DCGAN for the synthesis of multivariate multifractal textures: How do we know it works ?

V. Mauduit, P. Abry, *Fellow, IEEE*, R. Leonarduzzi, S. G. Roux and E. Quemener,

***Abstract*—Deep Learning is nowadays widely used for several tasks in image processing. Notably, it has been massively used for image synthesis, mostly with strong geometrical contents. It has however been much less used for texture synthesis. Also, the issue of assessing the quality of the synthesized textures has not often been addressed. The present study aims to study the ability of Deep Convolutional Generative Adversarial Networks to synthesize multivariate textures characterized by rich multiscale cross-statistics (multifractals). The focus is on quantifying the quality of the synthesized textures, on assessing the reproducibility of the learning procedure and on studying the impact of loss functions and of training dataset sizes.**

***Index Terms*—Generative Adversarial Network, Multivariate Texture Synthesis, Quality Assessment, Multifractals.**

I. Introduction

Context. After impressive successes in numerous image processing tasks (classification, segmentation,…) [1], deep learning (DL) is nowadays massively used for image synthesis [2], [3]. However, most applications are related to images with dominant *cartoon* content (edges, shapes, angles, objects, …) [4], [5], [6]. Texture synthesis with DL architectures has been much less addressed, while it is often considered that *intrinsic dimensions* are larger for textured than for geometric images, hence inducing larger difficulties for DL [7]. Studying issues in DL-based texture synthesis is thus at the heart of this work.
Related Works. Despite massive use [2], [8], [9], image synthesis by inverting convolutional neural networks (CNN) was shown to be practically unstable and theoretically poorly understood (cf. a contrario [10]). Autoencoder (AE) were also used for image synthesis [11]. However, since designed to create latent variable representations, AE often fail as generators of new instances and synthesized images suffer from large artifacts or lack important content such as high-frequency information [11], [12]. The recently introduced Generative Adversarial Networks (GAN) attempt to overcome these limitations. They have been successfully used in many domains with state-of-the-art results [3], [13], [14], [15], which motivates their use here.

One unsolved issue consists of the assessment of the quality of the generated images. Often, this is simply done by human visual inspection (cf. e.g. [16]). Quantitative scores were also proposed, based on information theoretic measures [16]. While potentially valid for images with strong geometric content, they are much less relevant for textures, whose characteristics are statistical in nature. The quality assessment of DL texture synthesis thus remains an open question, of high relevance in practice to assess efficiency, robustness and reproducibilty, issues at the core of the present work.
Goals, Contributions and outlines The overall goal of the present work is to assess the quality of GAN-synthesized multivariate textures and the reproducibility of the training procedure. To that end, we make use of multivariate multifractal textures, widely used in applications and of interest per se [17], [18], but used here only as archetypal examples for realistic multivariate textures with complex spatial and cross-spatial dynamics. The originality of the present work lies in the use of such models with known and controlled statistics (described in Section II) to construct a posteriori indices (based on state-of-the-art wavelet and wavelet leader analysis) to quantify the quality of GAN-synthesized textures, and thus to assess the reproducibility of the training procedure. GAN architectures used here are described in Section III. Experiments and results, reported in Section IV, show i) the relevance of the proposed a posteriori indices to assess synthesis quality, ii) a quantified significant lack of reproducibility in training GAN architectures, iii) a quantified impact of the learning dataset size on performance, iv) the relevance of the construction of an a priori quality index based on the loss functions only.

II. Multivariate multifractal textures

Modeling. After Mandelbrot's seminal work [19], Multifractal random walks (MRW) [20] are nowadays used as versatile models for real-world textures, well-characterized with multiscale scale-free statistics [21], [22]. Elaborating on bivariate MRW 1D signals [18], [23], we propose here to define multivariate MRW textures (MMRW), as $\forall m = 1,\ldots,M$, $X_m(\underline{x}) = G_m(\underline{x})e^{\omega_m(\underline{x})}$, with $\underline{G}(\underline{x}) = \{G_1(\underline{x}),\ldots,G_M(\underline{x})\}$ and $\underline{\omega}(\underline{x}) = \{\omega_1(\underline{x}),\ldots,\omega_M(\underline{x})\}$ two independent zero-mean multivariate 2D-Gaussian processes. $\underline{G}(\underline{x})$ is defined as the multivariate extension of the univariate 2D fractional Gaussian noise (2D-fGn) [24], [25]. $\underline{G}(\underline{x})$ is fully defined by its $M \times M$ covariance functions, controlled by M Hurst exponents $\underline{H} = (H_1,\ldots,H_M)$ and a $M \times M$ pointwise covariance matrix Σ [26], [27]. $\underline{\omega}(\underline{x})$ is defined via $M \times M$ covariance functions, designed to induce multifractality in spatial statistics as: $\{\Sigma_{mf}\}(m_1,m_2) = \rho_{mf}(m_1,m_2)\lambda_{m_1}\lambda_{m_2}\log\left(\frac{L}{||\underline{x}||+1}\right)$, (for $||\underline{x}|| \leq L$ and 0 otherwise, with L an arbitrary integral scale) thus fully controlled by a vector of multifractality

Submitted 2020/01/xx. V. Mauduit, P. Abry and S. Roux are with the Univ Lyon, ENS de Lyon, Univ Claude Bernard, CNRS, Laboratoire de Physique, Lyon, France, E. Quemener is with Univ Lyon, ENS de Lyon, Univ Claude Bernard, Laboratoire d'Informatique et du Parallélisme, Lyon, France, R. Leonarduzzi is with ENS, PSL, Paris, France. Work supported by CBP (Blaise Pascal Center) with the use of SIDUS (Single Instance Distributing Universal System) and by ACADEMICS Grant, under IDEXLYON project, within PIA ANR-16-IDEX-0005

$\underline{\Lambda} = (\lambda_1, \ldots, \lambda_M)$ and a $M \times M$ correlation matrix ρ_{mf}.

Analysis. It is well-documented that wavelet representations constitute reference tools to analyze multifractal properties in textures [17], [25]. Let $\{d_{X_m}(j, \underline{k})\}$, $m = 1, \ldots, M$, denote the discrete wavelet transform coefficients of the M-variate texture X, defined as inner products between each component X_m and dilated (at scale 2^j) and translated (at location $2^j \underline{k}$) templates a tensor-product based 2D wavelet [28].

Calculations not detailed here show that, for any pair of components $X_{m_1}(\underline{x}), X_{m_2}(\underline{x})$, the wavelet covariance $S_{m_1,m_2}(j) = \sum_k d_{X_{m_1}}(j,k) d_{X_{m_1}}(j,k)$ satisfies $S_{m_1,m_2}(2^j) \simeq \Sigma_{m_1,m_2} C_\psi 2^{j\zeta_{m_1,m_2}}$, 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 $C^{ss}_{m_1,m_2}(j) = S_{m_1,m_2}(j)/\sqrt{S_{m_1,m_1}(j) S_{m_2,m_2}(j)}$ 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 the 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 [29], [17]. Extending calculations in [18], [23], it can then be shown that the first $C_1^{(m)}(2^j)$ and second order $C_2^{(m_1,m_2)}(2^j)$ cumulants of $\ln L_X$ behave linearly in $\ln 2^j$: $C_1^{(m)}(2^j) = c_1^{(0,m)} + c_1^{(m)} \ln 2^j$, and $C_2^{(m_1,m_2)}(2^j) = c_2^{(0,m_1,m_2)} + c_2^{(m_1,m_2)} \ln 2^j$, with $c_2^{(m_1,m_2)} = \rho_{mf}(m_1,m_2) \lambda_{m_1}^2 \lambda_{m_2}^2$. This permits to define a wavelet multifractal coherence function as $C^{(mf)}_{m_1,m_2}(2^j) = C_2^{(m_1,m_2)}(2^j)/\sqrt{C_2^{(m_1,m_2)}(2^j) C_2^{(m_1,m_2)}(2^j)}$,, which boils down to $C^{(mf)}_{m_1,m_2}(2^j) = \rho_{mf}(m_1,m_2)$. As functions of scales 2^j, $C_1^{(m)}(2^j)$, $C_2^{(m_1,m_2)}(2^j)$ and $C^{(mf)}_{m_1,m_2}(2^j)$ thus characterize spatial dependencies amongst components not already encoded in the wavelet coherence functions.

These wavelet and wavelet-leader multiscale statistics will be used here to assess the quality of the DL-synthesized textures. They are implemented using the documented toolbox made available on the authors' websites.

III. Deep-Convolutional GAN

Generative adversarial networks constitute an increasingly popular class of deep learning architectures, widely used for image synthesis over the past years. Originally proposed in [3], GAN rely on the competition between two neural networks, the *Generator*, G, and the *Discriminator*, D. The Generator consists of a nonlinear filter producing, with white gaussian noise as input, *fake* images resembling *in some sense* target images. The Discriminator outputs the probability that image X, used as input, belongs to the target distribution. 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 [3]. GAN implementations were traditionally based on fully-connected layers, followed by (nonlinear) activation functions and max-pooling. However, the original GAN has been documented to be unstable and difficult to train. Numerous attempts to stabilize optimization were reported, e.g., [13], [16], [15].

Deep Convolutional GAN (DCGAN) were proposed in [13] to scale up the original GAN by 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 [13]. The Generator is fed with a low-dimensional white noise, and it progressively upsamples the output of each convolutional layer to the target resolution. The Discriminator is a standard CNN using strided convolutions to downsample the outputs.

Loss Functions. Cross Binary Entropy is used as loss function [3]. The optimal discriminator is obtained by maximizing $L_D(\theta_D) = \frac{1}{m}\sum_{i=1}^m \left[\log(D(x^{(i)})) + \log(1 - D(G(z^{(i)})))\right]$, with θ_D the weights for D, $x^{(i)}$, $z^{(i)}$ training and white noise samples. The generator is trained by minimizing $L_G(\theta_G) = \frac{1}{m}\sum_{i=1}^m \log(D(G(z^{(i)})))$, with θ_G the weights for G.

IV. Synthesis quality quantitative assessment

A. Experimental set-up

GPU facilities, Architectures, optimization and learning. Numerical experiments are conducted on dedicated workstations, each with several General Purpose Graphical Processing Unit (Nvidia RTX 2080 Super), under SIDUS Operating System and a Linux Debian Stretch distribution with Nvidia backported packages [30]. DCGAN are implemented using Keras, following architectures proposed in [13]. For the Discriminator, five convolutional layers are used, with strides $(2,2)$ and filters of size 3×3, and Leaky ReLU activation functions. For the Generator, white noise is filtered and resampled using a fully-connected layer, and a ReLu activation. Upsampling is performed by pixel duplications followed by convolutions with filters of size 3×3. For optimization, Adam amsgrad variant [31] is used, with learning rate of 2.10^{-4} and momentum of 0.5. Weights are initialized using uniform initialization [32]. The discriminator is trained using dropout at rate 1/4, batches of size of 32 and batch normalization [33]. The size of the input white Gaussian noise is set to 100.

Training dataset. The training dataset consists of 20000 independent copies of 4-variate MMRW textures, each of size 512×512, synthesized using Matlab routines devised by ourselves, implementing directly definitions via circulent matrix embedding[34]. Selfsimilarity and multifractality exponents are chosen equal ($\forall m, H_m = 0.7$ and $\lambda_m = \sqrt{0.03}$), so that all components have exactly the same statistics (notably same marginal distributions and covariance functions) and can hence not be distinguished. The off-diagonal entries of the cross-correlation Σ and cross-multifractal Σ_{mf} matrices are however chosen different for each pair of components, with some pairs having opposite signs for cross-correlation and cross-multifractality, other pairs have same signs, other pairs with cross-correlation but no cross-multifractality, and conversely. We believe that this constitutes a complicated texture model where all the richness lies in cross-dependencies, both at the second statistical order, and/or at higher-orders. A 4-variate sample of such MMRW textures is shown in Fig. 1 (top row). For MMRW, the corresponding theoretical wavelet coherence

$C^{(ss)}_{m_1,m_2}(2^j)$ and wavelet multifractal coherence $C^{(mf)}_{m_1,m_2}(2^j)$ functions are constant across scales 2^j), at levels that depend only on matrices Σ and Σ^{mf} respectively. These 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. 3 (blue lines, with lower triangles) and used as ground truth.

reproducibility. To test reproducibility, DCGAN training is repeated 24 times from scratch.

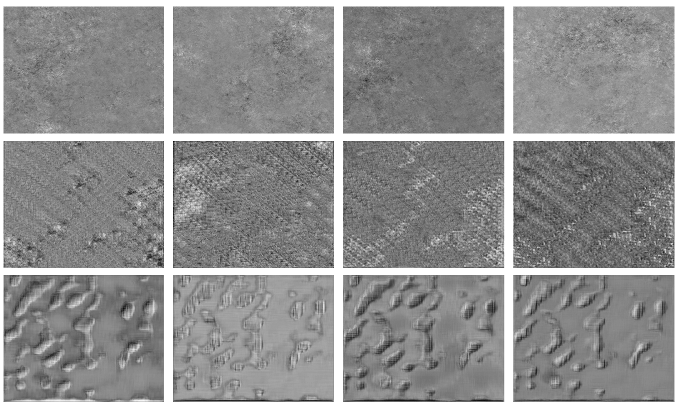

Fig. 1. **Samples of a 4-variate textures.** MMRW Textures (top row), DCGAN synthetized textures best trial (middle row), worst trial (bottom row).

B. DCGAN synthesis quality assessment

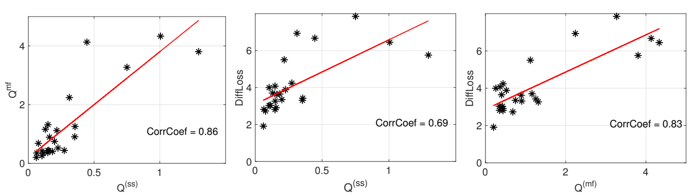

Fig. 2. **DCGAN synthesis quality assessment.** A posteriori Cross-Correlation index $Q^{(ss)}$ vs. Cross-Multifractality index $Q^{(mf)}$ (left), A posteriori $Q^{(ss)}$ (middle) and $Q^{(mf)}$ vs. a priori Loss-based, DiffLoss, indices (right).

Definition of wavelet based a posteriori indices. The wavelet and wavelet leader analyses described in Section II are applied to DCGAN generated textures. The functions $S_{m_1,m_2}(j)$, $C_1^{(m)}(2^j)$, $C_2^{(m)}(2^j)$, $C^{(ss)}_{m_1,m_2}(2^j)$ and $C^{(mf)}_{m_1,m_2}(2^j)$ are computed as averages from 10 samples of DCGAN generated textures (red stars) and compared against the ground truth functions obtained from average over 100 true MMRW textures (blue triangles), as illustrated in Fig. 3. Because the richness of 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 scale 2^j and across all (6) pairs of components. These indices are referred to as a posteriori as they can be computed only after training is completed, and DCGAN textures generated.

Quality assessment and reproducibility. Fig. 2(left) compares $Q^{(ss)}$ and $Q^{(mf)}$ for each of the independent training trials. It shows that i) $Q^{(mf)}$ is systematically larger than $Q^{(ss)}$, indicating that higher-order statistics are less easy to reproduce than 2nd order statistics ; ii) $Q^{(ss)}$ and $Q^{(mf)}$ are significantly correlated (correlation coefficient of 0.86), indicating that DCGAN synthesized textures reproduce cross-coherences (2nd order statistics) and cross-multifractalities (higher-order statistics) between pairs in comparable manner, i.e., DCGAN works for either all statistics or none ; iii) and, last and foremost, there is a significantly large variability across trials in the quality of the DCGAN synthesized textures.

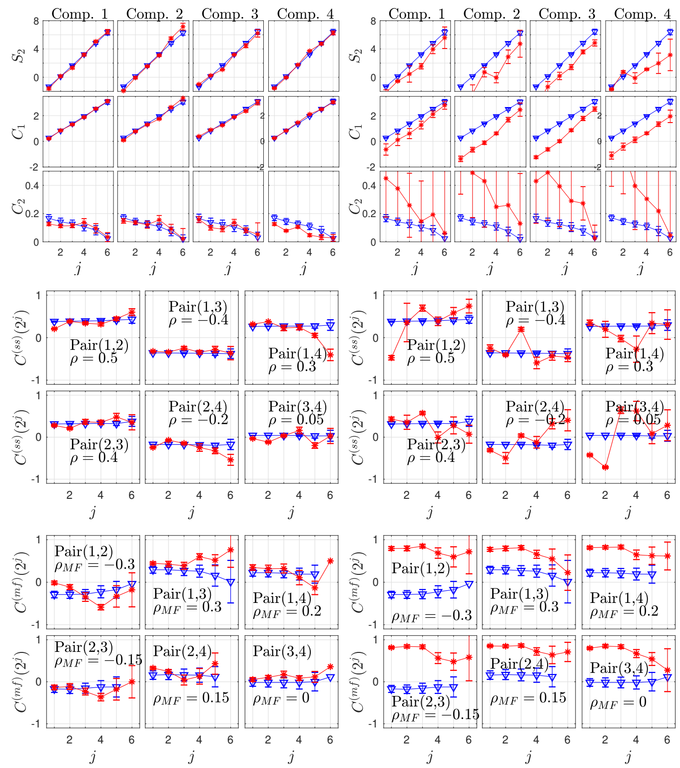

Fig. 3. **DCGAN synthesis quality assessment.** Univariate multiscale analysis $\log_2 S_2(2^j), C_1(2^j)$ and $C_2(2^j)$ for each component (top), Multiscale cross-coherence $C^{(ss)}_{m_1,m_2}(2^j)$ for each pair (middle), Multiscale cross-mutifractality $C^{(mf)}_{m_1,m_2}(2^j)$ for each pair (bottom). Best trial (left column), worst trial (right column).

To further inspect this large variability, Fig. 1 illustrates true MMRW (top) against DCGAN textures for the *best* and *worst* trials, corresponding respectively to the bottom left (smallest $Q^{(ss)}$ and $Q^{(mf)}$) and top right (largest $Q^{(ss)}$ and $Q^{(mf)}$) points in Fig. 2(left). Fig. 3 further compares ground truth functions $S_{m_1,m_2}(j)$, $C_1^{(m)}(2^j)$, $C_2^{(m)}(2^j)$, $C^{(ss)}_{m_1,m_2}(2^j)$ and $C^{(mf)}_{m_1,m_2}(2^j)$ (blue triangles) against those (red stars) obtained on average for the *best* (left) and *worst* (right) trials. For the best trial, Fig. 3(left) shows that, despite mild imperfections visible in Fig. 1(middle row), the statistics of DCGAN textures satisfactorily match those of true MMRW across all scales, both component-wise and across components, both at the 2nd order and at higher statistical orders. Notably, positive and/or negative cross-correlations and cross-multifractalities are well achieved. Conversely, for the *worst* trial, Fig. 1(bottom row) shows DCGAN textures strongly differ visually from true MMRW. In accordance, Fig. 3(right) shows quantitatively that DCGAN statistics do not reproduce those of the targeted MMRW, notably both univariate multifractality (higher-order statistics) and all cross-statistics (cross-correlations and cross-multifractalities) are totally missed. Additionally, quantile-

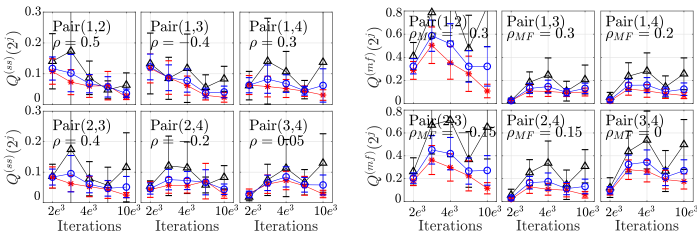

Fig. 4. **Convergence as functions of iterations.** Mean values and confidence intervals (from averages across trials) for $Q^{(ss)}$(left) and $Q^{(mf)}$ (right), for each pair of components, for all (blue), best only (red), worst only (black) trials.

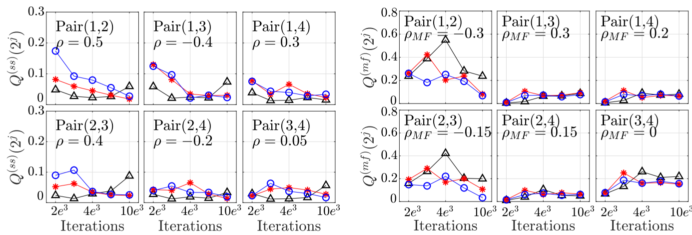

Fig. 5. **Convergence as functions of iterations, for learning datasets of different sizes.** Mean values and confidence intervals (from averages across trials) for $Q^{(ss)}$(left) and $Q^{(mf)}$ (right), for each pair of components, for best trials only, when the learning dataset sizes vary from 20000 (red stars) to 5000 (blue o) and 50 (black triangles) samples.

quantile plots, not reported here for space reasons, show that marginal distributions (1st order univariate statistics) are well reproduced for the best trial and not at all for the worst one.

In sum, DCGAN performance for texture synthesis can range from excellent to extremely poor across independent trials, despite identical settings (same architectures, training dataset, initialization procedures,. . .).

Convergence. To investigate the lack of reproducibility in DCGAN synthesis, Fig. 4 reports $Q^{(ss)}$ and $Q^{(mf)}$ for each pair of components independently as functions of the number of iterations, computed as average across *all*, *best* and *worst* trials (defined as $Q^{(ss)}$ and $Q^{(mf)}$ being respectively below or above empirically chosen threholds). Fig. 4 shows, for *best* trials, smooth decreases in $Q^{(ss)}$ and $Q^{(mf)}$ when the number of iterations increase, indicating a smooth convergence of the training. For *worst* trials, Fig. 4 shows an increase of $Q^{(ss)}$ and $Q^{(mf)}$ along iterations, indicating that the learning is diverging and that further iterating is not improving convergence: corresponding trials are definitely lost and useless. Interestingly, Fig. 4 also illustrates the benefits to iterate longer when larger (absolute values of) cross-correlation or cross-multifractality (hence more complicated cross-statistics) are targeted.

Impact of training set size. To test the impact of the training size on performance, the same experiments were repeated with training dataset sizes of 15000, 10000, 5000, 500, and 50 samples. Fig. 5 compares $Q^{(ss)}$ and $Q^{(mf)}$ independently for each pair of components as functions of the number of iterations for each training set size (for clarity, only 15000, 5000, and 50 are shown). These comparisons show that all conclusions drawn so far (lack of reproducibility and large variability in performance) are not impacted despite a drastic reduction in the training set size, from 20000 to 500 samples. A mild decrease in performance starts to be quantifiable for a training set of 50 samples only!

Loss function based a priori index. Indices $Q^{(ss)}$ and $Q^{(mf)}$ are of great use to assess the quality of the DCGAN textures and the reproducibility of the training procedure. They, however, are a posteriori indices: their computation requires that training is completed, textures are generated and analyzed and foremost 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 constructed an a priori index from the only quantities available during training: the losses functions as functions of iterations. Empirically, several constructions were tested and it turns out that the simplest, the absolute value of the difference of the Generator and Discriminator loss functions, hereafter DiffLoss, correlates significantly with both $Q^{(ss)}$ and $Q^{(mf)}$ (Fig. 2(middle and right)). This validates that this a priori index can 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 and an a priori known ground truth, and hence to continue iterating for a trial which will not converge in the end.

Loss function. Wasserstein loss functions [15], with same architectures, were also used for their potential better training properties. However, achieved results were equivalent in terms of overall large variability in performance and lack of reproducibility, and are hence not reported here.

V. Discussion and conclusions

The originality of the proposed work lied in defining a posteriori indices permitting to assess quantitatively the quality of DCGAN synthesized textures. This has permitted to quantify that indeed DCGAN architectures have significant potential to learn relevant features from sample textures. Here, focus was on cross-multifractality, a non trivial and intricate higher-order cross statistics. This has also permitted to show that this success comes at the price of a significantly large variability in performance and hence lack of reproducibility in training for the DCGAN architectures used here: Out of 24 independent trials, less than a half led to low enough indices $Q^{(ss)}$ and $Q^{(mf)}$, hence to relevant DCGAN textures. Also, it showed that, as opposed to common belief, the training set needs not be large and that 50 samples of 4-variate textures were enough to train the neural network. Finally, an a priori index, constructed from the loss functions, hence directly available and usable while the network is being trained, has been shown to be significantly correlated with the a posteriori quality indices $Q^{(ss)}$ and $Q^{(mf)}$, only available when the training has been completed. Further investigations include exploring a priori/a posteriori synthesis quality index designs and tuning architecture complexity to task complexity.

References

[1] V. Andrearczyk and P. Whelan, "Texture segmentation with fully convolutional networks," *arXiv:1703.05230*, 2017.

[2] L. Gatys, A. S. Ecker, and M. Bethge, "Texture synthesis using convolutional neural networks," in *Advances in neural information processing systems*, 2015, pp. 262–270.

[3] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[5] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[7] S. Basu, M. Karki, S. Mukhopadhyay, S. Ganguly, R. Nemani, R. DiBiano, and S. Gayaka, "A theoretical analysis of deep neural networks for texture classification," in *2016 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2016, pp. 992–999.

[8] D. Ulyanov, V. Lebedev, A. Vedaldi, and V. S. Lempitsky, "Texture networks: Feed-forward synthesis of textures and stylized images.," in *ICML*, 2016, p. 4.

[9] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[10] J. Bruna and S. Mallat, "Multiscale sparse microcanonical models," *Mathematical Statistics and Learning*, vol. 1, no. 3, pp. 257–315, 2019.

[11] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*, MIT press, 2016.

[12] T. Angles and S. Mallat, "Generative networks as inverse problems with scattering transforms," in *Proc. ICLR*, 2018.

[13] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[14] A. Brock, J. Donahue, and K. Simonyan, "Large scale gan training for high fidelity natural image synthesis," *Proc ICLR*, 2019.

[15] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *arXiv preprint arXiv:1701.07875*, 2017.

[16] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Advances in neural information processing systems*, 2016, pp. 2234–2242.

[17] H. Wendt, S. G. Roux, P. Abry, and S. Jaffard, "Wavelet leaders and bootstrap for multifractal analysis of images," *Signal Proces.*, vol. 89, no. 6, pp. 1100–1114, 2009.

[18] H. Wendt, R. Leonarduzzi, P. Abry, S. Roux, S. Jaffard, and S. Seuret, "Assessing cross-dependencies using bivariate multifractal analysis," in *IEEE Int. Conf. Acoust., Speech, and Signal Proces. (ICASSP)*, Calgary, Canada, April 2018.

[19] B. B. Mandelbrot, "Intermittent turbulence in self-similar cascades: divergence of high moments and dimension of the carrier," *J. Fluid Mech.*, vol. 62, pp. 331–358, 1974.

[20] E. Bacry, J. Delour, and J.-F. Muzy, "Multifractal random walk," *Phys. Rev. E*, vol. 64: 026103, 2001.

[21] C. Meneveau, K. Sreenivasan, P. Kailasnath, and M. Fan, "Joint multifractal measures - theory and applications to turbulence," *Physical Review A*, vol. 41, no. 2, pp. 894–913, January 1990.

[22] S. Jaffard, P. Abry, and H. Wendt, *Irregularities and Scaling in Signal and Image Processing: Multifractal Analysis*, pp. 31–116, World scientific publishing, Singapore, 2015.

[23] R. Leonarduzzi, P. Abry, S. G. Roux, H. Wendt, S. Jaffard, and S. Seuret, "Multifractal characterization for bivariate data," in *Proc. European Signal Processing Conference (EUSIPCO)*, Rome, Italy, September 2018.

[24] G. Samorodnitsky and M. Taqqu, *Stable non-Gaussian random processes*, Chapman and Hall, New York, 1994.

[25] S. G. Roux, M. Clausel, B. Vedel, S. Jaffard, and P. Abry, "Self-Similar Anisotropic Texture Analysis: The Hyperbolic Wavelet Transform Contribution," *IEEE Trans. Image Proc.*, vol. 22, no. 11, pp. 4353–4363, 2013.

[26] G. Didier, V. Pipiras, et al., "Integral representations and properties of operator fractional brownian motions," *Bernoulli*, vol. 17, no. 1, pp. 1–33, 2011.

[27] P. Abry, G. Didier, et al., "Wavelet estimation for operator fractional brownian motion," *Bernoulli*, vol. 24, no. 2, pp. 895–928, 2018.

[28] S. Mallat, *A Wavelet Tour of Signal Processing*, Academic Press, San Diego, CA, 1998.

[29] H. Wendt, P. Abry, and S. Jaffard, "Bootstrap for empirical multifractal analysis," *IEEE Signal Proc. Mag.*, vol. 24, no. 4, pp. 38–48, 2007.

[30] E. Quemener and M. Corvellec, "Sidus, the solution for extreme deduplication of an operating system," *The Linux Journal*, vol. January, 2014.

[31] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of adam and beyond," *arXiv preprint arXiv:1904.09237*, 2019.

[32] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[33] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[34] H. Helgason, V. Pipiras, and P. Abry, "Fast and exact synthesis of stationary multivariate Gaussian time series using circulant embedding," *Signal Processing*, vol. 91, no. 5, pp. 1123–1133, MAY 2011.