv4_1}, \text{conv5_1}\}. \quad (3.20)$$

Training The method is implemented using *PyTorch* and the parameters for the descriptor network were fetched from the Visual Geometry Group. The parameters of the generator network are optimized using the Adam algorithm [KB17] with a learning rate of 0.1 for 3000 iterations. At each iteration, a batch of 8 samples per slicing direction (24 samples in total) is used. The generated slices have dimensions 256^2 . Even if PyTorch allows to work with batches naturally, we compute the gradient individually for each sample; this slows down the training process but allows to concentrate the available memory on the resolution of the samples. The required training time as a function of the resolution of the training slice is given in table 3.1. We see that the training time is roughly proportional to the size of the training slice.

Synthesis For each porous model, a set of 10 three-dimensional samples of size 320^3 is generated in a fully periodic fashion by the trained network. The sampling phase is very fast (a few seconds per sample) since only a single forward pass is needed.

3.3.2 Experimental data

We now introduce the three models on which we will test the reconstruction procedure.

Model blobs The first model is a truncated Gaussian random field (GRF) inspired from the PoreSpy toolbox [Gos+19]. The GRF $Y(\mathbf{x})$ is obtained by convoluting a Gaussian kernel G with a Gaussian white noise random field W on $\mathbb{R}^d$:

$$Y(\mathbf{x}) = (G * W)(\mathbf{x}). \quad (3.21)$$

Note that this operation amounts to generate a GRF with a Gaussian covariance function. The spatial resolution can be adjusted by changing the width of the Gaussian kernel G . Once we have a (discretized) realization of Y , we can make it binary by setting an appropriate threshold that will achieve the desired volume fraction ϕ (here we choose $\phi = 0.5$).

For this model, we have generated a 2D (512^2) image (fig. 3.8a) that will serve as training image in the three principal directions, as well as a 3D (320^3) volume (fig. 3.9a) that will be used for a visual comparison of the results, and for validation in chapter 4. Note that the training image and the volume are generated with the same Gaussian kernel, but the image is not a slice that is physically present in the volume.

Model beadpack Models based on truncated Gaussian random fields are easy to generate and analyze, and they have been successfully reconstructed since decades. A much more challenging model is the packing of non-overlapping spheres (or beads), which is closer to real porous rocks that are composed of *grains*.

The **beadpack** model is based on a real packing of equally sized spheres in a disordered close packing. The 3D image, publicly made available by [MDB17], consists of 500^3 voxels; the size of an individual sphere is 50 voxels. A slice of this volume is displayed in fig. 3.8b and will serve as training image in the three principal directions. A 320^3 subset of the volume (fig. 3.10a) will be used for visual comparison.

Model berea Finally, a real rock sample is considered. *Berea sandstone* is a fluvial sandstone composed of angular grains bonded by clays. Again we have collected a 3D image made available by [MDB17]. The image represents a sample of 1.2 mm^3 and has dimensions 400^3 . Contrary to the two previous models, we have no theoretical guarantee about the isotropy of the sample. Therefore, to train the generator, we extract three orthogonal slices aligned with the principal directions; these slices are displayed in fig. 3.8c. A 320^3 subset of the volume (fig. 3.11a) will be used for visual comparison.

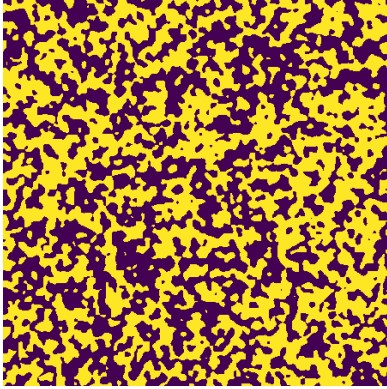
3.3.3 Experimental results

For each of the three models, we have trained the generator network using the training images in fig. 3.8. In figs. 3.9 to 3.11, we compare side by side the ground truth (3D) sample with one of the generated samples, as well as one of their slices.

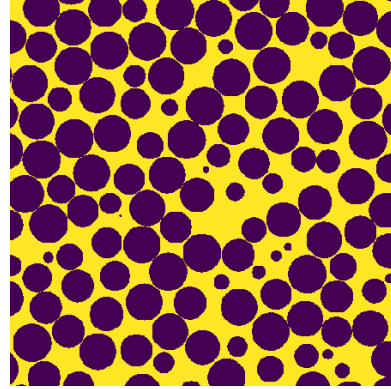
For the **blobs** model (fig. 3.9), the reconstruction is quite successful: it is indeed hard to distinguish the synthetic from the ground truth 3D sample. If we look at the slices, we can observe that blobs are slightly thinner for the synthetic slice, but the difference is minimal and not necessarily representative of the whole sample.

For the **beadpack** model (fig. 3.10), the network is able to reconstruct a packing of grains. They are not exactly spherical (more like ellipsoids), but their average size is quite similar to the size of the original spheres.

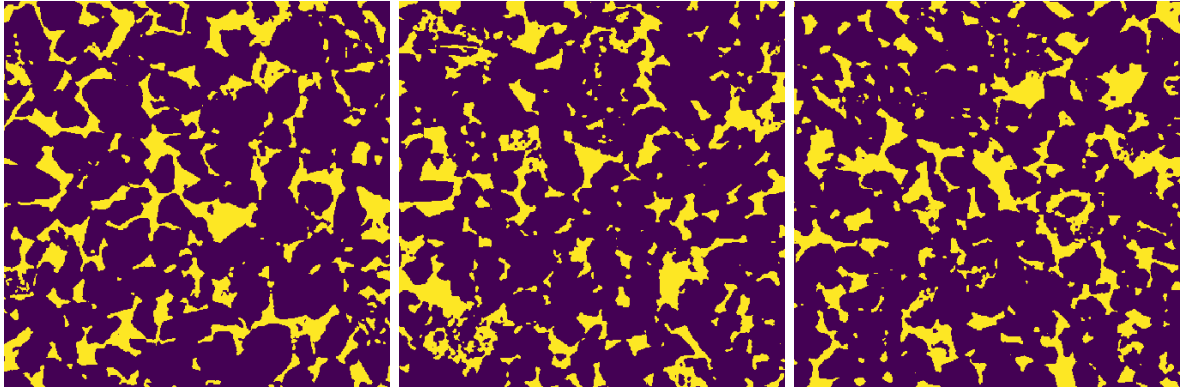
For the **berea** model (fig. 3.11), the synthetic sample is visually very close to the ground truth. We can notice that the synthetic sample is slightly more porous, but this is not really visible on the slices. The network is able to learn the angular shape of grains reasonably well.



(a) Training image (512×512) for *blobs*.

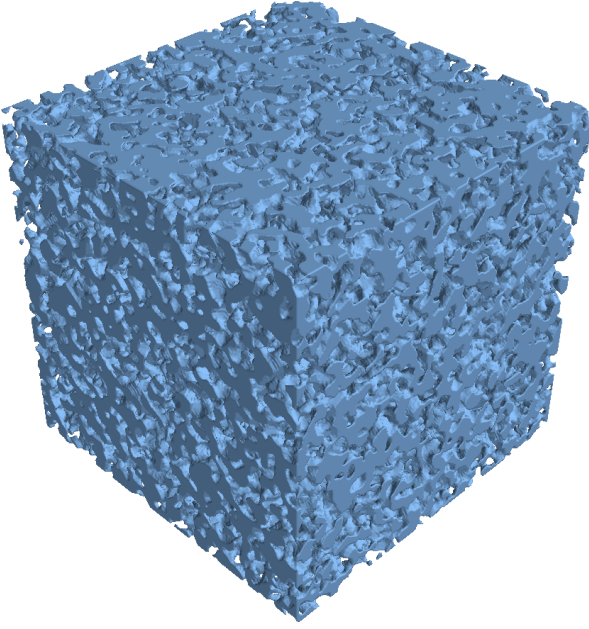


(b) Training image (500×500) for *beadpack*.

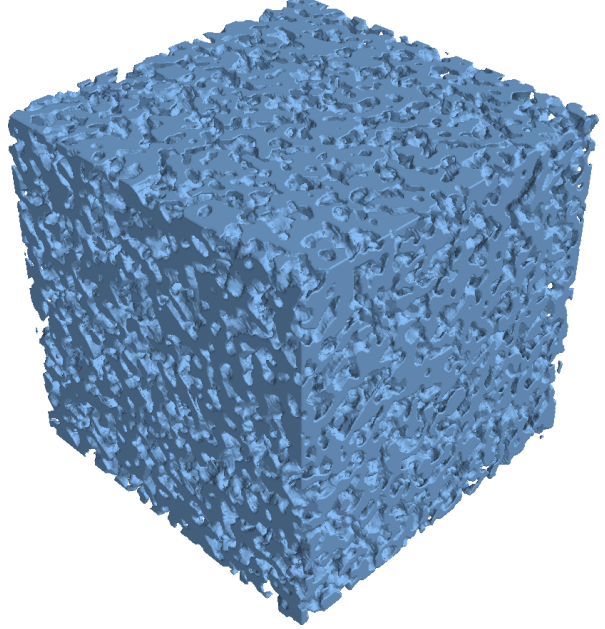


(c) Training images (400×400) in the three principal directions for *berea*.

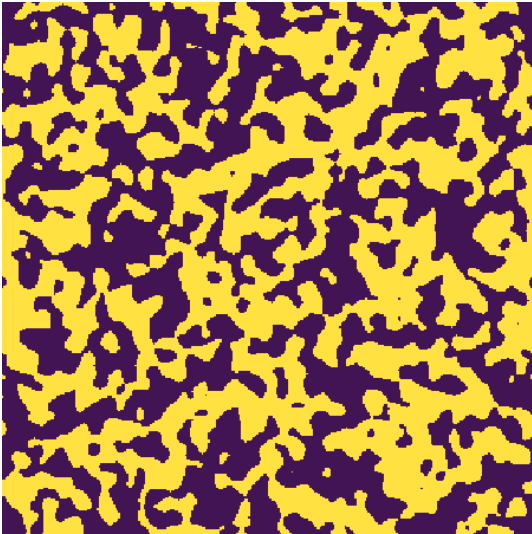
Figure 3.8: Training images for the models *blobs*, *beadpack* and *berea*.



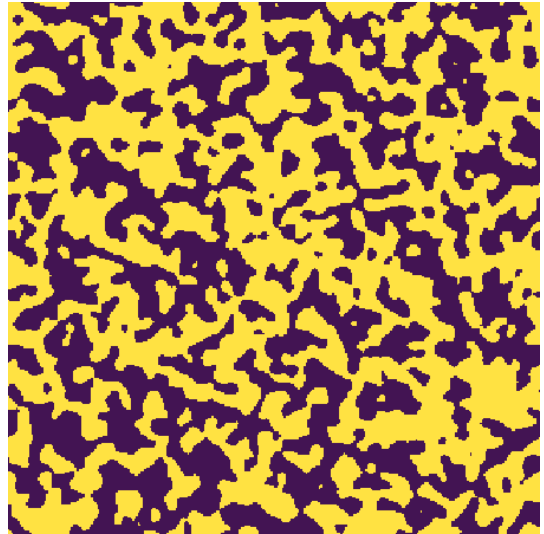
(a) Ground truth sample ($320 \times 320 \times 320$).



(b) Generated sample ($320 \times 320 \times 320$).

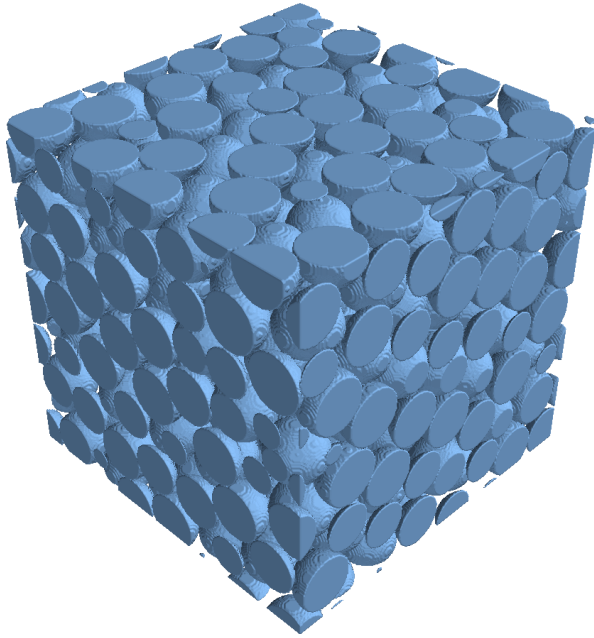


(c) Slice of ground truth sample (320×320).

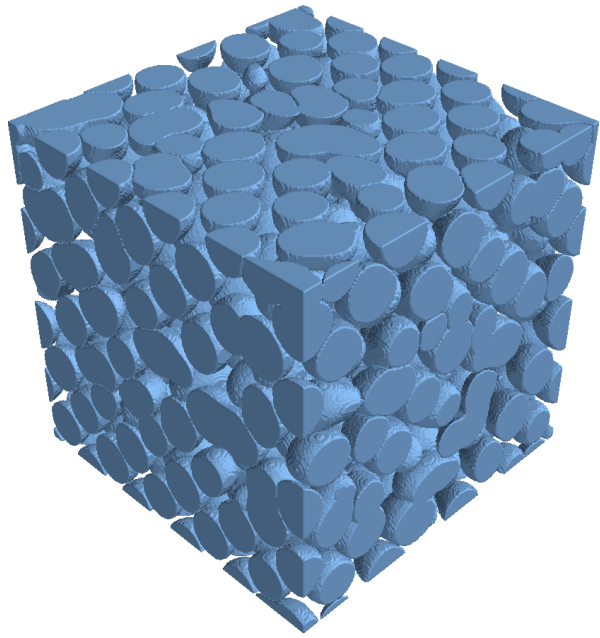


(d) Slice of generated sample (320×320).

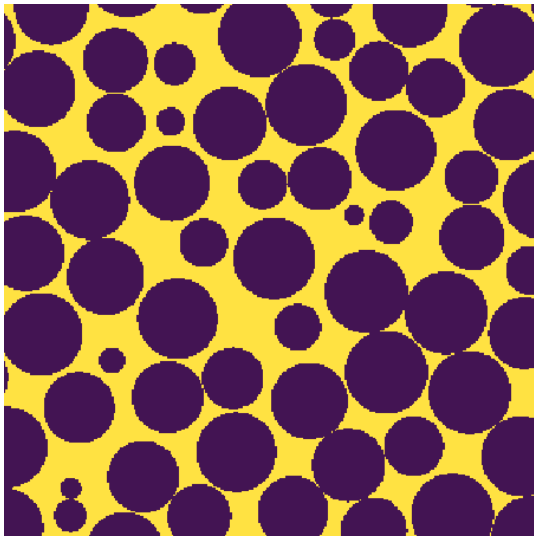
Figure 3.9: Reconstruction results for the `blobs` model.



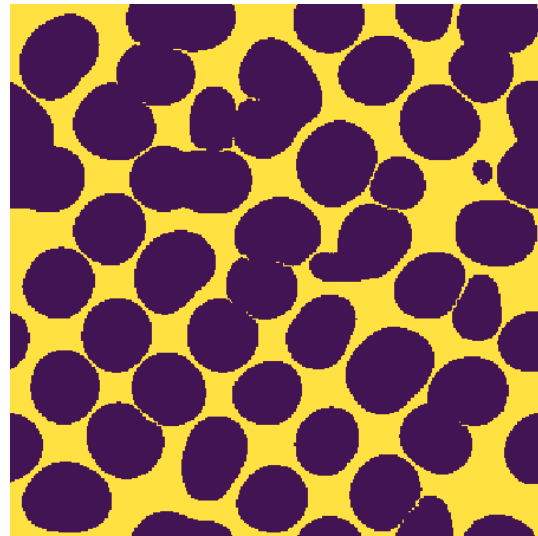
(a) Ground truth sample ($320 \times 320 \times 320$).



(b) Generated sample ($320 \times 320 \times 320$).

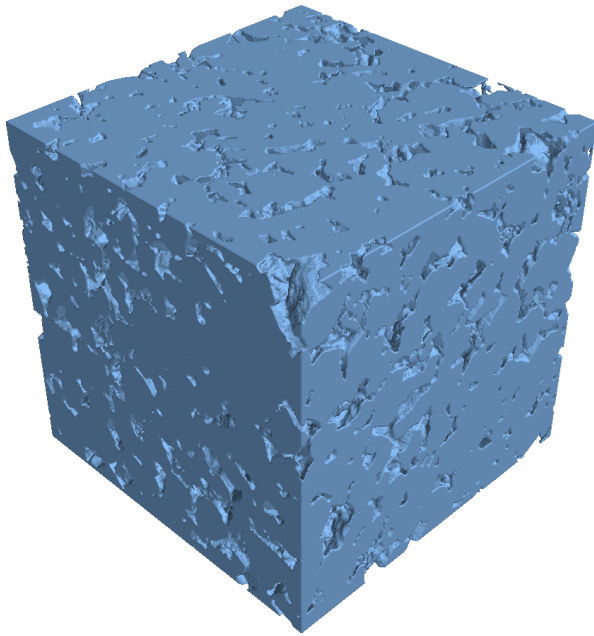


(c) Slice of ground truth sample (320×320).

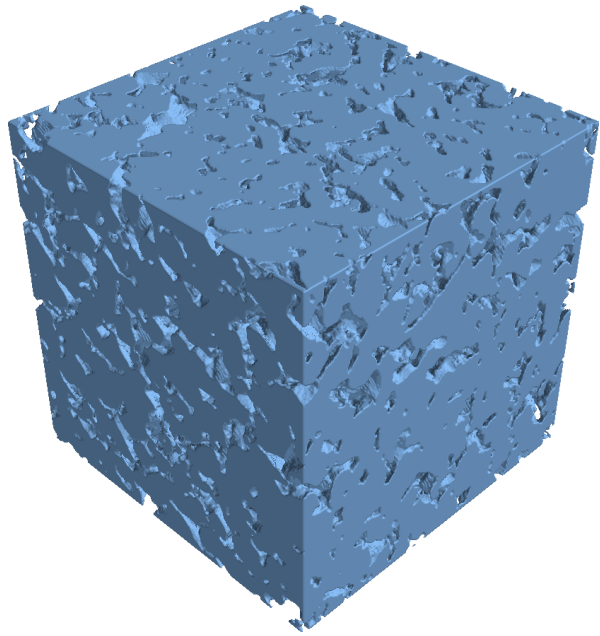


(d) Slice of generated sample (320×320).

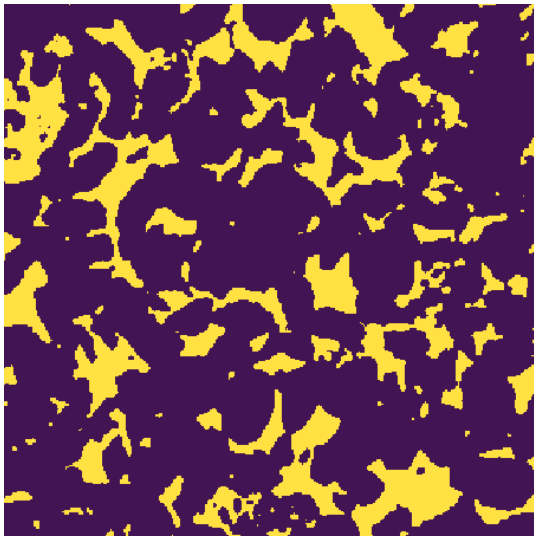
Figure 3.10: Reconstruction results for the **beadpack** model.



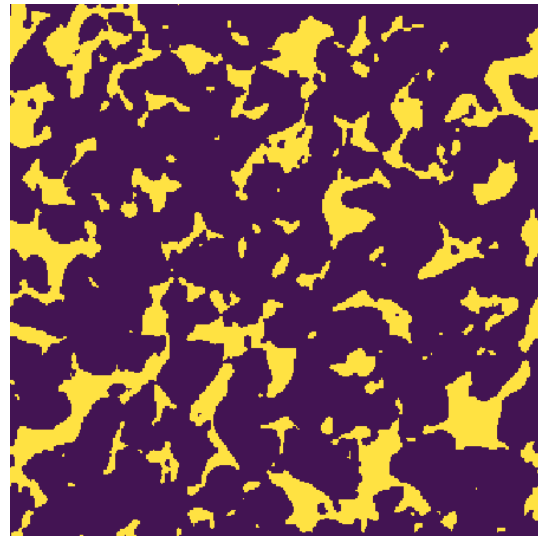
(a) Ground truth sample ($320 \times 320 \times 320$).



(b) Generated sample ($320 \times 320 \times 320$).



(c) Slice of ground truth sample (320×320).



(d) Slice of generated sample (320×320).

Figure 3.11: Reconstruction results for the **berea** model.

3.3.4 Conclusion

In this chapter we have presented in detail a deep learning solution to the porous media reconstruction task. The task is formulated as a texture synthesis problem, and we have leveraged the recent major advances achieved by texture networks in this domain. The solution learns a distribution of the original porous medium, which allows it to generate a variety of samples of the porous material, of arbitrary dimensions, very fast. The convolutional nature of the generator allows it to naturally create samples with periodic boundaries. These aspects make the solution particularly attractive in a Digital Rock Physics context.

We have trained a generator on three porous models, and we have compared, by visual inspection, the synthetic realization of the generator with the 3D ground truth from which the training images originate. The agreement is good, even for challenging models such as sphere packings, which shows that the descriptor is very powerful in extracting the relevant visual features.

At this point, the reader might feel that a visual comparison is not sufficient to assess the quality of the solution, in terms of preservation of the relevant macroscopic properties. In the next chapter, a rigorous validation will be done.

Chapter 4

Validation of the Reconstruction

4.1 Validation methodology

We wish to validate our reconstruction method by defining a set of *metrics* that will quantitatively compare the ground truth with the reconstructed samples. The validation methodology is outlined in fig. 4.1. We use the data set that was presented in section 3.3.2, i.e. the models **blobs**, **beadpack** and **berea**. For each model, we have a three-dimensional ground truth sample of size 320^3 , and two-dimensional training images of sizes 512^2 (**blobs**), 500^2 (**beadpack**) and 400^2 (**berea**). The fact that the training images are larger than the ground truth cube is explained by the fact that the permeability is computationally very costly to compute on large 3D models; therefore we had to limit their size to 320^3 . However we believe that this size is enough to have a representative volume, and the comparison is still fair since in practice high-resolution 2D images are very large.

4.2 Validation metrics and their computation

We now present the set of metrics that we use to assess the quality of the reconstruction, and we provide the numerical schemes for computing them. In order to avoid any confusion, we define the different representations of a porous material formally. A sample is defined on a finite, box-shaped volume $\mathcal{V} = [0, L_x] \times [0, L_y] \times [0, L_z]$. Given a realization ω of some probability space Ω , the volume is partitioned in two regions $\mathcal{V} = (\mathcal{V}_1(\omega), \mathcal{V}_2(\omega))$ corresponding to the pore and solid phases, respectively; the regions are separated by the interface $\partial\mathcal{V}(\omega)$. In the sequel, we omit the mention of the realization ω : $\mathcal{V}_1$, $\mathcal{V}_2$ and $\partial\mathcal{V}$ refer to deterministic sets. The characteristic function I is a binary random field (every $I(\mathbf{x})$ is a binary random variable), and a realization of this random field is denoted $c(\mathbf{x}) = I(\mathbf{x}, \omega)$. Samples are discretized with steps $h_x = L_x/N_x$, $h_y = L_y/N_y$ and $h_z = L_z/N_z$, and we define

$$x_i = \frac{h_x}{2} + ih_x, \quad i = 1, \dots, N_x, \quad (4.1)$$

$$y_j = \frac{h_y}{2} + jh_y, \quad j = 1, \dots, N_y, \quad (4.2)$$

$$z_k = \frac{h_z}{2} + kh_z, \quad k = 1, \dots, N_z, \quad (4.3)$$

such that $c_{ijk} = c(x_i, y_j, z_k)$. When we need to access the phase at unsampled locations, nearest-neighbor interpolation is used, that is,

$$c(x, y, z) = c_{ijk}, \quad \text{for all } x, y, z \text{ such that } |x - x_i| < \frac{h_x}{2}, |y - y_j| < \frac{h_y}{2}, |z - z_k| < \frac{h_z}{2}. \quad (4.4)$$

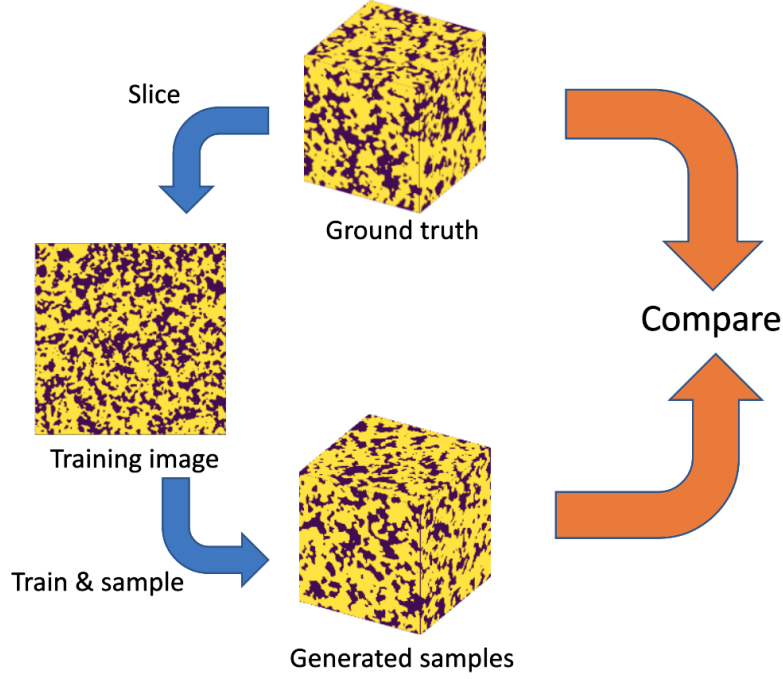


Figure 4.1: Overview of the validation methodology. Given a ground truth volume, a slice (or multiple slices) of the volume is used to train the generator network. When the generator is trained, a set of 10 synthetic samples are generated. These samples are compared against the ground truth volume, according to some metrics.

4.2.1 Porosity

Porosity (or volume fraction) is the fraction of volume that is in the pore phase. Porosity has a great influence on the macroscopic properties of a porous material; hence, preserving the porosity during the reconstruction is an essential requirement. It was defined previously as

$$\phi = \langle I(\mathbf{x}) \rangle. \quad (4.5)$$

The porosity is computed from the discretized sample as the mean phase:

$$\phi = \frac{1}{N_x N_y N_z} \sum_{i,j,k} c_{ijk}. \quad (4.6)$$

4.2.2 Two-point correlation

The two-point correlation function of a *stationary* porous medium was defined in chapter 2 as

$$S_2(\mathbf{r}) = \langle I(\mathbf{x})I(\mathbf{x} + \mathbf{r}) \rangle. \quad (4.7)$$

Computing this metric using the definition (that is, by counting pairs of voxels) is impractical for large volumes. It can be computed much faster by using the *Wiener-Khinchin theorem*, that states that the autocorrelation function (which is here the same as the two-point correlation) and the *power spectral density* (PSD) are Fourier transform pairs. The PSD is defined as

$$P(\omega) = |\hat{I}(\omega)|^2, \quad (4.8)$$

where $\hat{I}(\omega)$ is the Fourier transform of the characteristic function. The theorem states that $P(\omega)$ and $S_2(\mathbf{r})$ are Fourier transform pairs. The PSD can be computed efficiently from the

definition with the fast Fourier transform of c , and the inverse Fourier transform of the PSD gives an estimate of the two-point correlation function.

Isotropic case If the porous medium is assumed to be isotropic, the two-point correlation reduces to a scalar function $S_2(r)$. Obtaining $S_2(r)$ can be done by averaging $S_2(\mathbf{r})$ over all directions in $\mathbb{R}^3$; more precisely, if $\hat{\mathbf{e}}$ is a unit vector randomly distributed over the unit sphere,

$$S_2(r) = \mathbb{E}_{\hat{\mathbf{e}}} [S_2(r\hat{\mathbf{e}})]. \quad (4.9)$$

To perform the averaging over a discretized grid, we directly apply this equation by randomly drawing a large number M of directions $\hat{\mathbf{e}}_1, \dots, \hat{\mathbf{e}}_M$:

$$S_2(r) = \frac{1}{M} \sum_{m=1}^M S_2(r\hat{\mathbf{e}}_m), \quad (4.10)$$

where $S_2(r\hat{\mathbf{e}}_m)$ is obtained by linear interpolation on the 3D grid, which helps getting a more precise estimate, especially near $r = 0$ where the available information is sparse.

4.2.3 Isotropy verification

When we only dispose of a single training image, we provide the image as an example for each of the three principal directions, claiming that the reconstructed volume will be isotropic. We have no guarantee for this, and it is therefore interesting to check if the generated volumes are indeed isotropic. To assess this, we can compute the deviation of the two-point correlation $S_2(r\hat{\mathbf{e}})$ with respect to direction $\hat{\mathbf{e}}$ when r is fixed, i.e.,

$$\sigma_{S_2}(r) = \sqrt{\text{Var}_{\hat{\mathbf{e}}} [S_2(r\hat{\mathbf{e}})]}. \quad (4.11)$$

On a discretized sample, the approach is the same as for computing $S_2(r)$: we compute, for every r , the standard deviation of the set $\{S_2(r\hat{\mathbf{e}}_1), \dots, S_2(r\hat{\mathbf{e}}_M)\}$.

4.2.4 Effective electrical conductivity

Formally, the electrical conductivity of a homogeneous material is represented by a second-order tensor $\boldsymbol{\sigma}$ that relates the (vector) *current density* $\mathbf{J}$ [A/m²] to the (vector) *electric field* $\mathbf{E}$ [V/m] through the vector form of Ohm's law:

$$\mathbf{J} = \boldsymbol{\sigma} \cdot \mathbf{E}. \quad (4.12)$$

If the material is isotropic (which we will assume throughout), the tensor conductivity reduces to a scalar σ_{eff} :

$$\mathbf{J} = \sigma_{\text{eff}} \mathbf{E}. \quad (4.13)$$

Let us recall that this equation is a macroscopic one, i.e. it is only valid at a scale much larger than the pore scale. At the pore scale we assume that the pore phase has conductivity σ_1 , and the solid phase is insulating. For a porous medium $\mathcal{V} = \mathcal{V}_1 \cup \mathcal{V}_2$, we can write

$$\sigma(\mathbf{x}) = \begin{cases} \sigma_1, & \text{if } \mathbf{x} \in \mathcal{V}_1 \\ 0, & \text{if } \mathbf{x} \in \mathcal{V}_2. \end{cases} \quad (4.14)$$

At the pore scale the electrical flow is governed by the equations of electrostatics. We will first lay out these equations, then provide a numerical scheme to compute the electrical flow in the porous medium and obtain an estimate of the effective conductivity in eq. (4.13).

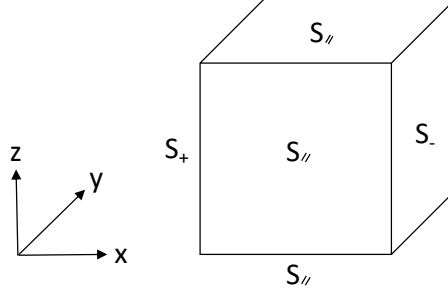


Figure 4.2: Definition of the different faces of the sample. The flow is induced in the $+x$ direction.

Equations of electrostatics In the absence of charge density, *Gauss's law* states that the electric field is divergence-free:

$$\nabla \cdot \mathbf{E} = 0. \quad (4.15)$$

In the absence of a changing magnetic field $\mathbf{B}$, Faraday's law of induction states the the electric field is irrotational:

$$\nabla \times \mathbf{E} = 0, \quad (4.16)$$

and consequently it is the gradient of some scalar field, which is defined as the *electric potential field* φ :

$$\mathbf{E} = -\nabla\varphi. \quad (4.17)$$

Note that the potential field φ is a *harmonic* function, i.e. its laplacian is zero: $\nabla \cdot \mathbf{E} = \nabla \cdot (-\nabla\varphi) = -\nabla^2\varphi = 0$. Since the solid phase is insulating, the flux of electric field (or, equivalently, the potential gradient) across the pore-solid interface must be zero: $\mathbf{E} \cdot \hat{\mathbf{n}} = (\nabla\varphi) \cdot \hat{\mathbf{n}} = 0$, where $\hat{\mathbf{n}}$ is the unit vector normal to the interface. We can summarize the equations governing the electrical flow in the pore phase $\mathcal{V}_1$ with the boundary-value problem

$$\nabla^2\varphi = 0 \quad \text{in } \mathcal{V}_1, \quad (4.18)$$

$$\frac{\partial\varphi}{\partial n} = 0 \quad \text{on } \partial\mathcal{V}. \quad (4.19)$$

Boundary conditions for a finite sample In order to estimate the effective conductivity of a porous sample, we induce a current in the x -direction by imposing a potential difference $\Delta\varphi$. Let S_+ and S_- denote the faces of $\mathcal{V}$ corresponding to $x = 0$ and $x = L_x$, respectively, and let $S_{||}$ denote the remaining faces parallel to the direction of the flow (see fig. 4.2). We need to define boundary conditions on these faces; these will depend on whether the sample is periodic or not:

Periodic sample If the sample is periodic, in the sense that it can be tiled seamlessly along any dimension, then we can take advantage of this property and impose periodic boundary conditions which are beneficial for the accuracy of the estimation. For the faces parallel to the flow direction, $S_{||}$, the periodic boundary conditions can be written as

$$\varphi(x, 0, z) = \varphi(x, L_y, z) \quad \text{for all } x, z, \quad (4.20)$$

$$\varphi(x, y, 0) = \varphi(x, y, L_z) \quad \text{for all } x, y. \quad (4.21)$$

For the faces perpendicular to the flow, S_+ and S_- , we need to prescribe the potential difference $\Delta\varphi$ that will drive the flow, i.e.,

$$\varphi(0, y, z) - \varphi(L_x, y, z) = \Delta\varphi \quad \text{for all } y, z. \quad (4.22)$$

Note that the potential φ is thus not necessarily uniform over S_+ and S_- .

Non-periodic sample If we cannot benefit from periodicity, we need to define the physical behavior of the electrical flow at the boundary. For the faces parallel to the flow direction, $S_{\parallel}$, we choose to insulate the boundaries, which can be written as the Neumann boundary condition

$$(\nabla\varphi) \cdot \hat{\mathbf{n}} = 0 \quad \text{on } S_{\parallel}, \quad (4.23)$$

where $\hat{\mathbf{n}}$ is the unit vector normal to the face. For the faces perpendicular to the flow, S_+ and S_- , we simply impose the value of the potential on each face, which gives the Dirichlet boundary conditions

$$\varphi = \Delta\varphi \quad \text{on } S_+, \quad (4.24)$$

$$\varphi = 0 \quad \text{on } S_-. \quad (4.25)$$

Computation of σ_{eff} Now that all boundary conditions have been defined, we can solve for φ over the whole pore phase $\mathcal{V}_1$. The solution is unique for a non-periodic sample, and unique up to an additive constant for a periodic sample (indeed, one can easily verify that if φ^* verifies the equations, then $\varphi^* + C$ does too, for any constant C ; this is not an issue since we only care about its gradient $\nabla\varphi$). Let $\tilde{S}$ be any slice parallel to S_+ and S_- . The effective current density J induced by the voltage $\Delta\varphi$ is given by

$$J = \langle \mathbf{J} \cdot \hat{\mathbf{n}} \rangle_{\tilde{S}} = \langle \sigma_1 \mathbf{E} \cdot \hat{\mathbf{n}} \rangle_{\tilde{S}} = \langle \sigma_1 (-\nabla\varphi) \cdot \hat{\mathbf{n}} \rangle_{\tilde{S}}, \quad (4.26)$$

where $\langle \cdot \rangle_{\tilde{S}}$ denotes the average over $\tilde{S}$. The value of J does not depend on the chosen slice $\tilde{S}^1$. The effective conductivity is then given by

$$\sigma_{\text{eff}} = \frac{J}{\Delta\varphi/L_x} = \frac{\sigma_1 \langle -\nabla\varphi \cdot \hat{\mathbf{n}} \rangle_{\tilde{S}}}{\Delta\varphi/L_x}. \quad (4.27)$$

The *formation factor*, which may be seen as the dimensionless version of effective conductivity, is given by

$$\frac{1}{F} = \frac{\sigma_{\text{eff}}}{\sigma_1} = \frac{\langle -\nabla\varphi \cdot \hat{\mathbf{n}} \rangle_{\tilde{S}}}{\Delta\varphi/L_x}. \quad (4.28)$$

Non-dimensionalization of the equations To determine how the physical quantities scale with the size of the problem, it is instructive to get rid of the physical dimensions. The governing equations for the electrical flow can be made dimensionless using the substitutions

$$\nabla = \frac{1}{L_x} \tilde{\nabla}, \quad \varphi = (\Delta\varphi) \tilde{\varphi}, \quad (4.29)$$

which yields, for the formation factor eq. (4.28),

$$\frac{1}{F} = \langle -\tilde{\nabla} \tilde{\varphi} \cdot \hat{\mathbf{n}} \rangle_{\tilde{S}} \quad (4.30)$$

This shows that the electrical flow solution and the formation factor do not depend on any of the parameters chosen, and they are therefore independent of the problem's scale.

Numerical scheme The boundary-value problem can be solved by the means of a *finite-difference scheme*. The scheme is presented for the two-dimensional case to lighten the notations, but the extension to three dimensions is direct.

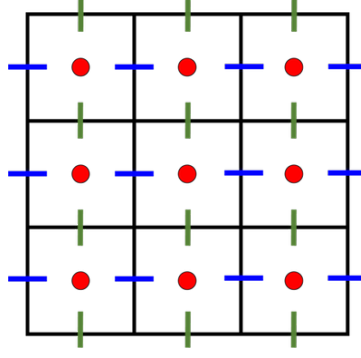


Figure 4.3: Two-dimensional staggered grid. In the present problem, the red dots represent the potential $\varphi_{i,j}$, the blue dashes the horizontal potential gradient $\left(\frac{\partial\varphi}{\partial x}\right)_{i+\frac{1}{2},j}$ and the green dashes the vertical potential gradient $\left(\frac{\partial\varphi}{\partial y}\right)_{i,j+\frac{1}{2}}$.

Let φ_{ij} denote the approximation of the electric potential field at (x_i, y_j) . Using a *staggered grid* (see fig. 4.3), the components of the potential gradient $\nabla\varphi$ are computed halfway between the mesh points:

$$\left(\frac{\partial\varphi}{\partial x}\right)_{i+\frac{1}{2},j} = c_{ij}c_{i+1,j}\left(\frac{\varphi_{i+1,j} - \varphi_{ij}}{h_x}\right) \quad (4.31)$$

$$\left(\frac{\partial\varphi}{\partial y}\right)_{i,j+\frac{1}{2}} = c_{ij}c_{i,j+1}\left(\frac{\varphi_{i,j+1} - \varphi_{ij}}{h_y}\right) \quad (4.32)$$

In order to verify the boundary condition eq. (4.19), the gradient is set to zero when evaluated at the pore-solid interface. We can then discretize the Laplacian and use eq. (4.18):

$$0 = \left(\nabla^2\varphi\right)_{ij} = (\nabla \cdot \nabla\varphi)_{ij} = \frac{\left(\frac{\partial\varphi}{\partial x}\right)_{i+\frac{1}{2},j} - \left(\frac{\partial\varphi}{\partial x}\right)_{i-\frac{1}{2},j}}{h_x} + \frac{\left(\frac{\partial\varphi}{\partial y}\right)_{i,j+\frac{1}{2}} - \left(\frac{\partial\varphi}{\partial y}\right)_{i,j-\frac{1}{2}}}{h_y}. \quad (4.33)$$

The finite-difference scheme results in a system of N_p linear equations with N_p unknowns, where N_p is the number of points in the pore phase. The system can be written compactly as

$$\mathbf{L}\varphi = \mathbf{b}, \quad (4.34)$$

where it can be shown that $\mathbf{L}$ is a positive definite $N_p \times N_p$ matrix. Since it is very large, we need to use a sparse iterative solver. We have used the *biconjugate gradient stabilized method* (BiCGSTAB) [Vor92], run on a single GPU with the `pyamgx` library.

When the system is solved, the formation factor can be computed using eq. (4.30): for any slice i ,

$$\frac{1}{F} = \frac{\sigma_{\text{eff}}}{\sigma_1} = -\sum_{j=1}^{N_y} h_y \left(\frac{\partial\varphi}{\partial x}\right)_{i+\frac{1}{2},j}. \quad (4.35)$$

¹ This can be proved by applying the divergence theorem on the volume enclosed by S_+ and $\tilde{S}$, and using the fact that the divergence of $\mathbf{E}$ is zero.

4.2.5 Hydraulic permeability

Single-phase (or absolute) hydraulic permeability (hereafter simply called permeability) quantifies the ability of a porous material to let fluids flow through it. Just like electrical conductivity describes the current induced by an applied voltage, permeability describes the fluid flow induced by an applied pressure gradient.

We assume that the fluid in question is an incompressible Newtonian fluid, such that its velocity field $\mathbf{v}(\mathbf{x}) \in \mathbb{R}^3$ is divergence-free:

$$\nabla \cdot \mathbf{v} = 0. \quad (4.36)$$

The flow is then governed by the following Navier-Stokes equations, given here for a fixed viscosity μ [Bat00]:

$$\mu \nabla^2 \mathbf{v} = \rho \left(\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} \right) + \nabla p, \quad (4.37)$$

where p is the pressure field and ρ the (constant) fluid density. We further make two approximations that are usual with flows in porous media:

Creeping flow The *Reynolds number* Re of the flow is defined as

$$\text{Re} = \frac{\rho v l}{\mu}, \quad (4.38)$$

where v is a characteristic flow speed and l a characteristic length scale, in this context a representative pore length. The Reynolds number is a representative ratio of the nonlinear inertia $\rho \mathbf{v} \cdot \nabla \mathbf{v}$ to the viscous dissipation $\mu \nabla^2 \mathbf{v}$: indeed, inertia is of the order of $\rho v^2/l$ and viscous dissipation is of the order of $\mu v/l^2$. A distinctive feature of porous media flows is that v and l are very small²: roughly, $v = 10^{-7}..10^{-5}$ m/s and $l = 10^{-6}..10^{-3}$ m. For water, $\rho \approx 10^3$ kg/m³ and $\mu \approx 10^{-3}$ Pa · s. In all cases we see that $\text{Re} \ll 1$. This means that inertial effects are negligible compared to the viscous dissipation. The inertial term can therefore be ignored in eq. (4.37) and

$$\mu \nabla^2 \mathbf{v} = \rho \frac{\partial \mathbf{v}}{\partial t} + \nabla p. \quad (4.39)$$

Steady-state flow We assume that p and $\mathbf{v}$ are time-independent, i.e. they are functions of space only. Time-dependent phenomena can happen in porous media, but usually they take place at time scales that are much slower than the time scale of the microscopic fluid displacement. We can therefore safely say that the flow remains largely static, and the time-dependent term in eq. (4.39) can be neglected.

We then arrive at the well-known *Stokes flow* equations:

$$\mu \nabla^2 \mathbf{v} = \nabla p \quad \text{in } \mathcal{V}_1, \quad (4.40)$$

$$\nabla \cdot \mathbf{v} = 0 \quad \text{in } \mathcal{V}_1. \quad (4.41)$$

At the pore-solid interface, we impose a no-slip boundary condition for the velocity and a Neumann boundary condition for the pressure, which are valid for Newtonian fluids:

$$\mathbf{v} = \mathbf{0} \quad \text{on } \partial \mathcal{V}, \quad (4.42)$$

$$\frac{\partial p}{\partial n} = 0 \quad \text{on } \partial \mathcal{V}. \quad (4.43)$$

² Note that we have not formally defined v nor l , and a standard definition does not exist. For a more complete discussion we refer the reader to [Blu17].

where $\frac{\partial p}{\partial n}$ is the directional derivative of p in the direction orthogonal to the interface. Given the equations for the flow, we can derive *Darcy's law* that defines the permeability. We provide here an informal derivation as in [Blu17]; a formal proof can be found in [Sah11]. Stokes flow leads to a linear relationship between the pressure gradient and the flow field. To see this, let $\mathbf{v} = \mathbf{v}_0$ be a solution to the problem. Now consider that we increase the pressure gradient ∇p by some arbitrary factor a . Then we can see that $\mathbf{v} = a\mathbf{v}_0$ is the solution for this new pressure gradient, i.e. $\mathbf{v}$ is proportional to ∇p . The same argument can be applied to show that the velocity is inversely proportional to the viscosity. Hence we can write

$$\mathbf{v} = -\frac{f(\mathbf{x})}{\mu} \nabla p, \quad (4.44)$$

where f is some positive function of space that depends only on the porous geometry and not on the fluid property μ nor the driving pressure gradient ∇p . A proper averaging of eq. (4.44) leads to Darcy's law:

$$\mathbf{q} = -\frac{\kappa}{\mu} \nabla p, \quad (4.45)$$

where $\mathbf{q}$ is the *Darcy velocity*, defined as the volume of fluid flowing per unit area (this area includes both solid and pore). ∇p is no longer a microscopic quantity, but is defined over the macroscopic scale. κ is the permeability and is an intrinsic property of the pore structure of the system; it has the units of m^2 . Note that in general, like electrical conductivity, permeability is a second-order tensor; however for anisotropic materials it reduces to a positive scalar.

Boundary conditions for a finite sample We induce a flow in the x -direction by imposing a pressure difference Δp . Let S_+ , S_- and $S_{\parallel}$ be defined as before. The boundary conditions applied on the faces of the domain depend on whether the sample is periodic or not:

Periodic sample If the sample is periodic, we impose that the velocity field is fully periodic; the same applies to the pressure gradient, except for the faces S_+ and S_- for which we apply the pressure difference:

$$\mathbf{v}(x, 0, z) = \mathbf{v}(x, L_y, z), \quad p(x, 0, z) = p(x, L_y, z), \quad \text{for all } x, z, \quad (4.46)$$

$$\mathbf{v}(x, y, 0) = \mathbf{v}(x, y, L_z), \quad p(x, y, 0) = p(x, y, L_z), \quad \text{for all } x, y, \quad (4.47)$$

$$\mathbf{v}(0, y, z) = \mathbf{v}(L_x, y, z), \quad p(0, y, z) - p(L_x, y, z) = \Delta p, \quad \text{for all } y, z. \quad (4.48)$$

Note that the pressure p is not necessarily uniform over S_+ and S_- .

Non-periodic sample For the faces parallel to the flow direction, $S_{\parallel}$, we put a virtual solid boundary, which amounts to apply the same conditions as on $\partial\mathcal{V}$, eqs. (4.42) and (4.43). For the in- and outflow faces S_+ and S_- , we apply natural in- and outflow conditions for the velocity field components $\mathbf{v} = (u, v, w)$, and a Dirichlet condition for the pressure field:

$$\frac{\partial u}{\partial x} = \frac{\partial v}{\partial x} = \frac{\partial w}{\partial x} = 0 \quad \text{on } S_+ \text{ and } S_-, \quad (4.49)$$

$$p = \Delta p \quad \text{on } S_+, \quad (4.50)$$

$$p = 0 \quad \text{on } S_-. \quad (4.51)$$

Computation of the permeability κ Now that all boundary conditions have been defined, we can solve for $\mathbf{v}$ and p over the whole pore phase $\mathcal{V}_1$. The solution for $\mathbf{v}$ is unique, and the solution for p is unique for a non-periodic sample and unique up to an additive constant for a periodic samples. Let $\tilde{S}$ be any slice parallel to S_+ and S_- . The Darcy velocity (flow per unit area) in the x direction induced by the pressure difference Δp is given by

$$q = \langle \mathbf{v} \cdot \hat{\mathbf{n}} \rangle_{\tilde{S}} = \langle u \rangle_{\tilde{S}}. \quad (4.52)$$

Again, the value of q does not depend on the chosen slice $\tilde{S}$. The permeability is then given by eq. (4.45):

$$\kappa = \frac{\mu q}{\Delta p / L_x} = \frac{\mu \langle u \rangle_{\tilde{S}}}{\Delta p / L_x}. \quad (4.53)$$

Non-dimensionalization of the equations Equations (4.40) and (4.41) can be made dimensionless using the substitutions

$$\nabla = \frac{1}{L_x} \tilde{\nabla}, \quad \mathbf{v} = \frac{L_x \Delta p}{\mu} \tilde{\mathbf{v}}, \quad p = (\Delta p) \tilde{p}, \quad (4.54)$$

which yield

$$\tilde{\nabla}^2 \tilde{\mathbf{v}} = \tilde{\nabla} \tilde{p} \quad \text{in } \mathcal{V}_1, \quad (4.55)$$

$$\tilde{\nabla} \cdot \tilde{\mathbf{v}} = 0 \quad \text{in } \mathcal{V}_1, \quad (4.56)$$

$$(4.57)$$

and

$$\frac{\kappa}{L_x^2} = \langle \tilde{\mathbf{v}} \cdot \hat{\mathbf{n}} \rangle_{\tilde{S}}. \quad (4.58)$$

From the dimensionless Stokes equations we see that the solution does not depend on any physical parameter; in particular, we do not need to specify the viscosity μ . When the geometry is fixed, the permeability scales quadratically with the size L_x of the domain.

Numerical scheme The Stokes flow can be solved by the means of a finite-difference scheme based on the work of [MBB13]. Again, we present the scheme in two dimensions, but it can be readily extended to three dimensions.

Using a staggered grid again (fig. 4.3), we define the approximated pressure, horizontal and vertical velocity fields:

$$p_{ij} \approx p(x_i, y_j), \quad u_{i+\frac{1}{2},j} \approx u\left(x_{i+\frac{1}{2}}, y_j\right), \quad v_{i,j+\frac{1}{2}} \approx v\left(x_i, y_{j+\frac{1}{2}}\right). \quad (4.59)$$

Let us ignore boundary conditions for the moment. In the bulk of the fluid, the Stokes flow equations eqs. (4.40) and (4.41) are discretized with centered finite differences:

$$\mu \left(\frac{u_{i+\frac{3}{2},j} - 2u_{i+\frac{1}{2},j} + u_{i-\frac{1}{2},j}}{h_x^2} + \frac{u_{i+\frac{1}{2},j+1} - 2u_{i+\frac{1}{2},j} + u_{i+\frac{1}{2},j-1}}{h_y^2} \right) = \frac{p_{i+1,j} - p_{i,j}}{h_x}, \quad (4.60)$$

$$\mu \left(\frac{v_{i,j+\frac{3}{2}} - 2v_{i,j+\frac{1}{2}} + v_{i,j-\frac{1}{2}}}{h_y^2} + \frac{v_{i+1,j+\frac{1}{2}} - 2v_{i,j+\frac{1}{2}} + v_{i-1,j+\frac{1}{2}}}{h_x^2} \right) = \frac{p_{i,j+1} - p_{i,j}}{h_y}, \quad (4.61)$$

$$\frac{u_{i+\frac{1}{2},j} - u_{i-\frac{1}{2},j}}{h_x} + \frac{v_{i,j+\frac{1}{2}} - v_{i,j-\frac{1}{2}}}{h_y} = 0. \quad (4.62)$$

We thus have a system of approximately $4N_p$ equations, where N_p is the number of pore voxels, with an equal number of unknowns. This system of equations can be written compactly as

$$\begin{pmatrix} A & B \\ B^T & 0 \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix}, \quad (4.63)$$

where u gathers all velocity components and p all pressure values, and f is determined by the boundary conditions. This system is symmetric and matrix A is positive definite³; however it can be proved [Che] that the whole system is *indefinite* (it is also called a *saddle point system*), which makes it much harder to solve comparing with positive definite systems. There exists several methods to deal with this kind of problems; we choose to employ the *Uzawa iterative scheme* [Che], which iteratively updates the velocity fields and the pressure field separately:

1. Update velocity by solving the momentum equations:

$$u^{k+1} = A^{-1}(f - B^T p^k), \quad (4.64)$$

2. Update pressure:

$$p^{k+1} = p^k + \omega B u^{k+1}, \quad (4.65)$$

where ω is a chosen parameters. The scheme can be shown to converge towards the solution with a linear rate when $\omega = 1$.

Convergence criterion For the choice of the convergence criterion, we follow [MBB13]. First we define the mass continuity imbalance for a voxel p (i.e. the net flux balance, which should be zero for an incompressible fluid):

$$b_p = h_x h_y h_z (\nabla \cdot \mathbf{v})_p. \quad (4.66)$$

Then we define the average flux passing through a single *pore* voxel:

$$Q_v = \frac{Q}{N_y N_z \phi}, \quad (4.67)$$

where Q is the total flux in the x direction. The convergence criterion CC is then defined as the average absolute continuity imbalance, normalized by the average flux:

$$\text{CC} = \frac{\frac{1}{N_p} \sum_{p=1}^{N_p} |b_p|}{Q_v} = \frac{\phi h_x |\nabla \cdot \mathbf{v}|_{\text{avg}}}{Q}. \quad (4.68)$$

The normalization makes the criterion dimensionless. In our applications we iterate until $|\text{CC}| < 10^{-6}$.

Treatment of boundaries The boundary conditions are treated as with the electrical flow problem, however a particular case requires special attention. Figure 4.4 illustrates the situation. Applying a no-slip boundary condition for the velocity component perpendicular to the boundary (here v) is simply done by setting it to zero. However, we cannot do the same for the component parallel to the boundary (here u) since it does not lie exactly on the boundary. This can be

³ The system is not completely symmetric when we add boundary conditions, due to their special treatment that is detailed later; however, the presented numerical scheme remains valid.

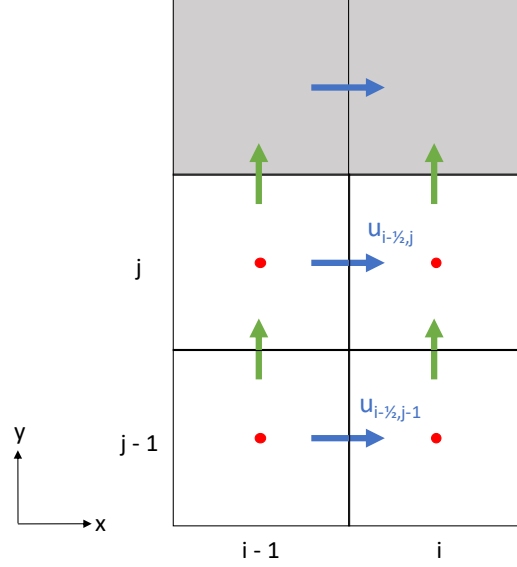


Figure 4.4: Special treatment of solid boundaries for parallel velocity components. The shaded area represents the solid boundary.

addressed by first extrapolating the value of $u_{i-\frac{1}{2},j+1}$ knowing the values of $u_{i-\frac{1}{2},j}$, $u_{i-\frac{1}{2},j-1}$, and $u_{i-\frac{1}{2},j+\frac{1}{2}} = 0$:

$$u_{i-\frac{1}{2},j+1} = \frac{1}{3} \left(u_{i-\frac{1}{2},j-1} - 6u_{i-\frac{1}{2},j} \right). \quad (4.69)$$

Substituting the expression into eq. (4.60) yields the following non-centered finite difference approximation to $\frac{\partial^2 u}{\partial y^2}$:

$$\left(\frac{\partial^2 u}{\partial y^2} \right)_{i-\frac{1}{2},j} = \frac{-12u_{i-\frac{1}{2},j} + 4u_{i-\frac{1}{2},j-1}}{3h_y^2}. \quad (4.70)$$

A similar expression can be written for the transpose case, $\frac{\partial^2 v}{\partial x^2}$.

When the system is solved, the permeability can be computed using eq. (4.53): for any slice i ,

$$\kappa = \frac{\mu}{\Delta p / L_x} \sum_{j=1}^{N_y} h_y u_{i+\frac{1}{2},j}. \quad (4.71)$$

Connection between conductivity and permeability As a comment, we note that there exists close connections between electrical conductivity and permeability. A particular approximate relationship, given by [johnsonNewPoreSizeParameter1986] allows to approximate the permeability given the electric field:

$$\kappa \approx \frac{\Lambda^2}{8F}, \quad \text{where } \frac{\Lambda}{2} = \frac{\int_{\mathcal{V}_1} \|\mathbf{E}\|^2 dV}{\int_{\partial\mathcal{V}} \|\mathbf{E}\|^2 dS}. \quad (4.72)$$

The parameter Λ has the units of length; it is a weighted pore volume-to-surface ratio that measures how the electric field behaves both in the bulk of the fluid and at the walls. This relationship suggests that a good agreement in the electrical properties will probably imply a good agreement in the hydraulic properties.

4.3 Results

The metrics described in the previous section have been computed on the ground truth samples and on a set of 10 reconstructed samples, for the three models **blobs**, **beadpack** and **berea**.

Figures 4.5, 4.8 and 4.11 show the results for the porosity ϕ , the effective electrical conductivity $\sigma_{\text{eff}}/\sigma$ and the hydraulic permeability κ/L^2 , both visually and quantitatively. The *relative error* is defined as the error of the mean of sample values, relative to the ground truth value; it measures how well the reconstruction preserves the macroscopic quantities. The *relative deviation* is defined as the standard deviation of sample values, divided by their mean; it measures the dispersion of the reconstructed samples.

Figures 4.6, 4.9 and 4.12 show the results for the (isotropic) two-point correlation function $S_2(r)$, and figs. 4.7, 4.10 and 4.13 show the isotropy verification.

For illustration, we have plotted, on fig. 4.14, cross-sections of the electric potential field, for both the ground truth sample and a generated sample of each of the three models. On fig. 4.15 we have done the same but for the amplitude of the velocity corresponding to Stokes flow.

4.4 Discussion

Discrepancies between metrics computed on the ground truth and the generated samples can have different origins. We have identified the following *error sources*:

- 1. Accuracy of numerical scheme.** The numerical determination of the different quantities is a first error source. This is especially true for the conductivity and the permeability: accurately computing these properties on the complex geometries involved in porous media is already a challenging task. However we assume that the error committed by our numerical schemes will be negligible (or at most much smaller) than the other errors.
- 2. Representativeness of samples.** Since all quantities are averages over the volumes, accurately estimating them requires the samples to be sufficiently large to be statistically representative. This representativeness is especially critical for the estimation of the physical properties, even more so when the porosity is low (the notion of representativeness for permeability estimation has been studied by [MBB13]). Since the size of the volumes is limited by the numerical scheme, it is probable that this error will be noticeable. The volume size also plays a role in the bias induced by the non-periodic boundary conditions: indeed, for small samples, the insulated side faces can cause a significant decrease in conductivity and permeability.
- 3. Variance of generator.** The generator produces a variety of different samples, and this variety induces variance on the quantities of interest. We believe that by averaging over a set of 10 samples is enough to limit this variance, and the corresponding error should therefore not be significant.
- 4. Reconstruction error.** Such error is committed when the reconstruction fails to preserve a quantity that could provably be deduced from the training image.
- 5. Representativeness of the training slice.** The training slice is a finite realization of an infinite binary random field; therefore, if we want a quantity to be preserved by the slicing, the slice should at least be large enough. We will assume that our training images are representative enough of their underlying random process, and thus that the corresponding error is insignificant.

6. Representativeness of two-dimensional information. Whether the information contained in a two-dimensional slice is enough to uniquely recover the macroscopic quantities of the three-dimensional volume is an open question. It is true for statistical measures such as porosity and two-point correlation, but we have no guarantee for the properties characterizing physical processes happening in three dimensions. Reconstruction can only be done successfully if this fundamental assumption holds true.

We assume that errors 1, 3, and 5 are negligible with respect to the other error types, for the reasons given above. When discussing the validation results, we will try to identify (or guess) which of the remaining error types (2, 4 or 6) is mainly causing the discrepancies.

Porosity We can observe that the porosity is not exactly matched by the reconstruction: there is a systematic bias, however this bias is acceptably low. Overall, the relative deviation in porosity across generated samples is very small (less than 0.5 %), which indicates that the bias is not due to the diversity of samples. Clearly, the porosity bias is a reconstruction error (type 4), since this information is available exactly in the training image, and the generated volumes are large enough to estimate it accurately.

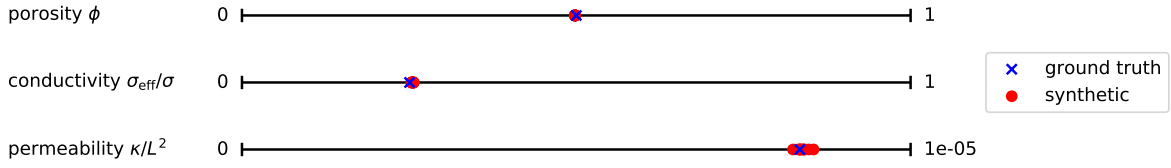
Two-point correlation function For **blobs** and **berea**, the agreement is good and the variance across samples is very low, although there is a slight bias with respect to the ground truth, that corresponds to the porosity bias. For **beadpack**, the synthetic samples lag behind the oscillatory pattern of the ground truth correlation. We have also shown the correlation function for the training slice, to demonstrate that the bias does not come from the training slice incorrectly capturing the correlation function. The lag may indicate that the sphere-like shapes in the reconstructed samples are a bit smaller than the perfect spheres in the ground truth. The variance across generated samples is still reasonably low. We conclude that the discrepancy is mainly due to a reconstruction error (type 4).

Isotropy We can systematically observe a spike, both for ground truth and samples, near $r = 0$: this is probably due to the sparsity of the correlation information near the origin. The isotropy measure for the ground truth is very low in all cases (even for **berea**, which we did not assume). For **blobs** and **berea**, there is clearly anisotropy when we look over short distances (between 0 and 20 pixels); however it is still quite low when compared to the two-point correlation function at these distances. On the other hand, for **beadpack** the variance is much higher than in the other cases, even over long distances. We suspect that it is due to the elongated sphere shapes that tend to align in a single direction.

Electrical conductivity Conductivity shows a higher variance across samples than porosity (up to 2 %); this variance suggests that the volumes are a bit too small to be representative for conductivity (type 2 error). The mean relative error is larger than this deviation (much larger for **berea**), which indicates that it is not the primary cause of the error. Since the relative error is still acceptably small for **blobs** and **beadpack**, we claim that the reconstruction procedure is able to correctly preserve the conductivity, and that the large error for **berea** is a type 6 error, i.e., 2D information is not enough to uniquely determine the electrical properties.

Hydraulic permeability Permeability shows an even higher variance across samples (up to 4.38 %) which demonstrates that volume representativeness is critical for permeability estimation. For **blobs**, the deviation is higher than the mean error, which suggests that the error is mainly of type 2. For **beadpack** and **berea**, the relative error is very high, and it is difficult to decide

whether this error is due to the reconstruction (type 4), or the information contained in the training slice (type 6).



	ground truth	mean of samples	relative error	relative deviation
ϕ	50.00 %	49.80 %	−0.40 %	0.03 %
$\sigma_{\text{eff}}/\sigma$	0.251	0.255	1.82 %	0.47 %
κ/L^2	8.34×10^{-6}	8.38×10^{-6}	0.48 %	0.96 %

Figure 4.5: Porosity, conductivity and permeability validation for **blobs**

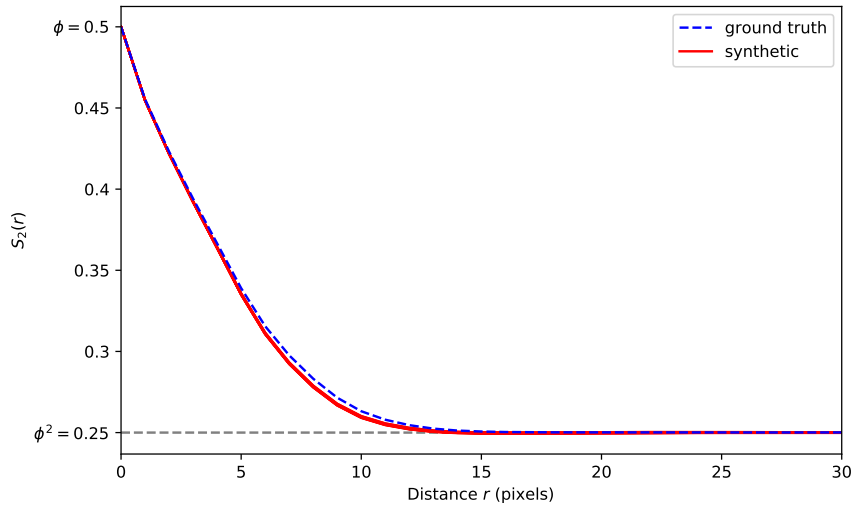


Figure 4.6: Validation of the two-point correlation function for the **blobs** model. The ground truth is compared with a set of 10 synthetic samples.

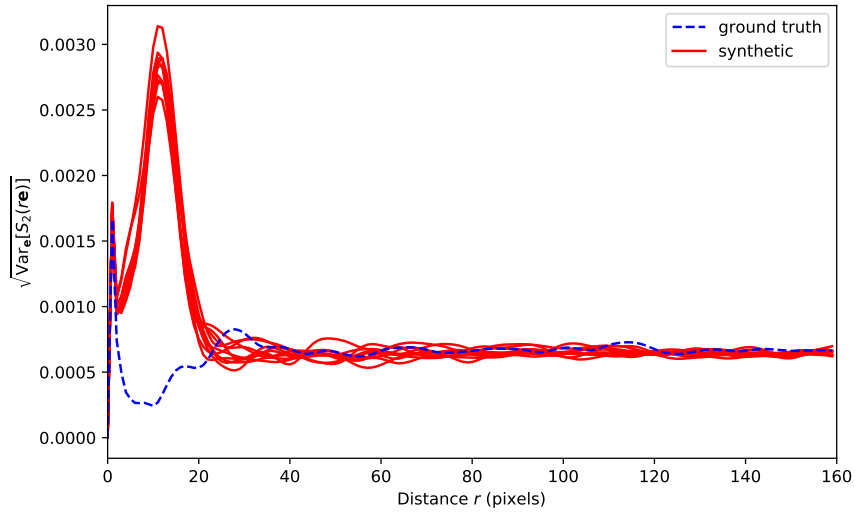
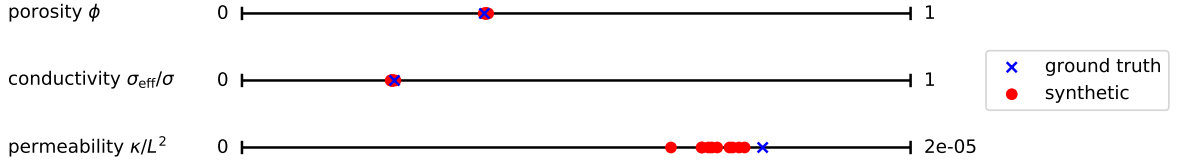


Figure 4.7: Validation of the isotropy for the **blobs** model. For each sample, we plot the deviation of the two-point correlation with respect to direction, for each fixed distance: $\sigma_{S_2}(r) = \sqrt{\text{Var}_{\hat{e}}[S_2(r\hat{e})]}$.



	ground truth	mean of samples	relative error	relative deviation
ϕ	36.26 %	36.48 %	0.62 %	0.51 %
$\sigma_{\text{eff}}/\sigma$	0.228	0.225	−1.32 %	0.91 %
κ/L^2	1.56×10^{-5}	1.42×10^{-5}	−9.01 %	4.38 %

Figure 4.8: Porosity, conductivity and permeability validation for *beadpack*

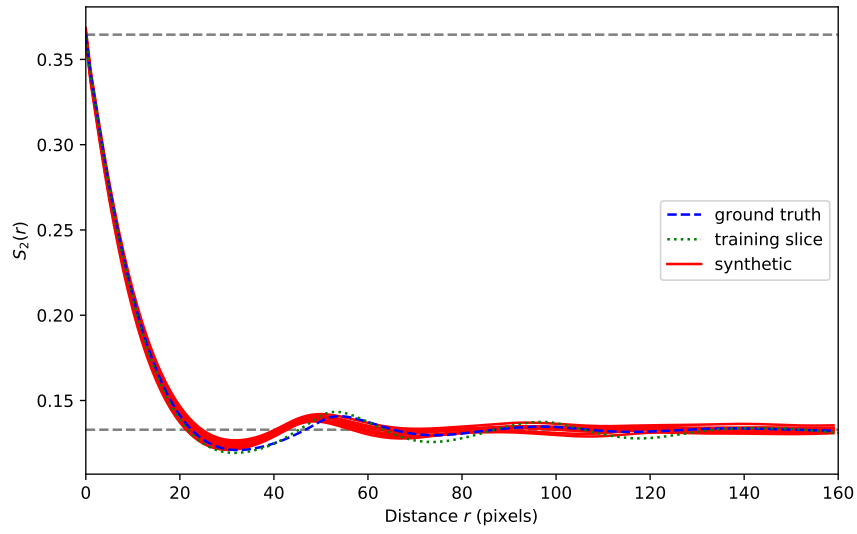


Figure 4.9: Validation of the two-point correlation function for the *beadpack* model. The ground truth is compared with a set of 10 synthetic samples.

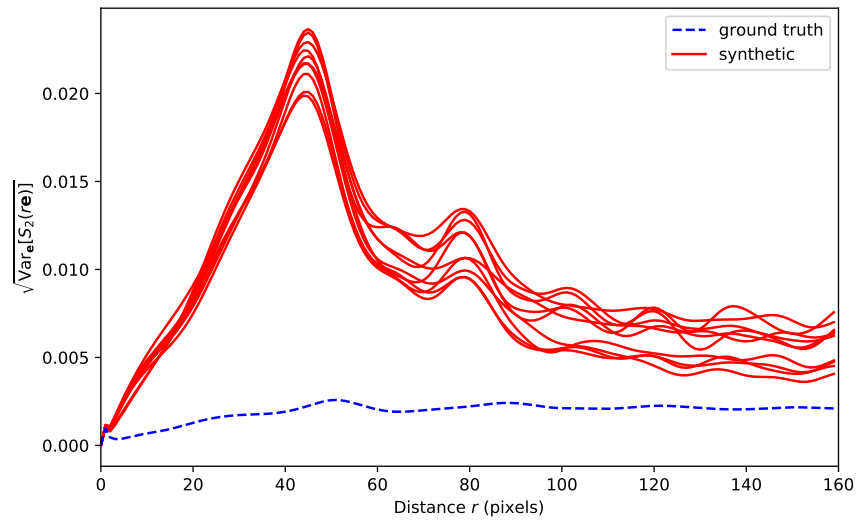
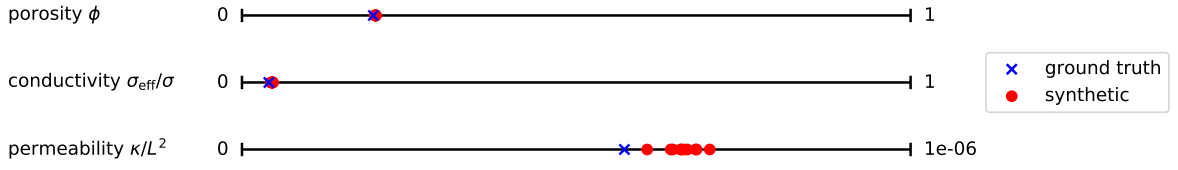


Figure 4.10: Validation of the isotropy for the *beadpack* model. For each sample, we plot the deviation of the two-point correlation with respect to direction, for each fixed distance: $\sigma_{S_2}(r) = \sqrt{\text{Var}_{\mathbf{e}}[S_2(r\mathbf{e})]}$.



	ground truth	mean of samples	relative error	relative deviation
ϕ	19.65 %	19.97 %	1.65 %	0.22 %
$\sigma_{\text{eff}}/\sigma$	4.02×10^{-2}	4.52×10^{-2}	12.35 %	2.03 %
κ/L^2	5.72×10^{-7}	6.59×10^{-7}	15.21 %	3.67 %

Figure 4.11: Porosity, conductivity and permeability validation for *berea*

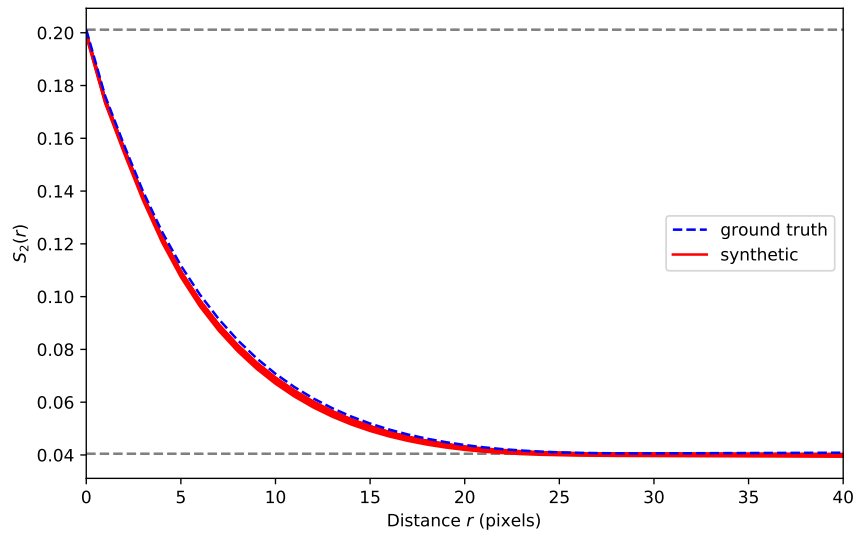


Figure 4.12: Validation of the two-point correlation function for the *berea* model. The ground truth is compared with a set of 10 synthetic samples.

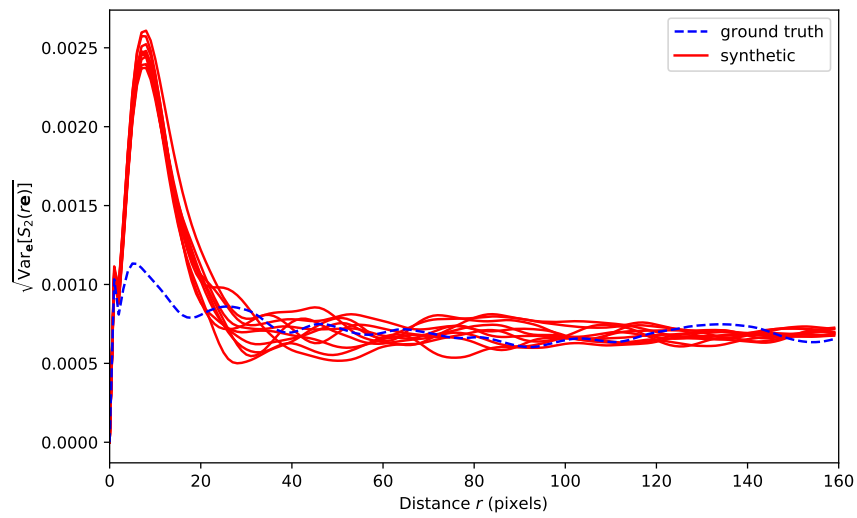
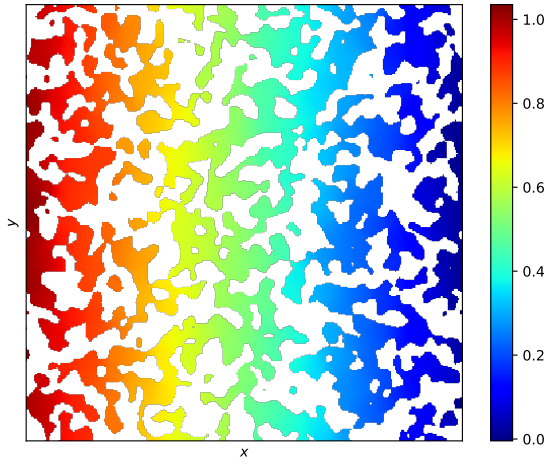
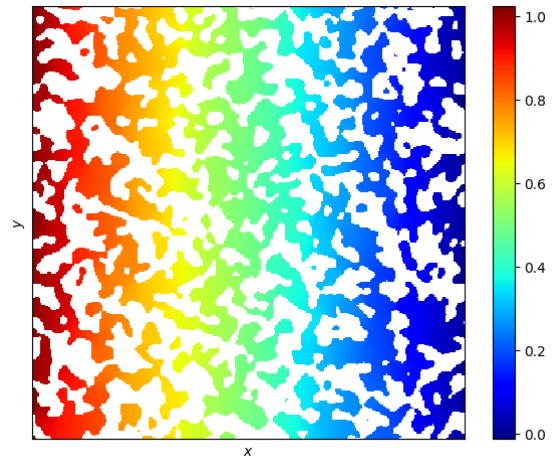


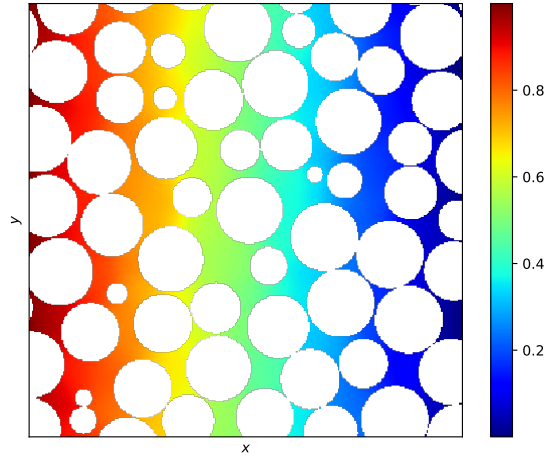
Figure 4.13: Validation of the isotropy for the *berea* model. For each sample, we plot the deviation of the two-point correlation with respect to direction, for each fixed distance: $\sigma_{S_2}(r) = \sqrt{\text{Var}_{\mathbf{e}}[S_2(r\mathbf{e})]}$.



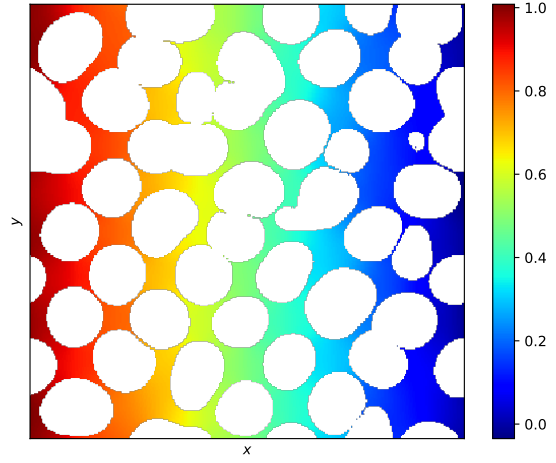
(a) *blobs*, ground truth (periodic).



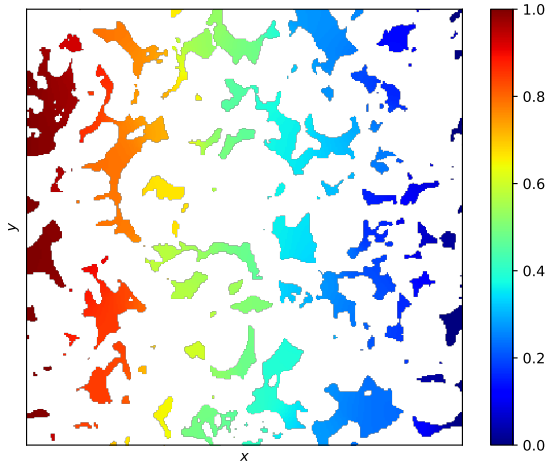
(b) *blobs*, generated sample (periodic).



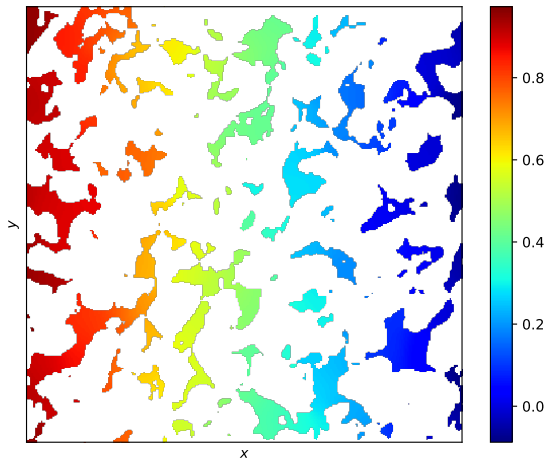
(c) *beadpack*, ground truth (non-periodic).



(d) *beadpack*, generated sample (periodic).

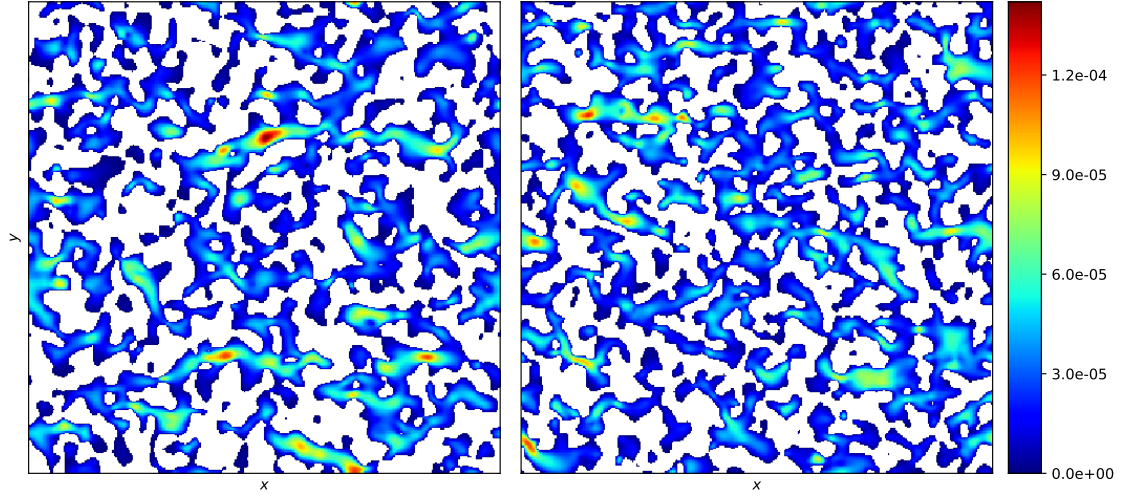


(e) *berea*, ground truth (non-periodic).

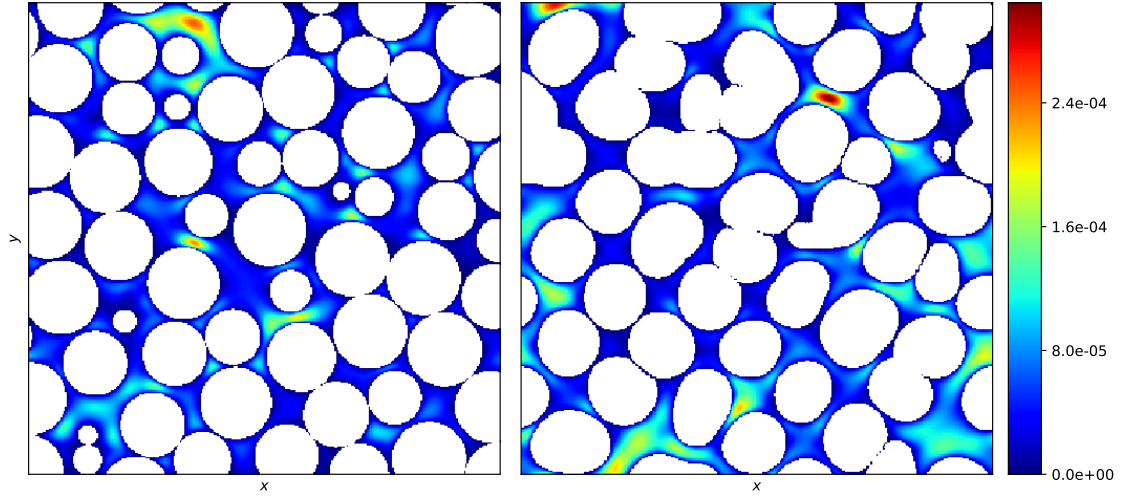


(f) *berea*, generated sample (periodic).

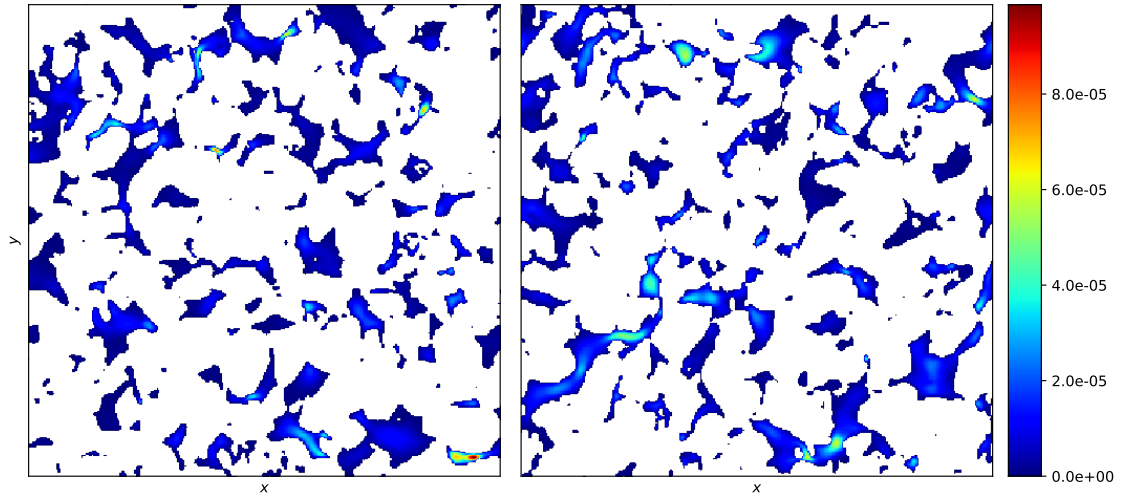
Figure 4.14: Cross-section of the electric potential field $\varphi/\Delta\varphi$ in the pore phase of several 3D samples.



(a) *blobs*, ground truth (left) and generated sample (right).



(b) *beadpack*, ground truth (left) and generated sample (right).



(c) *beroa*, ground truth (left) and generated sample (right).

Figure 4.15: Cross section of the dimensionless velocity norm $\frac{\mu}{L\Delta p}\|\mathbf{v}\|$ in the pore phase of several 3D samples.

Chapter 5

Conclusion and Future Work

This work has shown that *texture networks* are a valuable tool to tackle the *porous media reconstruction* task. We have presented the solution to the solid texture synthesis problem proposed by [Gut+20] as a way to generate porous media samples, given a single slice of the porous material. The visual results obtained in chapter 3 show that a pretrained, deep neural network provides a powerful description of the porous geometry, which allows to reconstruct three-dimensional samples that are plausible. In chapter 4 we have defined a set of metrics to quantitatively assess the quality of the reconstruction procedure, and provided the useful numerical schemes to compute these metrics. Our results show good agreement between the ground truth medium and its reconstructions, both for statistical measures and physical properties.

The discussion of the results has revealed that there is room for improvement:

- We have shown that our solution is not able to *exactly* match the porosity and the two-point correlation, which can be inferred from the training image. We could well imagine incorporating these statistical measures directly into the generator’s loss function, in order to achieve a perfect match.
- Our finite-difference solvers for the electrical and fluid flows limit the size and resolution of the generated samples due to their simplicity. This limitation could be alleviated by resorting to more sophisticated methods better suited to handle the complex pore geometries, e.g., finite element methods. Increasing the size of the samples would improve their representativeness and hence provide a better estimate of the error committed by the reconstruction.
- We have seen that the generator sometimes fails at producing isotropic volumes. We could improve this by comparing slices, during the training phase, in arbitrary directions, using interpolation.

This new framework for porous media reconstruction opens up numerous interesting research directions.

Recently, it has been demonstrated that deep convolutional neural networks based on representing information in a latent space enables to perform smooth interpolation in the image space, by interpolating in that space. The proposed generator network allows this, and we could

therefore imagine *interpolating between different porous media*. This idea could find interesting applications, such as reconstructing heterogeneous porous media by interpolating between the different textures, or quantifying uncertainties in a digital rock physics workflow.

The descriptor network was not specifically trained to describe porous media. Instead of using an off-the-shelf descriptor, one could use the idea of generative adversarial networks (GANs) and try to train a discriminator network in an adversarial way, i.e. a network that discriminates between fake slices (taken from generated samples) and real slices (taken from the training image).

Porous media reconstruction from two-dimensional slices is necessary because high-resolution information is only available in two-dimensional images. At a coarser scale, three-dimensional imaging using micro-CT is the way to go. To generate samples that account for the large range of scales involved in porous media, we could design a training scheme that trains the coarser layers using three-dimensional ground truth images, like in [MDB17], and train the finer layers using our slice-based approach. In this way, we would be able to generate large high-resolution samples that are correct at many different scales.

Bibliography

- [And+13] Heiko Andrä et al. “Digital rock physics benchmarks—part II: Computing effective properties”. en. In: *Computers & Geosciences*. Benchmark problems, datasets and methodologies for the computational geosciences 50 (Jan. 2013), pp. 33–43. ISSN: 0098-3004. DOI: 10.1016/j.cageo.2012.09.008.
- [Ara+20] Mauricio Araya-Polo et al. “Deep learning-driven permeability estimation from 2D images”. en. In: *Computational Geosciences* 24.2 (Apr. 2020), pp. 571–580. ISSN: 1573-1499. DOI: 10.1007/s10596-019-09886-9.
- [Bat00] G. K. Batchelor. *An Introduction to Fluid Dynamics*. Cambridge Mathematical Library. Cambridge: Cambridge University Press, 2000. ISBN: 978-0-521-66396-0. DOI: 10.1017/CB09780511800955.
- [Blu17] Martin J. Blunt. *Multiphase Flow in Permeable Media: A Pore-Scale Perspective*. en. Cambridge University Press, Feb. 2017. ISBN: 978-1-316-86188-2.
- [Che] Long Chen. “Fast Solvers for Stokes Equations”. en. In: (), p. 16.
- [Cli18] Intergovernmental Panel on Climate Change. *Global Warming of 1.5° C: An IPCC Special Report on the Impacts of Global Warming of 1.5° C Above Pre-industrial Levels and Related Global Greenhouse Gas Emission Pathways, in the Context of Strengthening the Global Response to the Threat of Climate Change, Sustainable Development, and Efforts to Eradicate Poverty*. Tech. rep. 2018.
- [De +16] Vincent De Andrade et al. “Nanoscale 3D imaging at the Advanced Photon Source”. In: *SPIE Newsroom* (May 2016). DOI: 10.1117/2.1201604.006461.
- [GEB15] Leon Gatys, Alexander S Ecker, and Matthias Bethge. “Texture Synthesis Using Convolutional Neural Networks”. In: *Advances in Neural Information Processing Systems* 28. Ed. by C. Cortes et al. Curran Associates, Inc., 2015, pp. 262–270.
- [Gos+19] Jeff T. Gostick et al. “PoreSpy: A Python Toolkit for Quantitative Analysis of Porous Media Images”. en. In: *Journal of Open Source Software* 4.37 (May 2019), p. 1296. ISSN: 2475-9066. DOI: 10.21105/joss.01296.
- [Gut+20] J. Gutierrez et al. “On Demand Solid Texture Synthesis Using Deep 3D Networks”. en. In: *Computer Graphics Forum* 39.1 (2020), pp. 511–530. ISSN: 1467-8659. DOI: 10.1111/cgf.13889.
- [Hec92] ROBERT Hecht-nielsen. “III.3 - Theory of the Backpropagation Neural Network**Based on “nonindent” by Robert Hecht-Nielsen, which appeared in Proceedings of the International Joint Conference on Neural Networks 1, 593–611, June 1989. © 1989 IEEE.” en. In: *Neural Networks for Perception*. Ed. by Harry Wechsler. Academic Press, Jan. 1992, pp. 65–93. ISBN: 978-0-12-741252-8. DOI: 10.1016/B978-0-12-741252-8.50010-8.
- [IGT09] Pavel Iassonov, Thomas Gebrenegus, and Markus Tuller. “Segmentation of X-ray computed tomography images of porous materials: A crucial step for characterization and quantitative analysis of pore structures”. en. In: *Water Resources Research* 45.9 (2009). ISSN: 1944-7973. DOI: 10.1029/2009WR008087.
- [IS15] Sergey Ioffe and Christian Szegedy. “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”. In: *arXiv:1502.03167 [cs]* (Mar. 2015).

- [KB17] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *arXiv:1412.6980 [cs]* (Jan. 2017).
- [LA87] Peter J. M. van Laarhoven and Emile H. L. Aarts. “Simulated annealing”. en. In: *Simulated Annealing: Theory and Applications*. Ed. by Peter J. M. van Laarhoven and Emile H. L. Aarts. Mathematics and Its Applications. Dordrecht: Springer Netherlands, 1987, pp. 7–15. ISBN: 978-94-015-7744-1. DOI: 10.1007/978-94-015-7744-1_2.
- [LP11] Annika Lang and Jürgen Potthoff. “Fast simulation of Gaussian random fields”. en. In: *Monte Carlo Methods and Applications* 17.3 (Sept. 2011), pp. 195–214. ISSN: 1569-3961, 0929-9629. DOI: 10.1515/mcma.2011.009.
- [LT92] Binglin Lu and S. Torquato. “Lineal-path function for random heterogeneous materials”. In: *Physical Review A* 45.2 (Jan. 1992), pp. 922–929. DOI: 10.1103/PhysRevA.45.922.
- [MBB13] Peyman Mostaghimi, Martin J. Blunt, and Branko Bijeljic. “Computations of Absolute Permeability on Micro-CT Images”. en. In: *Mathematical Geosciences* 45.1 (Jan. 2013), pp. 103–125. ISSN: 1874-8953. DOI: 10.1007/s11004-012-9431-4.
- [MDB17] Lukas Mosser, Olivier Dubrulle, and Martin J. Blunt. “Reconstruction of three-dimensional porous media using generative adversarial neural networks”. In: *Physical Review E* 96.4 (Oct. 2017), p. 043309. ISSN: 2470-0045, 2470-0053. DOI: 10.1103/PhysRevE.96.043309.
- [ML14] Gregoire Mariethoz and Sylvain Lefebvre. “Bridges between multiple-point geostatistics and texture synthesis: Review and guidelines for future research”. en. In: *Computers & Geosciences* 66 (May 2014), pp. 66–80. ISSN: 0098-3004. DOI: 10.1016/j.cageo.2014.01.001.
- [OB04] Hiroshi Okabe and Martin J. Blunt. “Prediction of permeability for porous media reconstructed using multiple-point statistics”. In: *Physical Review E* 70.6 (Dec. 2004), p. 066135. DOI: 10.1103/PhysRevE.70.066135.
- [Oqu+14] Maxime Oquab et al. “Learning and Transferring Mid-level Image Representations Using Convolutional Neural Networks”. In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. June 2014, pp. 1717–1724. DOI: 10.1109/CVPR.2014.222.
- [Pet19] British Petroleum. “BP Statistical Review of World Energy Report”. In: *BP London, UK* (2019).
- [Qui84] Jacques A. Quiblier. “A new three-dimensional modeling technique for studying porous media”. en. In: *Journal of Colloid and Interface Science* 98.1 (Mar. 1984), pp. 84–102. ISSN: 0021-9797. DOI: 10.1016/0021-9797(84)90481-8.
- [Rob97] Anthony P. Roberts. “Statistical reconstruction of three-dimensional porous media from two-dimensional images”. In: *Physical Review E* 56.3 (Sept. 1997), pp. 3203–3212. DOI: 10.1103/PhysRevE.56.3203.
- [Sah11] Muhammad Sahimi. *Flow and Transport in Porous Media and Fractured Rock: From Classical Methods to Modern Approaches*. en. John Wiley & Sons, May 2011. ISBN: 978-3-527-63670-9.
- [Sax+17] Nishank Saxena et al. “References and benchmarks for pore-scale flow simulated using micro-CT images of porous media and digital rocks”. en. In: *Advances in Water Resources* 109 (Nov. 2017), pp. 211–235. ISSN: 0309-1708. DOI: 10.1016/j.advwatres.2017.09.007.
- [SZ15] Karen Simonyan and Andrew Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *arXiv:1409.1556 [cs]* (Apr. 2015).
- [Uly+16] Dmitry Ulyanov et al. “Texture Networks: Feed-forward Synthesis of Textures and Stylized Images”. In: *arXiv:1603.03417 [cs]* (Mar. 2016).
- [Vol+19] Denis Volkhonskiy et al. “Reconstruction of 3D Porous Media From 2D Slices”. In: *arXiv:1901.10233 [cs]* (Nov. 2019).
- [Vor92] H. A. van der Vorst. “Bi-CGSTAB: A Fast and Smoothly Converging Variant of Bi-CG for the Solution of Nonsymmetric Linear Systems”. In: *SIAM Journal on Scientific and Statistical Computing* 13.2 (Mar. 1992), pp. 631–644. ISSN: 0196-5204. DOI: 10.1137/0913035.

- [YT98] C. L. Y. Yeong and S. Torquato. “Reconstructing random media. II. Three-dimensional media from two-dimensional cuts”. In: *Physical Review E* 58.1 (July 1998), pp. 224–233. DOI: 10.1103/PhysRevE.58.224.
- [ZYM18] Yufeng Zheng, Clifford Yang, and Aleksey Merkulov. “Breast cancer screening using convolutional neural network and follow-up digital mammography”. In: May 2018, p. 4. DOI: 10.1117/12.2304564.

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain

Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | www.uclouvain.be/epl