Journal of Signal Processing Systems
https://doi.org/10.1007/s11265-021-01701-y

Multivariate multifractal texture DCGAN synthesis: How well does it work ? How does one know ?

Patrice Abry[1] · Vincent Mauduit[1] · Emmanuel Quemener[2] · Stephane Roux[1]

Received: 20 February 2021 / Revised: 21 August 2021 / Accepted: 26 August 2021

Abstract
In a recent past, Deep Learning emerged as a standard tool in Image Processing, commonly involved in numerous and various tasks. Notably, Deep Learning has become increasingly popular for the synthesis of images in several applications different in nature. However, research efforts have been massively focused on designing new and increasingly complex architectures to achieve yet better performance, often at the price of overlooking the uneasy question of the assessment of the quality of the synthesized images. Focusing on the specific context of pure textures, i.e., of images with no geometrical contents, the present work aims to propose a methodology that permits to quantify the quality of Deep Learning synthesized images. It makes use of Deep Convolutional Generative Adversarial Networks, a specific class of trained neural networks, commonly used for image synthesis. Because they provide versatile and well-documented texture models, multivariate multifractal fields, with rich multiscale cross-statistics (scale-free and multifractal textures), are used. A posteriori synthesis quality indices are defined from the statistics of multiscale (wavelet) representations computed on deep learning generated multivariate textures and compared to those associated with the models. These comparisons permit to objectively quantify the quality of deep learning texture synthesis as well as the reproducibility of the training and learning procedures, an approach that departs from reporting only the training yielding best performance. This methodology further permits to quantify objectively the variation in the quality of deep learning generated multivariate textures with respect to the complexity of deep learning architectures. Moreover, a priori indices, constructed directly on loss functions, hence much easier to compute, are also proposed and shown to correlate significantly with the a posteriori and costly multiscale representation synthesis quality indices.

Keywords Generative Adversarial Network · Multivariate Texture Synthesis · Quality Assessment · Multifractals · Multiscale Statistics · Loss Functions

1 Introduction

1.1 Context

In the last decade, Deep Learning architectures have been massively used to produce synthetic images supposed to match target images from numerous real world applications, very different in nature (cf. e.g., [12, 56]). Potential benefits in applications triggered a tremendous amount of research works and publications across the world, often reporting outstanding results based on the empirical observation of *realistic images*. Despite indisputable promising results, the use of Deep Learning for image synthesis remains a scientific field with far more open questions than certainties and final answers. Beyond the obvious questions related to deep learning architecture design or choice in relation to a specified task, there are less obvious questions related to performance assessment: How does one quantify how well deep learning is performing in image synthesis ? Or even more simply, is it actually successful ?

Often in applications, targeted image are characterized by a strong *geometrical* or *cartoon* contents (edges, shapes, angles, objects, ...), such as e.g., faces, natural or urban landscapes,[21, 29, 31, 46, 47], so that synthesis *performance* are

Work supported by CBP (Blaise Pascal Center) with the use of SIDUS (Single Instance Distributing Universal System), by ACADEMICS Grant, under IDEXLYON project, within PIA ANR-16-IDEX-0005 and by ANR-16-CE33-0020 MultiFracs Grant.

✉ Vincent Mauduit
patrice.abry@ens-lyon.fr

[1] Univ Lyon, ENS de Lyon, Univ Claude Bernard, CNRS, Laboratoire de Physique, Lyon, France

[2] Univ Lyon, ENS de Lyon, Univ Claude Bernard, Centre Blaise Pascal, Lyon, France

often only estimated qualitatively as *satisfactory* by visual inspection, i.e., a synthetic face looks like a real-person face... (see, a contrario, [24, 45]). However, in applications, images of interest actually consists of both *cartoon* contents and *texture* information (patterns, periodicity,...). It is thus not always easy to disentangle whether performance stem from geometry or texture. Furthermore, images consisting of texture information only are of major importance in numerous applications, some with potential high societal stakes, such as e.g., biomedical applications [14], remote sensing [40] or art investigations [3]. While it is sometimes considered that *intrinsic dimensions* are larger for textured images than for geometric ones, hence potentially inducing larger difficulties for Deep Learning procedures [10], the assessment of the quality of Deep Learning texture synthesis has been much less considered.

Often overlooked, because of a possibly more rewarding strategies in contributing to a race towards always more complicated architectures for better performance, the fundamental question of investigating quantitative assessment of Deep-Learning-synthetized *texture only* images constitutes the focus of the present work.

1.2 Related Works

Several different types of neural network architectures were used for image synthesis. Convolutional neural networks can be used for image synthesis (cf. e.g., [17, 48]), by applying an *inversion* procedure, which has, however, been shown to be poorly stable and significantly lacks theoretical foundations. Autoencoders were also used for image synthesis [6, 19], based on latent variable representations. Autoencoder-based procedures were however reported to suffer from significant drawbacks, such as large artifacts, lack in high-frequency information and weak ability to generate new instances of images [19]. Generative Adversarial Networks (GAN) attempt to overcome these limitations. GAN were successfully used across an ever increasing variety of architectures (BigGAN, StyleGAN, SGAN, PSGAN,...) [12, 13, 29, 30, 56] and in numerous application fields with state-of-the-art results [7, 13, 20, 42], which motivates their use in the present work.

While the assessment of the quality of generated images is often simply conducted by visual inspection (cf. e.g. [45]), quantitative performance metrics were also proposed, based on information theoretic measures computed on neural network predictions [24, 45]. Such scores are often based on descriptive statistics (mean and covariance) that mix up geometry and texture contributions, making it hard to disentangle the respective contributions. Even if applied to texture-only, these first and second order statistics are well suited for Gaussian texture, but by construction overlook departures from Gaussianity that may be critical, e.g., in some medical applications. To finish with, image synthesis is often considered for univariate images only: each instance of synthesized image only consist of a single image, whereas in numerous applications (colored images, remote sensing images, multimodal images,...), there is a significant interest to produce multivariate synthetic images, i.e., each sample consists of a vector of images.

The Deep Learning based synthesis of multivariate non Gaussian textures, and the assessment of the synthesis that also accounts for cross-statistics, thus remains an open question, of critical importance in applications to quantify efficiency, robustness and reproducibility, issues at the heart of the present contribution.

1.3 Goals, Contributions and outlines

The present contribution aims to address the generic question of quantifying if and how well GAN architectures perform for multivariate texture synthesis. To that end, two important methodological choices are made. First, works in Deep Learning often seek to contributing to (and winning) the competition for an always *best* architecture. Instead, we will make use here of classical and well-documented Deep-Convolutional (DC) GAN architectures, so as to put the focus on quantifying if and and how well they do, and how reproducible performance are from one training to the other. Second, Deep Learning contributions often make use of one of the numerous real-world databases of images, with usually impressively large size, yet with unknown precise and prior knowledge of their statistics and characteristics. Instead, the originality of the present work consists in using synthetic multivariate texture models with rich and complex but known and controlled properties and statistics. This will provide us with a mean to quantify the quality and variability of the DCGAN synthesis procedures by comparing the statistics of multivariate textures produced by the trained DCGAN architectures to those of the theoretical models.

This work expands in several ways described below on a preliminary work presented in [39], in which, we used a single DCGAN architecture, and showed that synthesis quality performance was suffering from a wide variability across independent trainings, despite identical and well-controlled training conditions. We further showed that this significant variability was not reduced by modifying the training set size or the number of iterations in the training stage. Capitalizing on these results, training set size and number of iterations are kept fixed in the present work. Instead, we explore here the impact of varying DCGAN architecture complexities, in terms of depth (number of layers) or size (number of parameters), on the quality of the multivariate texture synthesis and on the variability across independent training of achieved performance.

We make use of multivariate multifractal textures, widely used in applications very different in nature ranging from medicine to art [3, 4, 15, 32, 36, 51, 52], and of interest per se, but chosen here only as archetypal models of non Gaussian multivariate textures with theoretically known and tunably intricate (cross)-statistics, beyond (cross-)covariance. The properties and statistics of such multivariate multifractal textures can be well quantified using multiscale wavelet and wavelet leader analyses [27, 28, 34, 54]. These multiscale analyses are used to design *a posteriori* assessment indices permitting to quantify the quality of DCGAN synthesized multivariate textures, by comparisons to groundtruth and thus to assess the reproducibility of the training procedure. The definitions, properties and synthesis procedures, together with the wavelet-based analysis procedure are detailed in Sections 2.1 and 2.2.

The DCGAN principle, the different architectures of increasing complexity are described in Section 3, together with the GPU facilities and optimization and training procedures.

The synthesis quality and its variability is assessed by comparing performance across repeated independent training trials, restarted from scratch, rather than reporting only the best trial performance, another originality of the present work (cf. Sections 4.1 and 4.2).

Further, we propose and assess the relevance of *a priori* quality indices, constructed directly from the loss functions computed while the training of DCGAN architectures is performed, by quantifying their correlations with the wavelet-based indices that can be computed a posteriori only, that is after the training phase is completed and the DCGAN multivariate textures are generated. (cf. Section 4.3).

These results, reported in Section 4, are further discussed and commented in Section 5.

2 Multivariate multifractal textures

2.1 Multivariate multifractal Models

After Mandelbrot's seminal work [38] in the field of hydrodynamic turbulence, Multifractal random walks (MRW) [8] are nowadays used as versatile models for numerous real-world applications, where the time series or textures actually measured by sensors display multiscale scale-free temporal or spatial dynamics. Interested readers are referred to e.g., [5, 26] for a review of the scale-free paradigm and of its use in applications.

In the present work, we extend the definition of bivariate MRW 1D signals, proposed in [52], to multivariate MRW textures (referred to as MMRW) as:

$$\forall m = 1, \dots, M,\ X_m(\underline{x}) = G_m(\underline{x}) e^{\omega_m(\underline{x})}, \tag{1}$$

with $\underline{x}$ the spatial cartesian coordinates, and where $\underline{G}(\underline{x}) = \{G_1(\underline{x}), \dots, G_M(\underline{x})\}$ and $\underline{\omega}(\underline{x}) = \{\omega_1(\underline{x}), \dots, \omega_M(\underline{x})\}$ consist of two independent zero-mean M-variate Gaussian 2D-fields with prescribed covariance functions.

The process $\underline{G}(\underline{x})$ is defined as the multivariate extension of the univariate 2D fractional Gaussian noise (2D-fGn), the reference Gaussian model for scale-free textures [43]. Being Gaussian, $\underline{G}(\underline{x})$ is fully defined by its $M \times M$ covariance functions. Their detailed analytical closed-form expressions can be found in [1]. For the purpose of the present work, it is sufficient to know that these covariance functions are fully controlled by M Hurst exponents $\underline{H} = (H_1, \dots, H_M)$ and a $M \times M$ pointwise covariance matrix Σ.

The process $\underline{\omega}(\underline{x})$ is defined via $M \times M$ cross-covariance functions, designed to induce multifractality in the spatial statistics. The component (m_1, m_2) of these cross-covariance functions reads:

$$\{\Sigma_{mf}\}(m_1, m_2) = \rho_{mf}(m_1, m_2)\lambda_{m_1}\lambda_{m_2} \log\left(\frac{L}{||\underline{x}|| + 1}\right), \tag{2}$$

for $||\underline{x}|| \leq L$ and 0 otherwise, with L an arbitrary integral scale. In this definition, $\underline{\Lambda} = (\lambda_1, \dots, \lambda_M)$ denotes a vector of parameters controlling the multifractality of each component independently ρ_{mf}., while ρ_{mf} corresponds to a $M \times M$ correlation matrix that controls cross-multifractalities amongst components.

Extending to 2D fields, the calculations made for 1D signals in univariate [8] and bivariate [5, 33, 53] settings, enables us (calculations not detailed here) to understand and control the complexity of the cross-statistics of the such M-component textures. Together, the selfsimilarity vector $\underline{H}$ controls the second-order covariances (or selfsimilarity) of the M-components of $X_m(\underline{x})$, while the pointwise matrix Σ modulates the intensity of the cross-covariances (or cross-selfimilarity) amongst components. Additionnaly and independently, the multifractality vector $\underline{\Lambda}$ controls the departures from Gaussianity of each component of $X_m(\underline{x})$ thus higher order statistics, and the pointwise matrix ρ_{mf} modulates the cross-multifractality amongst components, and hence the cross-higher order statistics. This is why ρ_{mf} is abusively but conveniently referred to as *multifractal correlation*.

MMRW thus provide practitioners with versatile models, permitting the independent tuning of the spatial dynamics of each component, via the choices of $\underline{H}$ and $\underline{\Lambda}$, from very simple (no-similarity and no-multifractality) to very intricate (self-similarity and multifractality). In addition and independently from the components themselves, the cross-statistics between components can be tuned, via the choices of Σ and

ρ_{mf}, from independence (Σ and ρ_{mf} are identity matrices), to second-order correlation only (Σ departs from identity but ρ_{mf} remains the identity matrix) and higher-order intricate statistical dependencies (both Σ and ρ_{mf} depart from identity).

MMRW textures are synthesized using dedicated codes, designed by ourselves and available upon request, implementing directly definitions via the circulent embedded matrix technique classically used to synthesize Gaussian processes [22, 23]. One sample of 4-variate MMRW is shown in Fig. 1. These MMRW synthesized from the theoretical model are hereafter referred to as true MMRW.

2.2 Multivariate multifractal Analysis

It is well-documented that wavelet representations constitute reference tools to analyze textures with scale-free (selfsimilarity and multifractality) spatial dynamics [43, 54].

The discrete wavelet transform coefficients of the M-variate texture X, $\{d_{X_m}(j,\underline{k})\}$, $m = 1,\dots,M$, are classically defined via inner products between each component X_m and dilated (at scale 2^j) and translated (at location $2^j\underline{k}$) templates of a tensor-product based 2D wavelet (see e.g., [37] for details).

It can be easily shown that, for any pair of components $X_{m_1}(\underline{x})$ and $X_{m_2}(\underline{x})$, the wavelet covariance matrix at scale 2^j, $S_{m_1,m_2}(2^j)$, behave as a power-law with respect to scales 2^j:

$$S_{m_1,m_2}(2^j) \equiv \sum_k d_{X_{m_1}}(j,k) d_{X_{m_2}}(j,k) \simeq \Sigma_{m_1,m_2} C_\psi 2^{j\zeta_{m_1,m_2}}, \tag{3}$$

with C_ψ a constant that depends on the chosen wavelet and on $H_{m_1}, H_{m_2}, \lambda_{m_1}, \lambda_{m_2}$, and $\zeta_{m_1,m_2} = H_{m_1} + H_{m_2} - (\lambda_{m_1}^2 + \lambda_{m_2}^2)/2$.

The wavelet coherence (or crosscorrelation) functions,

$$C^{(ss)}_{m_1,m_2}(2^j) \equiv S_{m_1,m_2}(j) / \sqrt{S_{m_1,m_1}(j) S_{m_2,m_2}(j)} \tag{4}$$

$$= \frac{\Sigma_{m_1,m_2} C_\psi(H_{m_1}, H_{m_2}, \lambda_{m_1}, \lambda_{m_2})}{\sqrt{\Sigma_{m_1,m_1} C_\psi(H_{m_1}, H_{m_1}, \lambda_{m_1}, \lambda_{m_1}) \Sigma_{m_2,m_2} C_\psi(H_{m_2}, H_{m_2}, \lambda_{m_2}, \lambda_{m_2})}}, \tag{5}$$

$$= \frac{\Sigma_{m_1,m_2}}{\sqrt{\Sigma_{m_1,m_1} \Sigma_{m_2,m_2}}}, \tag{6}$$

are thus constant across scales, with levels that depend only on the point-covariance matrix Σ. Therefore, the functions $S_{m_1,m_2}(j)$ and $C^{ss}_{m_1,m_2}(j)$ only probe the second order statistics of MMRW and thus do not quantify multifractality. Notably they are blind to cross-multifractalities (off-diagonal terms in Σ_{mf}).

To measure higher-order statistics and multifractality, wavelet leaders $L(j,k)$ have been further constructed, as local suprema of wavelet coefficients, taken over finer scales and within a short spatial neighborhood $3\lambda_{j,k}$, with $\lambda_{j,k} = [k2^j, (k+1)2^j)$ the dyadic interval of size 2^j and $3\lambda_{j,k}$ the union of $\lambda_{j,k}$ with its 2 neighbors [50, 54]:

$$L(j,k) = \sup_{\lambda' \subset 3\lambda_{j,k}} |d_X(\lambda')|. \tag{7}$$

Extending calculations in [52], it can then be shown that the first order and second order cumulants of $\ln L$, $C_1^{(m)}(2^j)$ and $C_2^{(m_1,m_2)}(2^j)$, behave linearly in $\ln 2^j$:

$$C_1^{(m)}(2^j) = c_1^{(0,m)} + c_1^{(m)} \ln 2^j, \tag{8}$$

$$C_2^{(m_1,m_2)}(2^j) = c_2^{(0,m_1,m_2)} + c_2^{(m_1,m_2)} \ln 2^j, \tag{9}$$

with $c_1^{(m)} = H_m + \lambda_m^2/2$ and $c_2^{(m_1,m_2)} = \rho_{mf}(m_1,m_2)\lambda_{m_1}\lambda_{m_2}$.

This permits to define a wavelet multifractal coherence function as:

$$C^{(mf)}_{m_1,m_2}(2^j) \equiv C_2^{(m_1,m_2)}(2^j) / \sqrt{C_2^{(m_1,m_1)}(2^j) C_2^{(m_2,m_2)}(2^j)}. \tag{10}$$

While $C^{(mf)}_{m_1,m_2}(2^j)$ has no explicit closed-form analytical expression, Eq. (8) shows that its dependence across scales

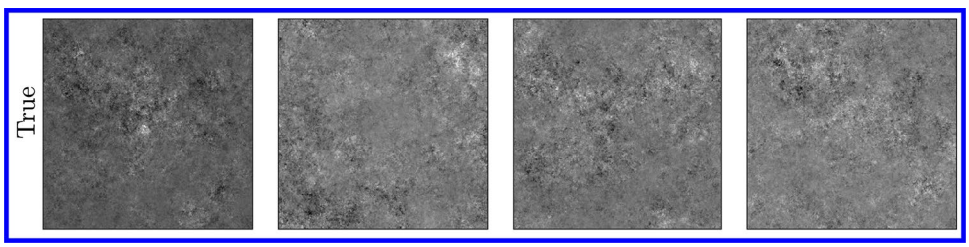

Figure. 1 Sample of a 4-variate textures with (identical) selfsimilarity and multifractality in the spatial dynamics of each component as well as intricate (and different) cross-selfsimilarities and cross-multifractalities amongst pairs of omponents. The parameters defining the MMRW model from which this 4-variate sample is generated are detailed in Section 4.1 (see also the caption of Fig. 2).

is essentially controlled by the multifractal correlation $\rho_{mf}(m_1, m_2)$.

These wavelet and wavelet-leader based multiscale functions $S_{m_1,m_2}(2^j)$, $C_1^{(m)}(2^j)$, $C_2^{(m_1,m_2)}(2^j)$, $C_{m_1,m_2}^{(ss)}(2^j)$ and $C_{m_1,m_2}^{(mf)}(2^j)$ characterize the dependencies as functions of scales 2^j of the statistics of X: $S_{m,m}(2^j)$ and $C_2^{(m,m)}(2^j)$ account respectively for the selfsimilarity and multifractality of each component independently ; $C_{m_1,m_2}^{(ss)}(2^j)$ characterize cross-selfsimilarity, hence cross-covariances amongst components, while $C_{m_1,m_2}^{(mf)}(2^j)$ describes cross-multifractality, hence, spatial dependencies amongst components beyond second order and cross-covariance.

Fig. 2 displays the multiscale representations $\log_2 S_{m,m}(2^j)$, $C_1^{(m)}(2^j)$, $C_2^{(m,m)}(2^j)$ (left plots) and $C_{m_1,m_2}^{(ss)}(2^j)$ and $C_{m_1,m_2}^{(mf)}(2^j)$ (right plots), obtained as averages over 100 independent 4-variate realizations of the MMRW (one 4-variate sample being illustrated in Fig. 1). As theoretically predicted, the wavelet coherence $C_{m_1,m_2}^{(ss)}(2^j)$ functions are constant across scales 2^j, at levels prescribed only by the off-diagonal entries of matrix Σ. Interestingly, Fig. 2 shows that the functions $C_{m_1,m_2}^{(mf)}(2^j)$, for which no explicit closed-form analytical expressions are available, are close to constants across scales, with values closely related to $\rho_{mf}(m_1, m_2)$, an outcome of interest per se, nicely matching the intuition associated with scale-free statistics.

Wavelet and wavelet leader analyses are implemented using the toolbox, available and documented on the author websites.

These multiscale representations will be used to construct *a posteriori* indices to assess the quality of DCGAN synthesized textures by comparing those computed on true MMRW to those computed from DCGAN synthesized textures.

3 Neural network architectures, training set and training procedures

3.1 Neural network architectures

3.1.1 Deep-Convolutional generative Adversarial Network

Generative adversarial networks (GAN) constitute an increasingly popular class of deep learning architectures, originally proposed in [20]. GAN are constructed as a competition between two neural networks, obtained by piling up elementary layers gathering linear filtering and pointwise non linear transformations (cf. Fig 3 for illustration). The *Generator G* defines a nonlinear filter that generates *fake* images resembling *in some sense* target images, from white Gaussian noise as input. The *Discriminator*, *D*, outputs the probability that image *X*, used as input, belongs to the target distribution.

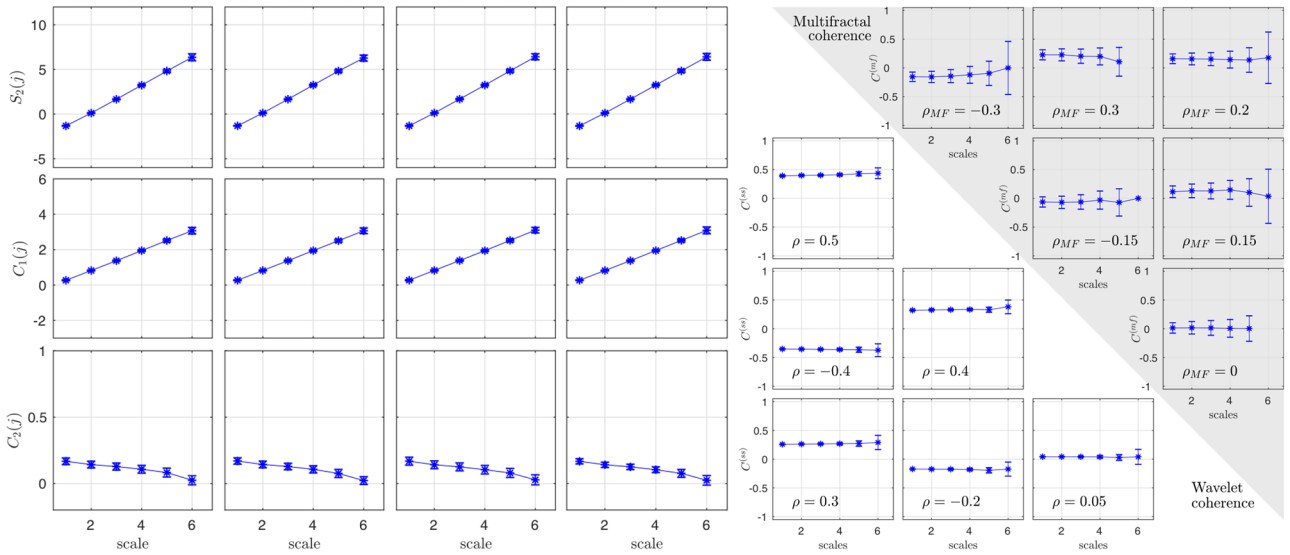

Figure. 2 Multiscale analysis of MMRW. The 3×4 left plots show, for each of the four components ($m = 1, 2, 3, 4$, in columns) the functions $\log_2 S_{m,m}(2^j)$, $C_1^{(m)}(2^j)$ and $C_2^{(m,m)}(2^j)$ as functions of $j = \log_2 2^j$. The right plots show, for each of the 6 pairs the wavelet coherence functions $C_{m_1,m_2}^{(ss)}(2^j)$ (bottom triangle) and the multifractal coherence function $C_{m_1,m_2}^{(mf)}(2^j)$ (upper triangle) as functions of $j = \log_2 2^j$. These functions are estimated empirically as averages over the multiscale analysis of 100 independent realizations of the 4-variate MMRW, with selfsimilarity and multifractality exponents chosen equal, $\forall m = 1, 2, 3, 4$, $H_m = 0.8$ and $\lambda_m = \sqrt{0.03}$, and off-diagonal entries of the correlation and multifractal correlation matrices Σ and Σ_{mf} as indicated on plots. (see also Section 4.1). These functions are used as ground truth to assess the quality of the DCGAN synthesis by comparisons of the same functions computed as averaged over the same analysis performed on DCGAN generated textures.

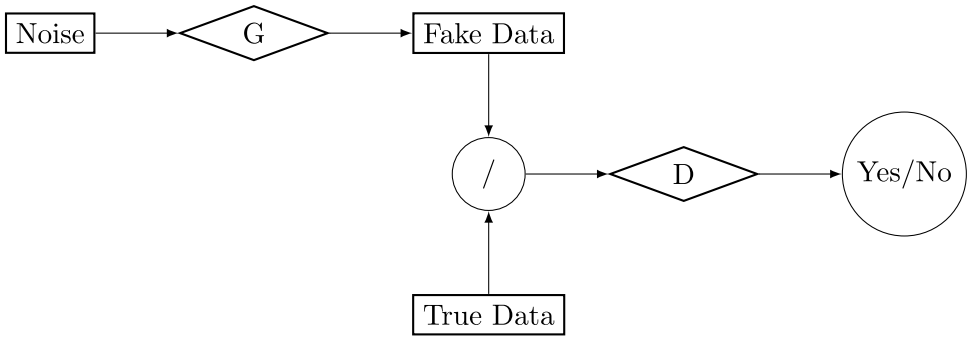

Figure. 3 Generative Adversarial Network Principle.

GAN implementations were traditionally based on fully-connected layers, followed by (nonlinear) activation functions and max-pooling to downsize images. However, the original GAN has been documented to be unstable and difficult to train. Numerous attempts to stabilize its optimization have been reported, e.g., [7, 42, 45].

Deep Convolutional GAN (DCGAN) were further proposed in [42] as a solution to unstable training. They consist in using convolutional (instead of fully-connected) layers. They rely on strategic architectural choices (e.g. batch normalization, strided convolutions, (leaky) ReLU activations) to achieve training convergence [42] detailed below.

3.1.2 Training and Loss Functions

The joint optimization of the Discriminator and Generator relies on the use of a training set, consisting of samples of the target distribution, and results from solving a MinMax problem based on Cross Binary Entropy as loss function [20]: The optimal discriminator and generator are obtained by respectively maximizing

$$L_D(\theta_D) = \frac{1}{m}\sum_{i=1}^{m}\left[\log(D(x)) + \log(1 - D(G(z)))\right],$$

$$\text{and } L_G(\theta_G) = \frac{1}{m}\sum_{i=1}^{m}\log(D(G(z))),$$

over the weights θ_D and θ_G respectively of D and G, and where x, z are samples of respectively the training set and white Gaussian noise.

3.1.3 Architectures

One aim of the work consists in testing if and how architecture complexity, within the DCGAN framework, impacts texture synthesis performance. To that end, the architecture of the Discriminator is kept fixed, while that of the Generator is varied from less to more complex.

The Discriminator consists of five convolutional layers, with strides (2, 2) and $3 \times 3 \times d$ convolutional filters followed by batch normalization (except for the first layer), Leaky ReLU activation functions and dropout. The depths d of the layers are indicated on the sketched representation of the Discriminator in Fig. 4 (bottom right). All together, the discriminator implies a total number of trainable parameters summing up to 169089.

The overall structure across Generators is preserved, with a white noise input of fixed and same size (100 samples) enhanced and reshaped by a fixed fully-connected layer and a ReLU activation, gathering 3309568 trainable parameters. This first layer is then followed by N Convolutional layers, each consisting of $3 \times 3 \times d$ convolutional filters, batch normalization, ReLU activations and possible upsampling by pixel duplication, except for the last layer consisting of 3×3 convolutional filters followed by tanh activation function. The depths d of the layers are indicated on the sketched representations of the Generators in Fig. 4.

Three Generator architectures are tested with $N = 3, 5$ or 7 convolutional layers, associated with respectively 24848, 39972 and 46404 additional trainable parameters, thus summing up to 3334052, 3349540 and 3455972 trainable parameters. These Generator architectures are sketched in Fig. 4.

3.2 Experimental set-up

3.2.1 GPU facilities

Optimization and learning are conducted on IT facilities, made available by The Blaise Pascal Center for research and educations. The largest technical bench is based on workstations equipped with large number of GPU (Graphical Processing Units) or GPGPU (General Purpose Graphical Processing Unit). All these hardware equipment run a unique Operating System: SIDUS (Single Instance Distributing Universal System) [41]. Provided by the network, SIDUS ensures the unicity of the distributed Operating System on all workstations: 2 machines that boot SIDUS at the same time cannot have a different operating system, bit to bit. On the 4 workstations used to perform Machine Learning

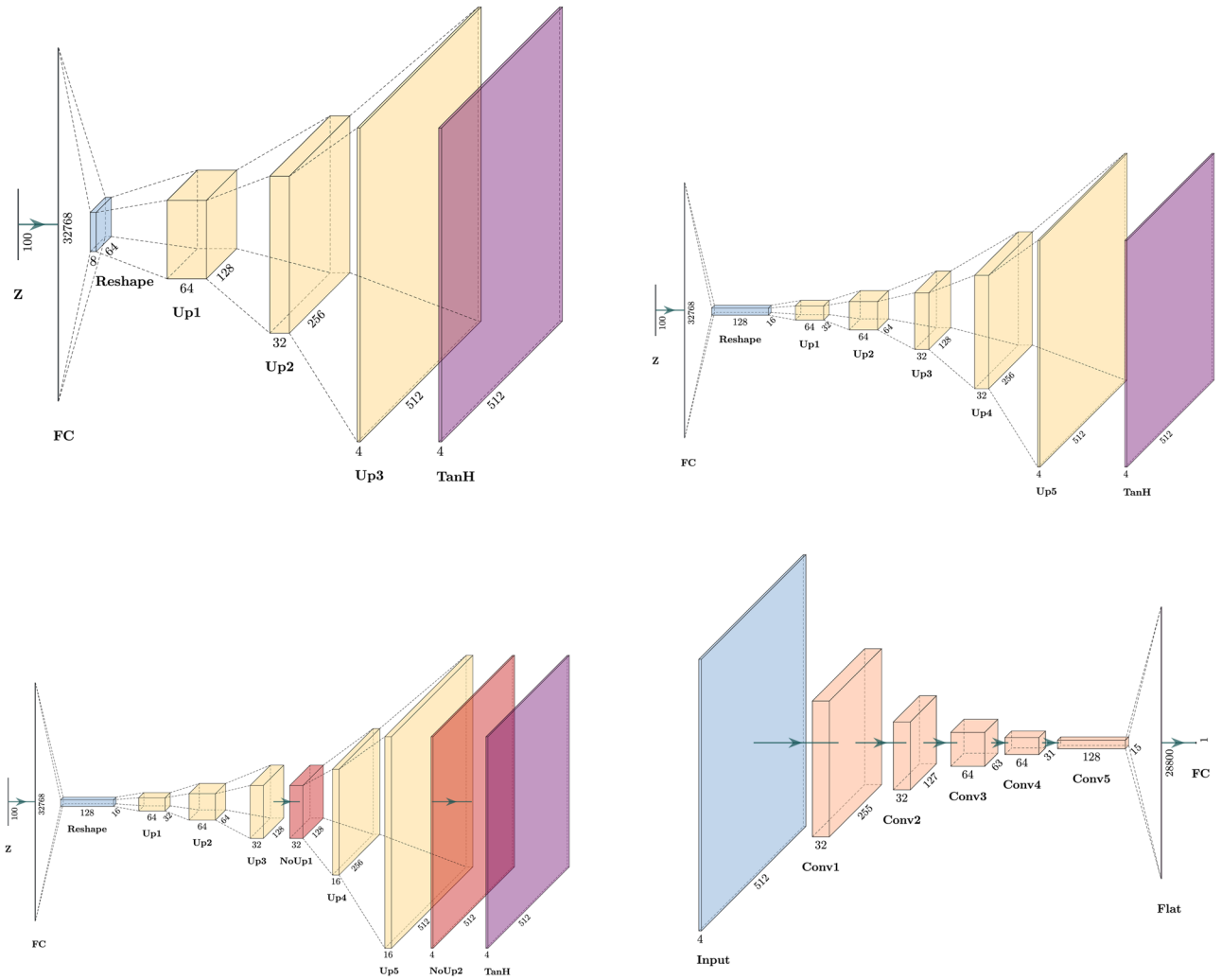

Figure. 4 DCGAN Architectures. Generators of increased complexities, with 3, 5 and 7 layers respectively and fixed Discriminator (bottom right). Architecture illustrations produced with the tools made available at https://github.com/HarisIqbal88/PlotNeuralNet.

operations, a specific Anaconda environment with all tools is used. It is launched on a Linux Debian Buster distribution with Nvidia backported packages. Each of these workstations have the same processor i7-4770K, the same amount of memory (32GB), the same mainboards with the same BIOS version, the same hard drive for local storage and, most important thing, the same GPU : Nvidia RTX 2080 Super.

3.2.2 Optimization and learning

DCGAN are implemented using Keras, following architectures proposed in [42]. From results obtained in our preliminary work [39], it was decided to set training parameters as follows. For optimization, Adam amsgrad variant is used, with learning rate of 2×10^{-4} and momentum of 0.5. Weights are initialized using uniform initialization [18]. The discriminator is trained using dropout at rate 1/4. Batch normalization is applied, which consists in normalizing independently the inputs of each network layer [25], with batch size of 32. The size of the input white Gaussian noise is set to 100 and the number of iterations in the training phases to 10000.

3.2.3 Performance reproducibility

To test the reproducibility and stability of the training, DCGAN training is repeated 17 times from scratch independently, for each architecture, under the exact same conditions (same training set, same architectures, same softwares, same operating system, same hardware). The performance and their variability are assessed from comparisons across these 17 independent training, an originality of the present work.

4 Synthesis quality quantitative assessment

4.1 Training dataset

From conclusions drawn obtained in our preliminary work [39], it was decided to construct the training dataset as 10000 independent samples of 4-variate (i.e, $M = 4$) MMRW textures, each of size 512×512. They are synthesized using Matlab routines devised by ourselves, implementing directly definitions via circulant matrix embedding [22].

Self-similarity and multifractality exponents are chosen equal ($\forall m = 1, 2, 3, 4$, $H_m = 0.8$ and $\lambda_m = \sqrt{0.03}$), so that all components have exactly the same statistics: Notably, they have the same marginal distributions, the same covariance functions and the same higher-order statistics ; they can hence not be distinguished one from the other.

The off-diagonal entries of the cross-selfsimilarity Σ and of the cross-multifractality Σ_{mf} matrices are however chosen different for each pair of components. Furthermore, some pairs of components have cross-selfsimilarity and cross-multifractality of same signs, some positive, some negative, while other pairs will have cross-selfsimilarity and cross-multifractality of opposite signs. This latter case corresponds to intricate cross-statistics, where the cross-selfsimilarity hence the cross-correlation is for example negative, yet with positive positive cross-multifractality and hence positive statistical dependencies beyond correlation, or conversely. We believe that these settings yield complicated multivariate textures, with undistinguishable components but rich and intricate cross-dependencies, both at second statistical order, and/or at higher-orders. The precise values of the off-diagonal entries of the pairwise correlation matrices Σ and Σ_{mf} are listed in the plots of Fig. 2.

A 4-variate sample of such MMRW textures is shown in Fig. 1. For that precise MMRW model, the functions $C^{(ss)}_{m_1,m_2}(2^j)$ and $C^{(mf)}_{m_1,m_2}(2^j)$, estimated from 100 independent copies of true MMRW, are shown in Fig. 2. They are used as ground truth for quality synthesis quality assessment, and are thus referred to as: $C^{(ss,true)}_{m_1,m_2}$ and $C^{(mf,true)}_{m_1,m_2}$.

4.2 A posteriori DCGAN synthesis quality assessment

4.2.1 A posteriori wavelet based synthesis quality indices

To asses the quality of the DCGAN synthesis, the following procedure is applied for each independent trial and each architecture.

First, 10 independent samples of 4-variate textures are synthetised from each trained DCGAN using independent white Gaussian noise samples as inputs for the Generators.

Second, the wavelet and wavelet leader analyses described in Section 2 are applied to each 4-variate texture. The functions $C^{(ss,DCGAN)}_{m_1,m_2}(2^j)$ and $C^{(mf,DCGAN)}_{m_1,m_2}(2^j)$ are computed and averaged across the 10 DCGAN-generated samples.

Third, because the richness of the targeted MMRW textures lies in their cross-statistics, two quantitative quality indices, $Q^{(ss)}$ and $Q^{(mf)}$, are constructed as sums of differences of respectively $C^{(ss)}_{m_1,m_2}(2^j)$ and $C^{(mf)}_{m_1,m_2}(2^j)$ computed from ground truth and DCGAN generated textures, across all available scales 2^j and across all (6) pairs of components:

$$Q^{(ss)} = \sum_{j,m_1,m_2 \neq m_1} (C^{(ss,true)}_{m_1,m_2}(j) - C^{(ss,DGGAN)}_{m_1,m_2}(j))^2, \tag{11}$$

$$Q^{(mf)} = \sum_{j,m_1,m_2 \neq m_1} (C^{(mf,true)}_{m_1,m_2}(j) - C^{(mf,DGGAN)}_{m_1,m_2}(j))^2. \tag{12}$$

These synthesis quality indices are referred to as a posteriori as they can be computed only after training is completed, and DCGAN textures generated.

Figure 5 compares the a posteriori synthesis quality indices $Q^{(ss)}$ and $Q^{(mf)}$, one against the other, for the 17 independent trials, and for each of the three different architectures of increasing complexities, respectively with 3, 5 or 7 convolution layers. Perfect texture synthesis would correspond to $Q^{(ss)} = Q^{(mf)} = 0$. Therefore, the closer the trial from the bottom left corner, the better the quality of the synthesis. Table 1 complements Fig. 5, reporting mean, median, minimum, maximum, standard deviation and maximum, absolute deviation across the independent trials, for each of the three architectures. Further, Table 2 reports the correlation between $Q^{(ss)}$ and $Q^{(mf)}$, for each of the three architectures.

Together, Fig. 5 and Tables 1 and 2 report the key findings of the work, as described below.

4.2.2 Correlation across synthesis quality indices

First, Fig. 5 shows that $Q^{(mf)}$ is systematically (all trials and all architectures) larger than $Q^{(ss)}$. This does not come as a major surprise as it appears as natural that higher-order cross-statistics are less easy to reproduce than second-order cross statistics.

Second, Table 2 shows that $Q^{(ss)}$ and $Q^{(mf)}$ are significantly correlated, for the three architectures. This is an informative outcome indicating that DCGAN synthesized textures reproduce cross-coherences (2nd order statistics) and cross-multifractalities (higher-order statistics) between pairs of components in correlated manner. There is thus no trial where the 2nd order cross-statistics would be well synthesized while the higher-order cross-statistics would be poorly reproduced, nor conversely.

Table 1 A posteriori synthesis quality indices $Q^{(ss)}$ and $Q^{(mf)}$: mean, median, minimum, maximum, standard deviation and maximum, absolute deviation across the 17 independent trials, as functions of architecture complexity.

	$Q^{(ss)}$					
	mean	median	min	max	std	mad
3-layer	0.09	0.09	0.03	0.18	0.04	0.03
5-layer	0.41	0.16	0.04	1.23	0.41	0.36
7-layer	0.36	0.15	0.10	1.38	0.39	0.29
	$Q^{(mf)}$					
	mean	median	min	max	std	mad
3-layer	0.29	0.26	0.20	0.73	0.12	0.07
5-layer	1.39	0.61	0.29	3.75	1.28	1.09
7-layer	1.16	0.63	0.30	4.39	1.43	1.05

4.2.3 Variability across independent trials

Third, Fig. 5 and Table 1 both show that there is a significantly large variability across trials in the quality of the DCGAN synthesized textures. In other words, the quality of the DCGAN multivariate texture synthesis may vary greatly from one trial to another for the same architecture, the same training set, the same training procedure. It has been carefully controlled that this large observed variability can not be caused by software and hardware issues and variations: The same softwares, same operating systems and even same machines were used for all trials and all architectures. No other user was allowed on the machines for the entire duration of the study. The observed variability across trials thus constitutes an intrinsic and characteristic feature of DCGAN texture synthesis.

To illustrate the extent to which the variability across trials can be large and can impact the synthesis quality, we have complemented Fig. 2, reporting, for each of the four components, the functions $\log_2 S_{m,m}(2^j)$, $C_1^{(m)}(2^j)$ and $C_2^{(m,m)}(2^j)$ as functions of $j = \log_2 2^j$, and, for each of the 6 pairs of components, the wavelet coherence functions $C^{(ss)}_{m_1,m_2}(2^j)$ and the multifractal coherence function $C^{(mf)}_{m_1,m_2}(2^j)$ as functions of $j = \log_2 2^j$ obtained as averages on true MMRW (blue), by systematically superimposing the same functions obtained from averages across DCGAN synthesized textures (red). These superimpositions are reported in Figs. 6, 7 and 8 for architectures with 7, 5 and 3 convolutional layers respectively. For each architecture, the three different trials achieving best, median and worst synthesis performance (computed as $Q^{(ss)} + Q^{(mf)}$) are reported. In addition, Figs. 9, 10 and 11 for architectures with 7, 5 and 3 convolutional layers respectively, show three different samples of 4-variate textures generated from these three different trials corresponding to best, median and worst performance.

Table 2 Correlation between the a posteriori synthesis quality indices $Q^{(ss)}$ and $Q^{(mf)}$ (across the 17 independent trials) as functions of architecture complexity.

.	3-layer	5-layer	7-layer
corr($Q^{(ss)}$, $Q^{(mf)}$)	0.64	0.93	0.94

Let us first further discuss in detail the results obtained for the most complex (7-layer) architecture. Fig. 9 (top row) shows that the *best case* (that is the trial with best a posteriori synthesis quality indices) yields samples of 4-variate texture that visually very much ressemble true MMRW textures. Further, Figs. 6 (top row) shows that the component-wise multiscale statistics of true (blue) and DCGAN (red) MMRW very well superimpose (left plots). For each component of the textures, the selfsimilarity (univariate second-order statistics) and the multifractality (univariate higher-order statistics) are extremely well reproduced by DCGAN synthesis. Moreover, Fig. 6 (top row) also indicates that cross-selfsimilarity (multivariate second-order statistics, middle plots) and cross-multifractality (multivariate higher-order statistics, right plots) for all pairs of components are well reproduced by DCGAN synthesis for that trial. However, Fig. 9 (bottom

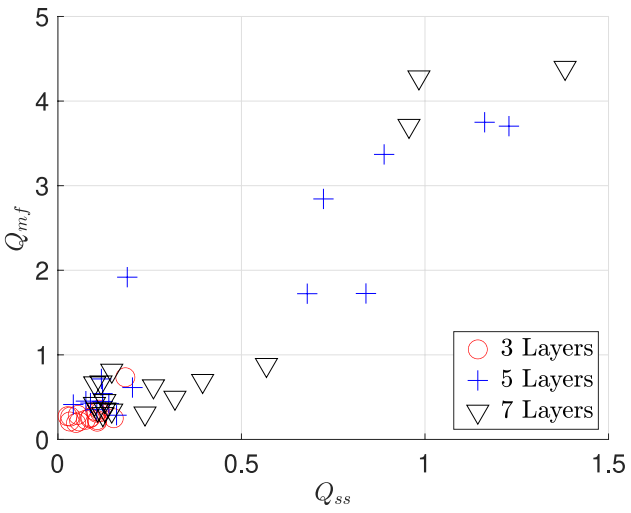

Figure. 5 A posteriori synthesis quality indices: scatter of the 17 independent trials, for each of the three different architectures of increasing complexities, respectively with 3, 5 or 7 convolution layers.

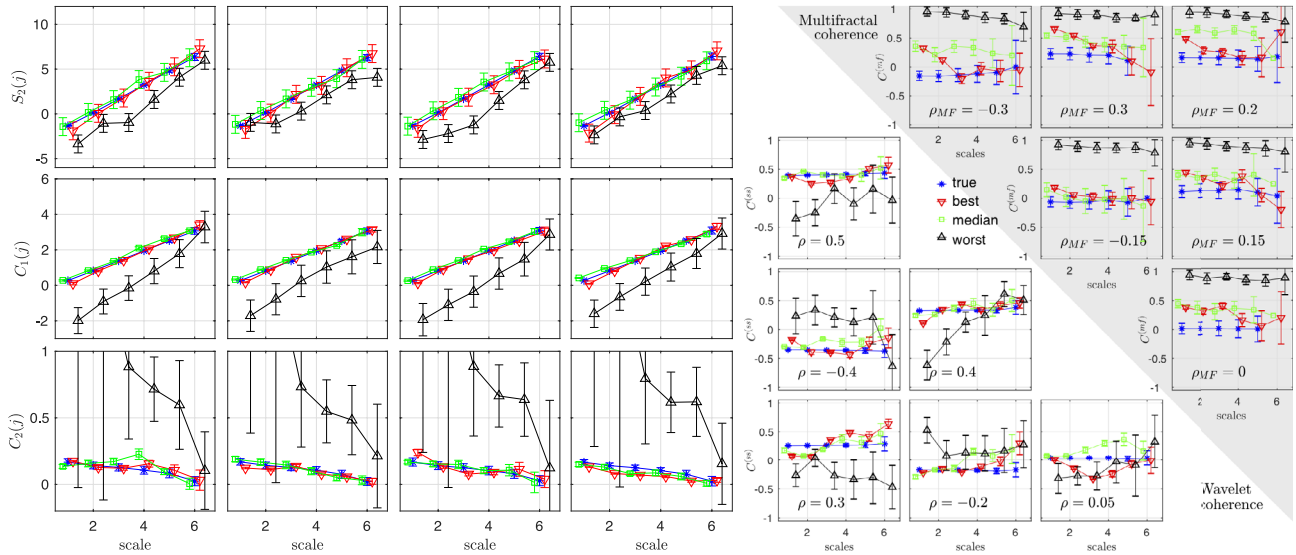

Figure. 6 DG-GAN (7-layer) synthesized multivariate texture multiscale analysis. Wavelet and wavelet leader analysis averaged over 10 DCGAN generated 4-variate samples of MMRW taken from the run with best (red-down-triangle), median (green-square) and worst (black-up-triangle) synthesis performance, compared to those obtained from true MMRW (blue-star). Legend as that in Fig. 2.

row) also shows that the *worst case* (that is the trial with poorest a posteriori synthesis quality indices) yields samples of 4-variate texture that visually do not ressemble at all true MMRW textures. Further, Fig. 6 (bottom row) shows that while component-wise multiscale statistics of true (blue) and DCGAN (red) MMRW superimpose approximatively for selfsimilarity (univariate second-order statistics), they do not, at all, superimpose for multifractality (univariate higher-order statistics, left plots): Univariate (component-wise) multifractality is thus not all reproduced by DCGAN synthesis. Moreover, Fig. 6 (bottom row) also indicates that cross-selfsimilarity (multivariate second-order statistics, middle plots) and cross-multifractality (multivariate higher-order statistics, right plots) for all pairs of components are not at all reproduced for that trial.

The very same conclusions hold for the intermediate complexity (5-layer) architecture (cf. Figs. 10 and 7). While

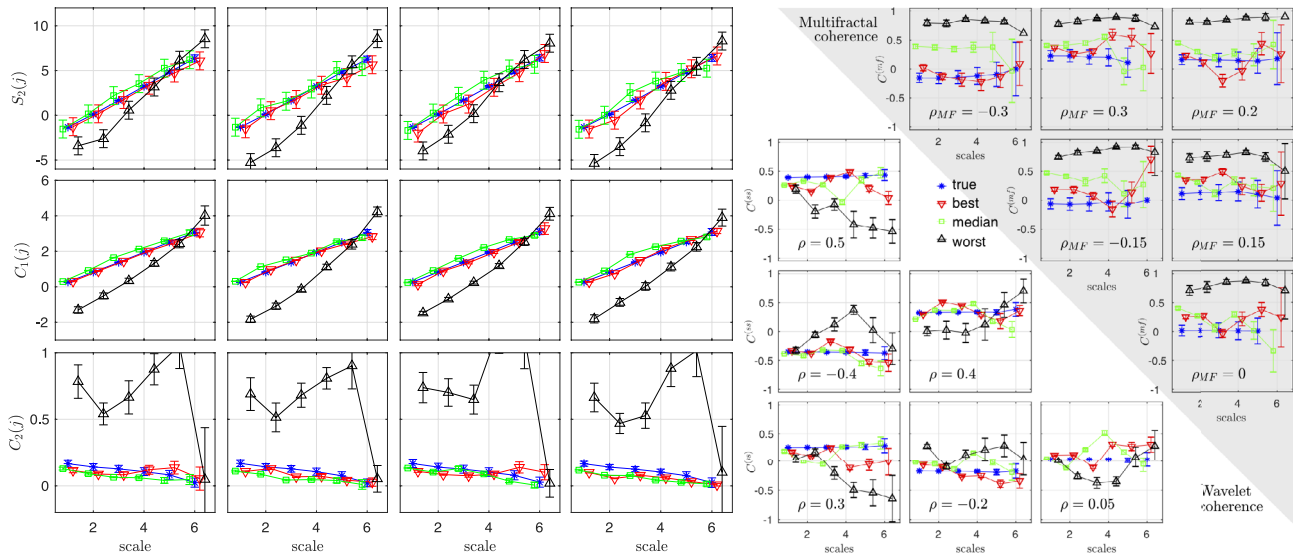

Figure. 7 DG-GAN (5-layer) synthesized multivariate texture multiscale analysis. Wavelet and wavelet leader analysis averaged over 10 DCGAN generated 4-variate samples of MMRW taken from the run with best (red-down-triangle), median (green-square) and worst (black-up-triangle) synthesis performance, compared to those obtained from true MMRW (blue-star). Legend as that in Fig. 2.

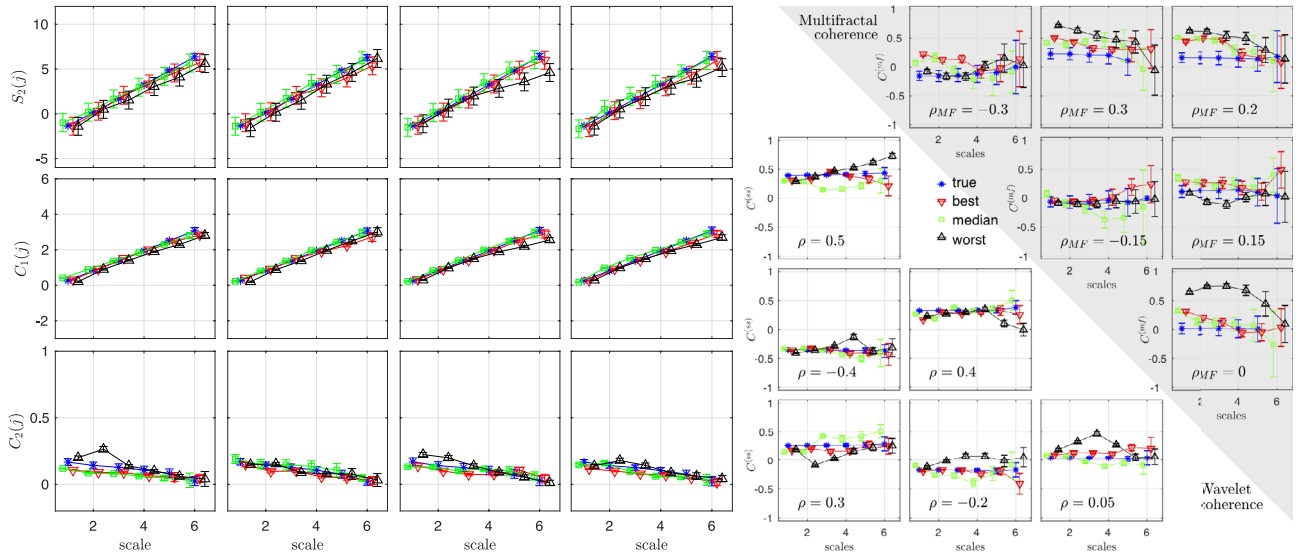

Figure. 8 DG-GAN (3-layer) synthesized multivariate texture multiscale analysis. Wavelet and wavelet leader analysis averaged over 10 DCGAN generated 4-variate samples of MMRW taken from the run with best (red-down-triangle), median (green-square) and worst (black-up-triangle) synthesis performance, compared to those obtained from true MMRW (blue-star). Legend as that in Fig. 2.

for the trial with best a posteriori synthesis quality indices (top row) the DCGAN texture satisfactorily well ressemble true MMRW and (cross-)selfsimilarity and (cross-) multifractality are well reproduced by DCGAN textures, the trial with worst synthesis performance (bottom row) does not yield texture that visually ressemble true MMRW, and multifractality, cross-selfimilarity and cross-multifractality are totally missed by DCGAN textures. The 5-layer architecture thus essentially shows the same variability in terms of performance across trials as the 7-layer architecture. In [39], we have ruled out the possibility that such variability may stem from not enough iterations and convergence issues or too small training dataset size. This is thus not further discussed here.

The less complex (3-layer) architecture also shows some variability across trials though significantly reduced

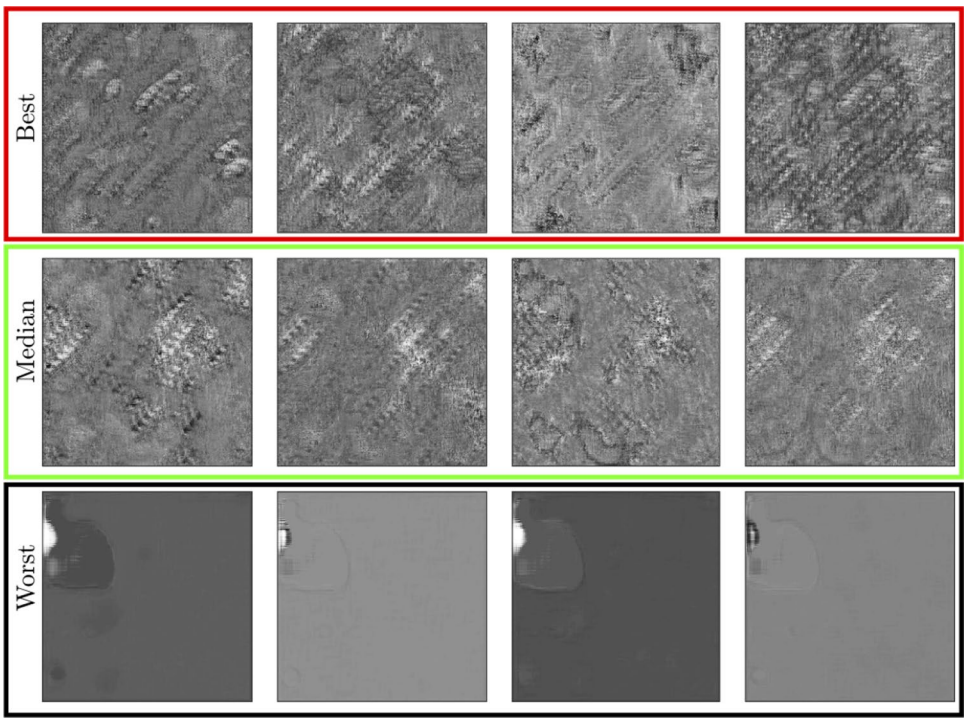

Figure. 9 DG-GAN (7-layer) synthesized multivariate multifractal textures. DCGAN generated 4-variate sample of MMRW taken from the run with best (top-red), median (middle-green) and worst (bottom-black) synthesis performance.

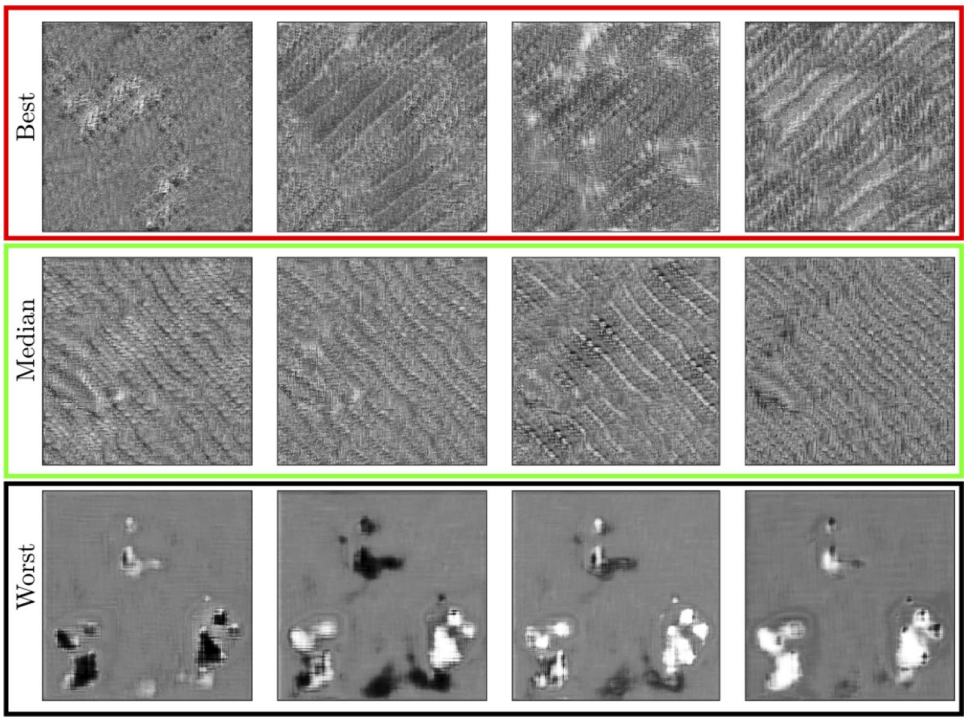

Figure. 10 DG-GAN (5-layer) synthesized multivariate multifractal textures. DCGAN generated 4-variate sample of MMRW taken from the run with best (top-red), median (middle-green) and worst (bottom-black) synthesis performance.

compared to those induced by the 5-layer and 7-layer architectures (cf. Table 1). Fig. 11 shows that samples of DCGAN 4-variate textures from best and worst trials very look alike and ressemble true MMRW samples. Fig. 8 show that (cross-)selfsimilarity and (cross-)multifractality are very well reproduced for the best trial and still very satisfactorily for the worst trial. In other words, all trials are equally satisfactory in terms of synthesis quality for the 3-layer architecture.

4.2.4 Performance vs. architecture complexity

Fourth and finally, Fig. 5 and Table 1 clearly indicate that the average synthesis quality performance for the two largest

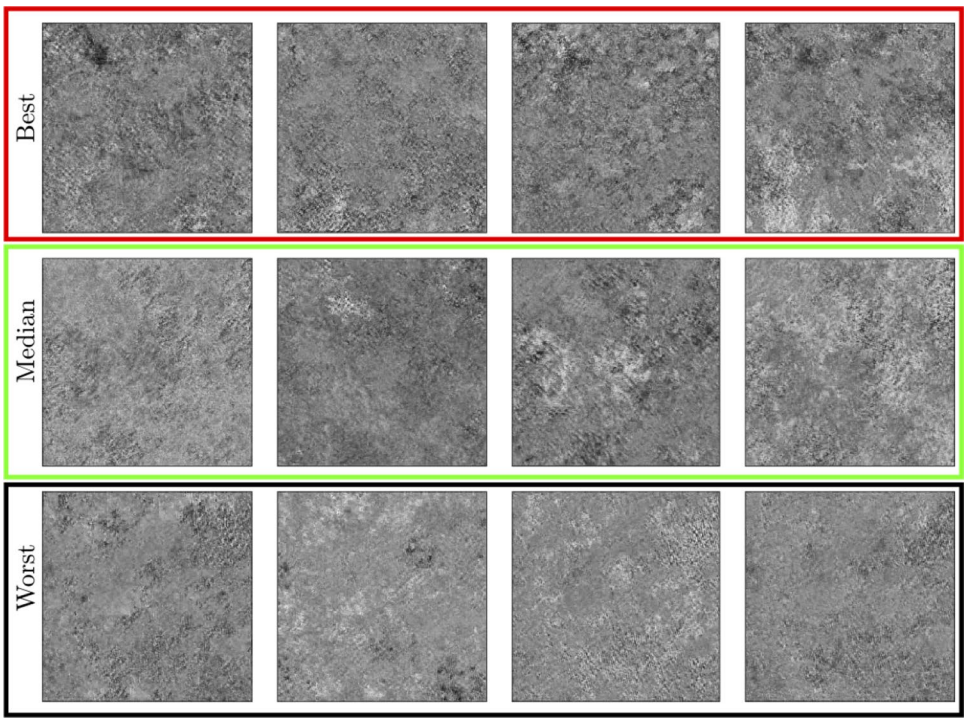

Figure. 11 DG-GAN (3-layer) synthesized multivariate multifractal textures. DCGAN generated 4-variate sample of MMRW taken from the run with best (top-red), median (middle-green) and worst (bottom-black) synthesis performance

complexity (5-layer and 7-layer) architectures are equally poor and with a large variability around the mean, while the synthesis quality performance achieved with the less complex (3-layer) architecture are far better with much less variability across trials.

This constitutes the most significant outcome of the present work. Synthesizing 4-variate Multifractal textures with intricate cross-statistics at second and higher order statistics can intuitively and a priori be thought as a complex task which lead to use a 5-convolution layer architecture for the generator in DCGAN-based synthesis [39]. The large variability across trials observed in [39] lead us to further increase architecture complexity in the present work to 7 convolution layers: Instead of yielding the expected reduction in variability, this lead to an even larger variability across trials.

To the converse, when the architecture complexity is reduced, the variability across trials disappears and all trials are of equivalent quality. This homogeneity in the trials for the 3-layer simplest architecture yields on average much better synthesis performance compared against the 5-layer and 7-layer more complex architectures. It is further interesting to note that the best trials obtained with the most complex architectures still have slightly poorer performance than typical performance obtained with the 3-layer simplest architecture.

4.2.5 DCGAN multivariate texture synthesis quality

In sum, DCGAN performance for multivariate texture synthesis can range from excellent to extremely poor across independent trials, despite identical settings (same architectures, training dataset, initialization procedures,...), when neural networks, with architectures too complex (here too many layers) for the task, are used. For architectures whose complexity matches the complexity of the task, DCGAN perform impressively well in multivariate multifractal texture synthesis. Fig. 11 displays 3 samples DGGAN generated 4-variate multifractal textures, that satisfactorily ressemble true MMRW shown in Fig. 1. Further, Fig. 8 shows that component-wise selfsimilarity and multifractality are perfectly reproduced and across pairs of components, cross-selfsimilarity and cross-multifractality are also well reproduced, though with residual variability across trials. Notably, positive and/or negative cross-selfsimilarities and cross-multifractalities are well reproduced, including for pairs of components where cross-selfsimilarity and cross-multifractality have opposite signs. This is per se an impressive performance of DCGAN as it implies capturing/learning a intricate feature of the textures: Pairs of components may be negatively correlated yet with positively correlated bustiness or intermittency.

4.3 A priori synthesis quality assessment

4.3.1 A priori loss function based synthesis quality index

Indices $Q^{(ss)}$ and $Q^{(mf)}$ proved relevant to serve as objective and automated texture synthesis quality and reproducibility indices. However, their computation requires that training is completed, that DCGAN textures are generated and analyzed: They are thus a posteriori indices. Further, and foremost, their computation requires that a ground truth is available for comparisons: Significant information regarding (the multiscale statistics of) targeted textures is hence needed, which somehow contradicts the use of deep learning to synthesize textures, as it is expected to discover relevant statistics by itself.

To overcome this issue, we propose the use of an a priori index, computed from the only quantities available during training and not requiring any ground truth: the loss functions of the generator and of the discriminator. This a priori index, hereafter referred to as DiffLoss, is defined as the absolute value of the difference between the Generator and Discriminator loss functions, averaged across the last 500 iterations in training.

4.3.2 Variability vs. architecture complexity

Table 3 reports the statistics of the DiffLoss synthesis quality index across trials for the three architectures. It shows that DiffLoss reproduces the same pattern as the a posteriori indices $Q^{(ss)}$ and $Q^{(mf)}$: For the two architectures with largest complexities (5-layer and 7-layer), DiffLoss show a very large variability across trials. To the converse,

Table 3 A priori synthesis quality index DiffLoss: mean, median, minimum, maximum, standard deviation and maximum, absolute deviation across the 17 independent trials, as functions of architecture complexity.

	DiffLoss					
	mean	median	min	max	std	mad
3-layer	3.14	2.92	2.03	4.39	0.76	0.62
5-layer	4.70	3.58	2.56	7.60	1.88	1.73
7-layer	3.52	3.32	1.36	6.70	1.60	1.20

Table 4 Correlation between the a priori index DiffLoss and each of the a posteriori synthesis quality indices $Q^{(ss)}$ and $Q^{(mf)}$ (across the 17 independent trials) as functions of architecture complexity.

	3-layer	5-layer	7-layer
corr($Q^{(ss)}$,DiffLoss)	0.67	0.72	0.81
corr($Q^{(mf)}$,DiffLoss)	0.33	0.79	0.70

for the architecture with the lowest complexity (3-layer), the variability is significantly reduced, suggesting that all trials are equivalent. Additionally, the average and typical values of DiffLoss are much smaller for the 3-layer architecture, compared to the 5-layer and 7-layer architectures.

4.3.3 A priori vs. a posteriori synthesis quality indices

Furthermore, and more importantly, Fig. 12 and Table 4 illustrate and quantify that the a priori index DiffLoss correlates significantly with each of the a posteriori indices $Q^{(ss)}$ and $Q^{(mf)}$. This constitutes the second significative outcome of the present work: The a priori loss functions-based index can thus actually be used to predict the quality of the DCGAN textures, while the training is being completed, without having to synthesize and analyze DCGAN textures nor to use any a priori known ground truth. This permits for instance to monitor while the training is being processed the index DiffLoss as a function of the number of iterations and to know whether the current trial is converging and thus will lead to meaningful synthesis, or not.

5 Discussion, conclusions and future work

The overarching goal of the paper was to devise a methodology permitting to quantify how well DCGAN performs in multivariate texture synthesis. The proposed solution relies on several key steps.

First, multiscale (wavelet and wavelet leader-based) statistics component-wise and cross-statistics are used to characterize the properties of the textures. Though they are *naturally* matched to the description of multifractal textures, it is worth mentioning that such multiscale representations provide relevant (cross-)statistics to characterize a very large variety of textures, irrespective of their having scale-free dynamics or not [2, 44, 55]. Synthesis quality indices can then be easily designed by comparing these multiscale statistics obtained as average across DCGAN generated textures against those computed as averages from the textures within the training set. Though multiscale statistics are naturally-suited to the assessment of fractal/multifractal textures, they also can be used as generic statistics representative of the properties of many textures, even non fractal ones, and this can be used as generic assessment tools.

Second, synthesis quality assessment is repeated across several independent trials, where the training is restarted from scratch, while keeping all other conditions identical (same training set, same training procedure, same architecture, same software, same hardware). Instead of reporting the performance of the best trial, we have assessed the variability across trials and hence the reproducibility of the procedure.

These methodological investigations permitted to draw several significant conclusions related to deep learning practice.

First, classically in deep learning, practitioners face the issue of choosing an appropriate architecture. The present investigations showed that when the chosen architecture is too complex, DCGAN suffer from a very large variability across independent training in synthesis performance, with trials yielding generated textures that extremely satisfactorily reproduce the statistics of the targeted textures, trials which totally fail to do so, and with a continuum of performance across trials from excellent to extremely poor. This variability reduces significantly when the complexity of the chosen architecture is decreased, yielding trials which are essentially statistically equivalent one to the other. This shows that exploring deep learning with very complex architectures is not always the best path to follow. This obviously raises the question of tuning the

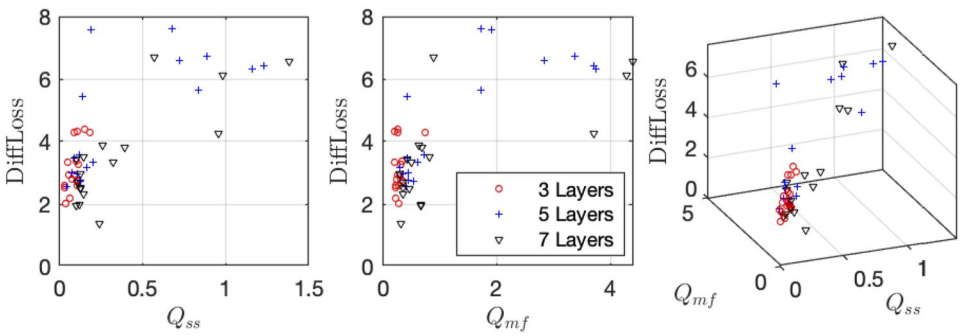

Figure. 12 A priori versus a posteriori synthesis quality indices. Left, $Q^{(ss)}$ vs. DiffLoss. Middle: $Q^{(mf)}$ vs. DiffLoss. Right, 3D plot of the three indices.

complexity of the architecture to the complexity of the task to achieve, an obviously difficult question. In texture synthesis, a potential direction to explore is to assess the complexity of the task by computing information theoretic quantities (entropies) on the statistics of the textures in the training set. The complexity of the neural network architectures is also complicated to assess. While obviously the number of trainable parameters, or the number of layers, are relevant descriptors of neural network complexities, other strategies aiming to quantify the approximation ability of non linear function using the Vapnik-Chervonenkis Dimension [9, 49] were used for classification task [11, 16, 35] and could be worth exploring here.

Second, for DCGAN architectures well-tuned to the complexity of the multivariate multifractal textures, synthesis performance are outstanding. DCGAN generate textures that reproduce extremely well not only the second-order and higher-order statistics of each component independently, but also the cross-statistics at second and higher orders between pairs of components. This is all the more impressive as all components were chosen to have identical statistics (so as to be undistinguishable) while all pairs of components had different cross-statistics, with possibly opposite signs at second-order and higher orders (e.g., negative cross-selfsimilarity but positive cross multifractality), corresponding thus to very fine details in cross-spatial dynamics that were discovered by the neural network. It is however worth mentioning that despite excellent reproduction of the statistics of the targeted textures, the eye is still able to see subtle differences in the visual aspects of DGCAN generated textures, mostly corresponding to anistropic-like perceptual features, while targeted textures are perfectly isotropic. This will be further explored.

Third, the synthesis quality assessment methodology proposed is costly and a posteriori as it requires that the neural network is trained, that textures are generated and analyzed using multiscale representations, and that these analyses are compared to the same analyses performed on the training set. We have shown here though that a metric based on combining the loss functions of the Generator and Discriminator along the training process, permit to devise a low-cost and a priori synthesis quality index, which correlates strongly withe a posteriori indices. This opens the road for rapidly testing if a current training will yield satisfactory synthesis performance or to tune empirically but efficiently architecture complexity to task complexity.

References

1. Abry, P., Didier, G., et al. (2018). Wavelet estimation for operator fractional brownian motion. *Bernoulli, 24*(2), 895–928.
2. Abry, P., Jaffard, S., & Wendt, H. (2015). Irregularities and scaling in signal and image processing: multifractal analysis. In Benoit Mandelbrot: a life in many dimensions. World Scientific, pp. 31–116.
3. Abry, P., Roux, S. G., Wendt, H., Messier, P., Klein, A. G., Tremblay, N., et al. (2015). Multiscale anisotropic texture analysis and classification of photographic prints: Art scholarship meets image processing algorithms. *IEEE Signal Processing Magazine, 32*(4), 18–27.
4. Abry, P., Wendt, H., & Jaffard, S. (2013). When van gogh meets mandelbrot: Multifractal classification of painting's texture. *Signal Processing, 93*(3), 554–572.
5. Abry, P., Wendt, H., Jaffard, S., & Didier, G. (2019). Multivariate scale-free temporal dynamics: From spectral (fourier) to fractal (wavelet) analysis. *Comptes Rendus Physique, 20*(5), 489–501.
6. Angles, T., & Mallat, S. (2018). *Generative networks as inverse problems with scattering transforms*. ICLR: In Proc.
7. Arjovsky, M., Chintala, S., & Bottou, L. (2017). Wasserstein gan. arXiv preprint arXiv:1701.07875.
8. Bacry, E., Delour, J., & Muzy, J.-F. (2001). Multifractal random walk. *Phys. Rev. E, 64,*.
9. Bartlett, P. L., & Maass, W. (2003). Vapnik-chervonenkis dimension of neural nets. *The handbook of brain theory and neural networks* 1188–1192.
10. Basu, S., Karki, M., & Mukhopadhyay, S., et al. (2016). A theoretical analysis of deep neural networks for texture classification. *In 2016 International Joint Conference on Neural Networks (IJCNN), IEEE*, pp. 992–999.
11. Baum, E. B., & Haussler, D. (1989). What size net gives valid generalization? In Advances in neural information processing systems, pp. 81–90.
12. Bergmann, U., Jetchev, N., & Vollgraf, R. (2017). Learning texture manifolds with the periodic spatial gan. *In Proc. of the 34th International Conference on Machine Learning-Volume 70*, 469–477.
13. Brock, A., Donahue, J., & Simonyan, K. (2018). Large scale gan training for high fidelity natural image synthesis. arXiv preprint arXiv:1809.11096.
14. Depeursinge, A., Foncubierta-Rodriguez, A., Van De Ville, D., & Müller, H. (2014). Three-dimensional solid texture analysis in biomedical imaging: review and opportunities. *Medical image analysis, 18*(1), 176–196.
15. Dobrescu, R., Dobrescu, M., Mocanu, S., & Popescu, D. (2010). Medical images classification for skin cancer diagnosis based on combined texture and fractal analysis. *WISEAS Transactions on Biology and Biomedicine, 7*(3), 223–232.
16. Friedland, G., & Krell, M. (2017). A capacity scaling law for artificial neural networks. arXiv preprint arXiv:1708.06019.
17. Gatys, L., Ecker, A. S., & Bethge, M. (2015). Texture synthesis using convolutional neural networks. In Advances in neural information processing systems pp. 262–270.
18. Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the thirteenth international conference on artificial intelligence and statistics pp. 249–256.
19. Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT press, 2016.
20. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. In Advances in neural information processing systems pp. 2672–2680.
21. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition pp. 770–778.
22. Helgason, H., Pipiras, V., & Abry, P. (2011). Fast and exact synthesis of stationary multivariate Gaussian time series using circulant embedding. *Signal Processing, 91*(5), 1123–1133.

23. Helgason, H., Pipiras, V., & Abry, P. (2014). Smoothing windows for the synthesis of gaussian stationary random fields using circulant matrix embedding. *Journal of Computational and Graphical Statistics, 23*(3), 616–635.
24. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., & Hochreiter, S. (2017). GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. *In Advances in Neural Information Processing Systems, 30,* 6626–6637.
25. Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167.
26. Jaffard, S., Abry, P., & Wendt, H. (2015). *Irregularities and Scaling in Signal and Image Processing: Multifractal Analysis* (pp. 31–116). Singapore: World scientific publishing.
27. Jaffard, S., Seuret, S., Wendt, H., Leonarduzzi, R., & Abry, P. (2019). Multifractal formalisms for multivariate analysis. *Proceedings of the Royal Society A, 475*(2229), 20190150.
28. Jaffard, S., Seuret, S., Wendt, H., Leonarduzzi, R., Roux, S., & Abry, P. (2019). Multivariate multifractal analysis. *Applied and Computational Harmonic Analysis, 46*(3), 653–663.
29. Jetchev, N., Bergmann, U., & Vollgraf, R. (2016). Texture synthesis with spatial generative adversarial networks. arXiv preprint arXiv:1611.08207.
30. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T. (2019). Analyzing and improving the image quality of stylegan. arXiv preprint arXiv:1912.04958.
31. Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems pp. 1097–1105.
32. Lennon, F. E., Cianci, G. C., Cipriani, N. A., Hensing, T. A., Zhang, H. J., Chen, C.-T., et al. (2015). Lung cancer?a fractal viewpoint. *Nature reviews Clinical oncology, 12*(11), 664.
33. Leonarduzzi, R., Abry, P., Roux, S. G., Wendt, H., Jaffard, S., & Seuret, S. (2018). Multifractal characterization for bivariate data. In Proc. European Signal Processing Conference (EUSIPCO) (Rome, Italy, September 2018).
34. Leonarduzzi, R., Wendt, H., Abry, P., Jaffard, S., Melot, C., Roux, S. G., & Torres, M. E. (2016). p-exponent and p-leaders, part ii: Multifractal analysis. relations to detrended fluctuation analysis. *Physica A: Statistical Mechanics and its Applications, 448*, 319–339.
35. Liotet, P., Abry, P., Leonarduzzi, R., Senneret, M., Jaffrès, L., & Perrin, G. (2020). Deep learning abilities to classify intricate variations in temporal dynamics of multivariate time series. In ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) IEEE, pp. 3857–3861.
36. Malkov, S., Shepherd, J. A., Scott, C. G., Tamimi, R. M., Ma, L., Bertrand, K. A., et al. (2016). Mammographic texture and risk of breast cancer by tumor type and estrogen receptor status. *Breast Cancer Research, 18*(1), 122.
37. Mallat, S. (1998). *A Wavelet Tour of Signal Processing*. San Diego, CA: Academic Press.
38. Mandelbrot, B. B. (1974). Intermittent turbulence in self-similar cascades: divergence of high moments and dimension of the carrier. *J. Fluid Mech., 62,* 331–358.
39. Mauduit, V., Abry, P., Leonarduzzi, R., Roux, S., & Quemener, E. (2020). Dcgan for the synthesis of multivariate multifractal textures: How do we know it works? In 2020 IEEE 30th International Workshop on Machine Learning for Signal Processing (MLSP) IEEE, pp. 1–6.
40. Myint, S. (2003). Fractal approaches in texture analysis and classification of remotely sensed data: Comparisons with spatial autocorrelation techniques and simple descriptive statistics. *International Journal of remote sensing, 24*(9), 1925–1947.
41. Quemener, E., & Corvellec, M. (2014). Sidus, the solution for extreme deduplication of an operating system. *The Linux Journal*.
42. Radford, A., Metz, L., & Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434.
43. Roux, S. G., Clausel, M., Vedel, B., Jaffard, S., & Abry, P. (2013). Self-Similar Anisotropic Texture Analysis: The Hyperbolic Wavelet Transform Contribution. *IEEE Trans. Image Proc., 22*(11), 4353–4363.
44. Roux, S. G., Clausel, M., Vedel, B., Jaffard, S., & Abry, P. (2013). Self-similar anisotropic texture analysis: The hyperbolic wavelet transform contribution. *IEEE Transactions on Image Processing, 22*(11), 4353–4363.
45. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., & Chen, X. (2016). Improved techniques for training gans. In Advances in neural information processing systems pp. 2234–2242.
46. Szegedy, C., Liu, W., Jia, Y., et al. (2015). Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition pp. 1–9.
47. Ulyanov, D., Lebedev, V., Vedaldi, A., & Lempitsky, V. S. (2016). Texture networks: Feed-forward synthesis of textures and stylized images. *In ICML, 1,* 4.
48. Ulyanov, D., Lebedev, V., & Vedaldi, A. (2016). *Lempitsky* (Vol. S, p. 4). Texture networks: Feed-forward synthesis of textures and stylized images. In ICML.
49. Vapnik, V. N., & Chervonenkis, A. Y. (2015). On the uniform convergence of relative frequencies of events to their probabilities. In Measures of complexity. Springer, pp. 11–30.
50. Wendt, H., Abry, P., & Jaffard, S. (2007). Bootstrap for empirical multifractal analysis. *IEEE Signal Proc. Mag., 24*(4), 38–48.
51. Wendt, H., Hourani, M., Barasab, A., & Kouame, D. (2020). Deconvolution for improved multifractal characterization of tissues in ultrasound imaging. In IEEE International Ultrasonic Symposium (IUS) (Las Vegas, USA).
52. Wendt, H., Leonarduzzi, R., Abry, P., Roux, S., Jaffard, S., & Seuret, S. (2018). Assessing cross-dependencies using bivariate multifractal analysis. In IEEE Int. Conf. Acoust., Speech, and Signal Proces. (ICASSP).
53. Wendt, H., Leonarduzzi, R., Abry, P., Roux, S., Jaffard, S., & Seuret, S. (2018). Assessing cross-dependencies using bivariate multifractal analysis. In IEEE Int. Conf. Acoust., Speech, and Signal Proces.
54. Wendt, H., Roux, S. G., Abry, P., & Jaffard, S. (2009). Wavelet leaders and bootstrap for multifractal analysis of images. *Signal Proces., 89*(6), 1100–1114.
55. Wendt, H., Roux, S. G., Jaffard, S., & Abry, P. (2009). Wavelet leaders and bootstrap for multifractal analysis of images. *Signal Processing, 89*(6), 1100–1114.
56. Zhou, Y., Zhu, Z., Bai, X., Lischinski, D., Cohen-Or, D., & Huang, H. (2018). Non-stationary texture synthesis by adversarial expansion. arXiv preprint arXiv:1805.04487.

Patrice Abry Patrice Abry was born in Bourg-en-Bresse, France in 1966. He received the degree of Professeur-Agrege de Sciences Physiques, in 1989 at Ecole Normale Superieure de Cachan and completed a PhD in Physics and Signal Processing, at Universite Claude-Bernard University in Lyon in 1994. He is a CNRS Senior Scientist, at the Physics dept. of Ecole Normale Superieure de Lyon, where he is in charge of the {Signal, systems and Physics} research team. Patrice Abry received the AFCET-MESR-CNRS prize for best PhD in Signal Processing for the years 93-94 and has been elected IEEE Fellow in 2011. He is the author of a book in French dedicated to wavelet, scale invariance and hydrodynamic turbulence and is also the coeditor of a book entitled "Scaling, Fractals and Wavelets" (ISTE). He has been elected IEEE fellow in 2011 and serve for the IEEE SPS Signal Processing Theory and Methods Committee.In 2020, he received the Michel Monpetit - INRIA prize, awarded by the French Academy of Sciences. His current research interests include wavelet-based analysis and modeling of statistical scale-free dynamics (self-similarity, stable processes, multi-fractal, 1/f processes, long-range dependence, local regularity of processes, infinitely divisible cascades, departures from exact scale invariance). Beyond theoretical developments and contributions in multifractal analysis and stochastic process design, Patrice Abry shows a strong interest into real-world applications, such as hydrodynamic turbulence, computer network teletraffic, Heart Rate Variability, neurosciences and art investigations, and more recently pandemic monitoring.

Vincent Mauduit Vincent Mauduit is student at the Ecole Centrale de Nantes, a French engineering school. After a specialisation in applied mathematics, He joined Dr Patrice Abry's team for studying multivariate textures using Deep Learning approaches. He then decided to continue his studies with a specialisation in bioinformatics. He is currently working on kidney transplantation with an immunogenetics team at the CHU de Nantes and will start a PhD in October 2021.

Emmanuel Quemener After a PhD in optical signal processing, Emmanuel Quémener became engineer in academic research. Invested in the use of free softwares in higher education since 1996, he generalized their use in 2001 throughout the IT infrastructure of ENS Cachan, from networks to critical services. Returning to scientific computing mid 2005, he worked successively at the Lyon observatory (JWST space project) then at the ENS de Lyon Computer Laboratory. Currently an engineer at the Blaise Pascal Center, his main projects focus on software integration, storage, scientific computing on GPUs, the use of Python as a scientific language and the development of SIDUS, a deployment solution ensuring the reproducibility of the system for the OS on a fleet of machines. His operational activity is completed by the creation and administration of technical platforms dedicated to researchers and teachers in the Lyon area. He also serves as "computer test pilot" to provide to the computing center of ENS de Lyon a full internalization for its 800 nodes.

Stephane Roux Stephane G. Roux was born in Montpellier, France in 1967. He graduated at University of Montpellier and completed a Ph.D. in Physics in 1996 at University of Aix-Marseille. Since September 2001, he is associate professor at the Laboratoire de Physique of Ecole Normale Supèrieure de Lyon. His research interests include scale invariance, wavelet analysis and their practical applications in physics, neuroscience, art and biology.