Exponential models

Bruno Galerne
bruno.galerne@univ-orleans.fr

Institut Denis Poisson
Université d'Orléans, Université de Tours, CNRS

Master MVA
Cours "Méthodes stochastiques pour l'image"
Lundi 15 mars 2021

Outline

Outline

Texture synthesis

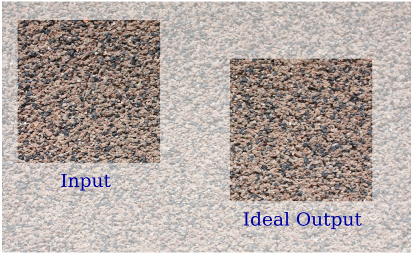
Macrocanonical models and exponential models

Exponential Models and Texture Synthesis

By-example Texture Synthesis

Notation:

- $\Omega \subset \mathbb{Z}^2$ finite discrete rectangle.
- Image $x : \Omega \to \mathbb{R}^3$
  $x(i) = (x_R(i), x_G(i), x_B(i))$
- π probability distribution on $\mathbb{R}^d$,
  $d = 3|\Omega|$
  (stationary random field).



By-example texture synthesis (with a probabilistic point of view):

- Analysis/Modeling: Estimate a (stationary) distribution π from an exemplar image x_0.
- Synthesis: Sample $x \sim \pi$.

Parametric Texture Synthesis

- Suppose that we have a family of statistical measurements ("features")

$$f = (f_k)_{1 \leqslant k \leqslant p} : \mathbb{R}^d \longrightarrow \mathbb{R}^p$$

  that captures the "perceptual aspect" of the texture, e.g. mean colors, color correlation, etc. (more examples next).
- We want to design a random field X on Ω such that

  $\mathbb{E}[f(X)] = f(x_0)$ **(macrocanonical model: same statistics in average)**.

  or even

  $f(X) = f(x_0)$ a.s. **(microcanonical model: exactly same statistics)**.
- We also need a model which is "as random as possible" (to avoid the trivial solution $X \sim \delta_{x_0}$)
- This will be achieved thanks to the **maximum entropy principle**.

Different Models for Different Statistics

- **Covariance/Fourier Spectrum**
  $\longrightarrow$ Sparse convolution, spectrum painting **[Lewis, 1984]**
  $\longrightarrow$ Spot noise, Random phase noise, Gaussian models
  [Van Wijk, 1991], **[Galerne et al., 2011]**, **[Xia et al., 2014]**
  $\longrightarrow$ Local random phase noise **[Gilet et al., 2014]**

- **Wavelet statistics**
  $\longrightarrow$ Histograms of subbands **[Heeger & Bergen, 1995]**
  $\longrightarrow$ First-order responses to a bank of filters FRAME **[Zhu et al., 1998]**
  $\longrightarrow$ Second-order wavelet statistics **[Portilla & Simoncelli, 2000]**
  $\longrightarrow$ First-order dictionary statistics + spectrum **[Tartavel et al., 2014]**

- **Neural networks statistics**
  $\longrightarrow$ First-order neural statistics **[Lu et al., 2015]**
  $\longrightarrow$ Second-order neural statistics **[Gatys et al., 2015]**

- **Scattering statistics**
  $\longrightarrow$ First-order scattering statistics **[Zhang & Mallat, 2017]**, **[Bruna & Mallat, 2019]**

Different Models for Different Statistics

Green: Macrocanonical Models = maximum entropy model with same statistics in expectation
Red: Microcanonical models = maximum entropy model with exactly same statistics

- **Covariance/Fourier Spectrum**
  $\longrightarrow$ Sparse convolution, spectrum painting **[Lewis, 1984]**
  $\longrightarrow$ Spot noise, **Random phase noise**, **Gaussian models**
  [Van Wijk, 1991], **[Galerne et al., 2011]**, **[Xia et al., 2014]**
  $\longrightarrow$ Local random phase noise **[Gilet et al., 2014]**

- **Wavelet statistics**
  $\longrightarrow$ **Histograms of subbands [Heeger & Bergen, 1995]**
  $\longrightarrow$ **First-order responses to a bank of filters FRAME [Zhu et al., 1998]**
  $\longrightarrow$ **Second-order wavelet statistics [Portilla & Simoncelli, 2000]**
  $\longrightarrow$ **First-order dictionary statistics + spectrum [Tartavel et al., 2014]**

- **Neural networks statistics**
  $\longrightarrow$ **First-order neural statistics [Lu et al., 2015]**
  $\longrightarrow$ **Second-order neural statistics [Gatys et al., 2015]**

- **Scattering statistics**
  $\longrightarrow$ **First-order scattering statistics [Zhang & Mallat, 2017], [Bruna & Mallat, 2019]**

Entropy

Let $\mathcal{P}$ be the set of probability distributions on $\mathbb{R}^d$.

Let μ be a reference probability measure on $\mathbb{R}^d$ (e.g. $\mu(dx) \propto e^{-J(x)}dx$ where $J(x) = \frac{\varepsilon}{2}\|x\|^2$, that is a white Gaussian noise distribution)

The entropy $H : \mathscr{P} \to [-\infty, +\infty)$ (w.r.t. μ) is defined by

$$\forall \pi \in \mathcal{P}, \quad H(\pi) = \begin{cases} -\displaystyle\int_{\mathbb{R}^d} \log\left(\frac{\mathrm{d}\pi}{\mathrm{d}\mu}(x)\right)\frac{\mathrm{d}\pi}{\mathrm{d}\mu}(x)\mu(\mathrm{d}x) & \text{if } \dfrac{\mathrm{d}\pi}{\mathrm{d}\mu} \text{ exists} \\ -\infty & \text{otherwise} \end{cases}$$

Macrocanonical/Microcanonical Models

Definition

Let $x_0 \in \mathbb{R}^d$ be the exemplar texture and $f : \mathbb{R}^d \to \mathbb{R}^p$ measurable.

- A **microcanonical** model associated with x_0 for the statistics f (with reference measure μ) is a probability distribution $\pi \in \mathcal{P}$ that solves

$$\max H(\pi)$$

over all $\pi \in \mathcal{P}$ such that $X \sim \pi \Rightarrow f(X) = f(x_0)$ a.s.

- A **macrocanonical** model associated with x_0 for the statistics f (with reference measure μ) is a probability distribution $\pi \in \mathcal{P}$ that solves

$$\max H(\pi)$$

over all $\pi \in \mathcal{P}$ such that $\mathbb{E}_{X\sim\pi}[f(X)] = f(x_0)$.

Course plan

Last week:

- Variational texture synthesis: Microcanonical models only
- Three algorithms discussed
- Lab session using CNN [Gatys et al 2015]
- No entropy maximization... so only approximately microcanonical!

Today:

- Mathematics for macrocanonical models: existence, entropy maximization, exponential models,...
- Sampling of macrocanonical models
- Lab session on sampling using Langevin dynamics
- Maximal entropy for texture synthesis

Motivation

Microcanonical models limitations...

- Only approximate in practice:
  1. Start with Gaussian white noise (that has maximal entropy)
  2. Minimize energy like $E(x) = \|f(x) - f(x_0)\|^2$ with some descent algorithm.
- Process recently studied by **[Bruna & Mallat, 2019]** and called " Microcanonical Gradient Descent Model".
  - Gradient descent transports the initial Gaussian distribution to the set of critical points of E.
  - The final distribution does not have maximal entropy (but some bounds are derived).

Motivation for studying macrocanonical models...

- Principled formulation of by-example texture synthesis.
- Link with the *modified* Julesz conjecture (1981):

  "It seems that only the first-order statistics of these textons [non-linear features] *have perceptual significance."*

- Helps to better understand the chosen statistics/features.
- Connections with nice results on MCMC and stochastic optimization.

Motivation

Microcanonical models limitations...

- Only approximate in practice:
  1. Start with Gaussian white noise (that has maximal entropy)
  2. Minimize energy like $E(x) = \|f(x) - f(x_0)\|^2$ with some descent algorithm.
- Process recently studied by **[Bruna & Mallat, 2019]** and called "Microcanonical Gradient Descent Model".
  - Gradient descent transports the initial Gaussian distribution to the set of critical points of E.
  - The final distribution does not have maximal entropy (but some bounds are derived).

Motivation for studying macrocanonical models...

- Principled formulation of by-example texture synthesis.
- Link with the *modified* Julesz conjecture (1981):

  "It seems that only the first-order statistics
  of these textons [non-linear features] *have perceptual significance."*
- Helps to better understand the chosen statistics/features.
- Connections with nice results on MCMC and stochastic optimization.

Outline

Maximum Entropy Principle and Exponential Models

- What follows is a generalization of [Mumford and Desolneux 2010] chapter 4.
- The main reference for texture synthesis is: S. Zhu, Y. Wu, and D. Mumford, *Filters, random fields and maximum entropy (FRAME): Towards a unified theory for texture modeling*, International Journal of Computer Vision, 27 (1998)
- Maximal entropy ideas go back to the 50s: E. T. Jaynes, *Information Theory and Statistical Mechanics* Phys. Rev., 1957

Maximum Entropy Principle and Exponential Models

For $\theta \in \mathbb{R}^p$, if $e^{-\theta \cdot f} \in L^1(\mu)$, we define

$$\pi_\theta(dx) = \frac{1}{Z(\theta)} e^{-\theta \cdot f(x)} \mu(dx) = p_\theta(x)\mu(dx) \qquad \text{where} \qquad Z(\theta) = \int_{\mathbb{R}^d} e^{-\theta \cdot f(x)} \mu(dx).$$

- The (intractable) constant $Z(\theta)$ is called the **partition function**.

Theorem (De Bortoli, Desolneux, Galerne, Leclaire, 2019)

Assume that

a) $\forall \theta \in \mathbb{R}^p, \quad \int_{\mathbb{R}^d} e^{\|\theta\| \|f(x)\|} \mu(dx) < \infty,$

b) $\forall \theta \in \mathbb{R}^p, \quad \mu\left(\left\{x \in \mathbb{R}^d \mid \theta \cdot f(x) < \theta \cdot f(x_0)\right\}\right) > 0.$

Then there exists $\theta_* \in \mathbb{R}^p$ such that π_{θ_*} is a macrocanonical model associated with x_0 for the statistics f. Besides, θ_* is a solution to the convex minimization problem

$$\operatorname{argmin}_{\theta \in \mathbb{R}^p} \left(\theta \cdot f(x_0) + \log Z(\theta)\right) = \operatorname{argmin}_{\theta \in \mathbb{R}^p} \log \left(\int_{\mathbb{R}^d} e^{-\theta \cdot (f(x) - f(x_0))} \mu(dx)\right) .$$

Proof: Guided Exercice

$$\pi_\theta(dx) = \frac{1}{Z(\theta)} e^{-\theta\cdot f(x)}\mu(dx) = p_\theta(x)\mu(dx) \qquad \text{where} \qquad Z(\theta) = \int_{\mathbb{R}^d} e^{-\theta\cdot f(x)}\mu(dx).$$

Assumptions:

a) $\forall \theta \in \mathbb{R}^p, \quad \int_{\mathbb{R}^d} e^{\|\theta\|\|f(x)\|}\mu(dx) < \infty,$

b) $\forall \theta \in \mathbb{R}^p, \quad \mu\left(\left\{x \in \mathbb{R}^d \mid \theta\cdot f(x) < \theta\cdot f(x_0)\right\}\right) > 0.$

Main idea: The parameter θ_* can be found by maximum-likelihood.

$$L(\theta) = \log p_\theta(x_0) = -\theta\cdot f(x_0) - \log Z(\theta).$$

Existence step:

1. Show that $Z(\theta) = \int_{\mathbb{R}^d} e^{-\theta\cdot f(x)}\mu(dx)$ is well-defined.
2. Show that $Z(\theta)$ is differentiable, compute $\frac{\partial L}{\partial \theta_k}$, and conclude that
$$\nabla L(\theta) = \mathbb{E}_{\pi_\theta}[f(X)] - f(x_0)\ .$$
3. Similarly,
$$\nabla^2 L(\theta) = -\mathbb{E}_{\pi_\theta}\left[(f(X)-\mathbb{E}_{\pi_\theta}[f(X)])(f(X)-\mathbb{E}_{\pi_\theta}[f(X)])^T\right] = -\operatorname{Cov}_{\pi_\theta}(f(X))$$
4. Convexity of L?
5. Show that for all $\theta \in \mathbb{R}^p$, $-L(t\theta) \to +\infty$ as $t \to +\infty$ (i.e. $-L$ coercive along each direction).

Proof: Guided Exercice

- We conclude the existence using that $-L$ is convex, continuous and coercive along each direction, which implies that $-L$ is covercive.
- So we have some θ_* maximizing $L(\theta)$.

Maximal entropy:
The entropy $H : \mathscr{P} \to [-\infty, +\infty)$ (w.r.t. μ) is defined by

$$\forall \pi \in \mathcal{P}, \quad H(\pi) = \begin{cases} -\int_{\mathbb{R}^d} \log\left(\frac{\mathrm{d}\pi}{\mathrm{d}\mu}(x)\right) \frac{\mathrm{d}\pi}{\mathrm{d}\mu}(x)\mu(\mathrm{d}x) & \text{if } \frac{\mathrm{d}\pi}{\mathrm{d}\mu} \text{ exists} \\ -\infty & \text{otherwise} \end{cases}$$

Let us recall that, for μ_1 absolutely continuous with respect to μ_2, the Kullback-Leibler divergence

$$KL(\pi_1|\pi_2) = \int_{\mathbb{R}^d} \log\left(\frac{d\mu_1}{d\mu_2}(x)\right) \mu_1(dx)$$

is always non negative.

1. For $\theta \in \mathbb{R}^p$, compute $H(\pi_\theta)$.
2. Let π be another distribution such that $\mathbb{E}_\pi(f(X)) = f(x_0)$. Show that $H(\pi) \leq H(\pi_{\theta_*})$, i.e. π_{θ_*} is a macrocanonical model.

Microcanonical model have exponential form

- A **macrocanonical** model associated with x_0 for the statistics f (with reference measure μ) is a probability distribution $\pi \in \mathcal{P}$ that solves

$$\max H(\pi)$$

  over all $\pi \in \mathcal{P}$ such that $\mathbb{E}_{X\sim\pi}[f(X)] = f(x_0)$.
- Problem in a large measure space "$\pi \in \mathcal{P}$ such that $\mathbb{E}_{X\sim\pi}[f(X)] = f(x_0)$." boils down to: $\pi = \pi_{\theta_*}$ with θ_* a solution to the convex minimization problem in $\mathbb{R}^p$

$$\operatorname{argmin}_{\theta\in\mathbb{R}^p} \Big(\theta\cdot f(x_0) + \log Z(\theta)\Big) = \operatorname{argmin}_{\theta\in\mathbb{R}^p} \log\left(\int_{\mathbb{R}^d} e^{-\theta\cdot(f(x)-f(x_0))}\mu(dx)\right) .$$

- Next question: Estimate a solution θ_*.

Model Estimation

- θ_* can be estimated by gradient descent to log-likelihood $-L$.
- $\nabla L(\theta) = \mathbb{E}_{\pi_\theta}[f(X)] - f(x_0)$.
- A Monte-Carlo method must generally be used to estimate $\nabla L(\theta)$.

Algorithm: Estimate θ_* from exemplar image x_0

- Compute observed statistics $f(x_0)$.
- Initialize $\theta \leftarrow 0$.
- For $n = 1, \dots, N$,
  - Sample $x_1, \dots, x_m \sim \pi_\theta$
  - Compute estimated statistics $f(x_j)$.
  - Update $\theta \leftarrow \theta + \delta_n \underbrace{\left(\frac{1}{m} \sum_{j=1}^{m} f(x_j) - f(x_0) \right)}_{\text{unbiased estimator of } \nabla L(\theta)}$
- Return θ.

- How to sample π_θ ?

How to sample π_θ ?

Let

$$V(x,\theta) = \theta \cdot (f(x) - f(x_0)) + J(x) \qquad \text{so that} \qquad \pi_\theta(x) \propto e^{-V(x,\theta)} dx.$$

We consider the **Langevin dynamics**

$$X_{n+1} = X_n - \gamma_{n+1} \nabla_x V(X_n, \theta) + \sqrt{2\gamma_{n+1}} Z_n$$

where

- (Z_n) is a collection of independent normalized Gaussian white noises
- $\gamma_n \geqslant 0$ is a sequence of step sizes

Equivalently, (X_n) is a inhomogeneous Markov chain with kernel

$$R_{\gamma_n}(x, \cdot) = \mathcal{N}(x - \gamma_n \nabla_x V(x,\theta), 2\gamma_n).$$

- If $\gamma_n = \gamma$ is constant, (X_n) has some stationary distribution $\Pi_\gamma \neq \pi_\theta$.
- But if γ decreases...

Theorem (Durmus, Moulines, 2016)

Under some hypotheses on V, and if $\sum \gamma_n = +\infty$ and $\sum \gamma_n^2 < \infty$, we have

$$X_n \xrightarrow[n\to\infty]{(d)} \pi_\theta$$

Sampling a GMM with Langevin Dynamics

Let

$$V(x,\theta) = \theta \cdot (f(x) - f(x_0)) + J(x) \qquad \text{so that} \qquad \pi_\theta(x) \propto e^{-V(x,\theta)} dx.$$

Langevin dynamics

$$X_{n+1} = X_n - \gamma_{n+1} \nabla_x V(X_n, \theta) + \sqrt{2\gamma_{n+1}} Z_n$$

where

- (Z_n) is a collection of independent normalized Gaussian white noises
- $\gamma_n \geqslant 0$ is a sequence of step sizes

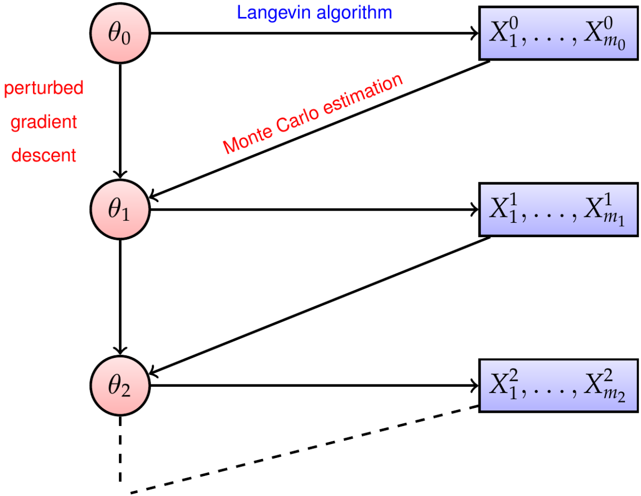
Practical session 1:

- Download the two files :
  - `www.idpoisson.fr/galerne/mva/TP_Langevin_SOUL.ipynb`
  - `www.idpoisson.fr/galerne/mva/draw_functions.py`
- Run the first part on Langevin sampling. Observe that with GMM having close modes the sampling is better than with GMM with well separated modes.

Combining dynamics

- Minimizing $-L(\theta)$ requires samples of π_θ to compute the gradient.
- Sampling π_θ requires a Langevin Markov Chain.
- **Combining dynamics:** Use the Langevin Markov chain to estimate the gradient...

Combining dynamics



– parameter sequence $\in \mathbb{R}^p$ (optimization)

– image sequence $\in \mathbb{R}^d$ (sampling)

Combined Dynamics

- Main idea: Use Langevin dynamic intermediary steps to approximate $\nabla L(\theta) = \mathbb{E}_{\pi_\theta} \nabla_\theta V(\theta, X)$.
- Stochastic Optimization with Unadjusted Langevin (SOUL).

SOUL algorithm

- Initialization: $\theta \leftarrow 0$; $X_0^0 \in \mathbb{R}^d$
- For $n = 1, \ldots, N$,
  - m_n steps of Langevin diffusion: for $k = 0, \ldots, m_n - 1$,

$$X_{k+1}^n = X_k^n - \gamma_{n+1} \nabla_x V(X_k^n, \theta_n) + \sqrt{2\gamma_{n+1}} Z_{k+1}^n$$

    with $Z_{k+1}^n \sim \mathcal{N}(0, I)$
  - Update θ with Langevin intermediary states:

$$\theta_{n+1} = \mathrm{Proj}_\Theta \left(\theta_n - \frac{\delta_{n+1}}{m_n} \sum_{k=1}^{m_n} \nabla_\theta V(X_k^n, \theta_n) \right)$$

  - Set warm start for next step: $X_0^{n+1} = X_{m_n}^n$

where Θ is a closed convex set of $\mathbb{R}^d$.

Convergence of SOUL algorithm

Notice that $-L$ is convex, $\mathscr{C}^1$ with Lipschitz gradient on Θ compact.

Theorem (De Bortoli, Durmus, Pereyra, Fernandez Vidal, 2019)

Assume that

1. Θ *is a convex compact set of* $\mathbb{R}^p$.
2. $J, f_1, \ldots, f_p$ *are differentiable on* $\mathbb{R}^d$ *with Lipschitz gradients.*
3. *There exist* $\eta, c, M > 0$ *such that*
   $\forall \theta \in \Theta,\ \forall x \in \mathbb{R}^d, \quad \langle \nabla_x V(x, \theta), x \rangle \geqslant \eta \|x\|^2 \mathbf{1}_{|x|>M} - c.$
4. $(\delta_n), (\gamma_n)$ *are non-increasing positive with* δ_0, γ_0 *sufficiently small and*

$$\sum \delta_n = +\infty, \quad \sum \delta_{n+1} \sqrt{\gamma_n} < \infty, \quad \sum \frac{\delta_{n+1}}{m_n \gamma_n} < \infty.$$

Then $\theta_n \longrightarrow \theta_* \in \mathrm{argmin}(-L)$ *almost surely and in* L^1.

NB: f may be non-convex (e.g. with differentiable neural networks).

Outline

Exponential Models for Textures

- Assume for simplicity that $x(i) \in \mathbb{R}$ for all $i \in \Omega$ (graylevel images).
- Let us consider $f(x) = (\bar{x}, x * \tilde{x})$ with

$$\bar{x} = \frac{1}{|\Omega|} \sum_{i \in \Omega} x(i) \qquad \text{and} \qquad \forall i \in \Omega, \quad x * \tilde{x}(i) = \sum_{i' \in \Omega} x(i')x(i+i').$$

- Then the associated macrocanonical model reads as

$$\pi_\theta(dx) = \frac{1}{Z(\theta)} \exp\left(-\theta_0 \bar{x} - \sum_{i,i' \in \Omega} \theta(i) x(i') x(i+i') - \frac{\varepsilon}{2} \|x\|^2 \right) dx.$$

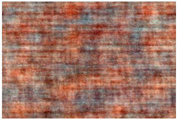
- This is a stationnary Gaussian model.
- For $\theta = \theta_*$ one has the Gaussian r.f. with same mean and covariance as the example x_0 (up to a Gaussian white noise of order ε): this is the ADSN model **[Galerne et al, 2011]**.

Exponential Models for Textures

Original

Gaussian

Original



Gaussian

- **Remark:** If (k_j) is a bank of *linear* filters and one takes $f_{j,j'}(x) = \frac{1}{|\Omega|} \sum_{i \in \Omega} k_j * x * \widetilde{k_{j'} * x}(i)$, then the associated macrocanonical model is still a Gaussian distribution.
- Going beyond the Gaussian model requires non-linear filters...

Exponential models and feature distribution

- A priori one can find an exponential model with the same expectation than a feature f_i.
- But one can in fact approximate the whole marginal distribution by enriching the set of features using indicator functions.
- Indeed if one divides $\mathbb{R}$ in bins $B_1 \cup B_2 \cup \ldots \ldots \cup B_m$ then one can consider the family of nm features

$$f'_{i,j}(x) = \mathbb{1}_{B_j}(f_i(x)).$$

- This allows for **approximating histograms of filter responses**.

FRAME: Exponential models and texture modeling

- The authors of [Zhu Wu Mumford 1998] use exponential models to model textures.
- The features f_i are the indicator function of locally supported filters (33×33 at most) :

$$f_{\alpha,j}(I) = \mathbb{1}_{B_j}(F^{(\alpha)} * I) = \mathbb{1}_{B_j}(I^{(\alpha)}) = H_j^{(\alpha)}.$$

Each filter response is quantized in L bins, so one filter $F^{(\alpha)}$ counts for L "features" f_i.

The FRAME distribution model with K filters S_K is

$$p(I; \Lambda_K, S_K) = \frac{1}{Z(\Lambda_K)} \exp\left(- \sum_{\alpha=1}^{K} \sum_{j=1}^{L} \lambda_j^{(\alpha)} H_j^{(\alpha)} \right)$$

- Hence this makes the exponential model a stationary Gibbs or equivalently a stationary Markov Random Field :

$$p(I(v)|I(-v)) = p(I(v)|I(\mathcal{N}_v))$$

where $\mathcal{N}_v$ is a local neighborhood of v.

FRAME: Exponential models and texture modeling

The FRAME distribution model with K filters S_K is

$$p(I; \Lambda_K, S_K) = \frac{1}{Z(\Lambda_K)} \exp\left(\sum_{\alpha=1}^{K} \sum_{j=1}^{L} \lambda_j^{(\alpha)} H_j^{(\alpha)} \right)$$

Practical problems:

1. Given a set of filters S_K, compute the optimal weights $\Lambda_K = (\lambda_j^{(\alpha)})$ (role of the a_i coefficients in the Theorem).
2. Given a set of filters S_K and weights Λ_K, how can we sample from $p(I; \Lambda_K, S_K)$... that is perform texture synthesis !
3. Given an input texture I^{obs} how can we select the good filters S_K for I^{obs} ?

FRAME Algorithm

From [Zhu Wu Mumford 1998]:

$$\frac{d\lambda^{(\alpha)}}{dt} = E_{p(\mathbf{I};\Lambda_K,S_K)}\left[H^{(\alpha)}\right] - H^{\mathrm{obs}(\alpha)}. \qquad (19)$$

Algorithm 1. The FRAME Algorithm

Input a texture image $\mathbf{I}^{\mathrm{obs}}$.
Select a group of K filters $S_K = \{F^{(1)}, F^{(2)}, \ldots, F^{(K)}\}$.
Compute $\{H^{\mathrm{obs}(\alpha)}, \quad \alpha = 1, \ldots, K\}$.
Initialize $\lambda_i^{(\alpha)} \leftarrow 0, \quad i = 1, 2, \ldots, L, \quad \alpha = 1, 2, \ldots, K$.

Initialize $\mathbf{I}^{\mathrm{syn}}$ as a uniform white noise texture.
Repeat
  Calculate $H^{\mathrm{syn}(\alpha)}$ $\alpha = 1, 2, \ldots, K$ from $\mathbf{I}^{\mathrm{syn}}$, use it for $E_{p(\mathbf{I};\Lambda_K,S_K)}(H^{(\alpha)})$.
  Update $\lambda^{(\alpha)}$ $\alpha = 1, 2, \ldots, K$ by Eq. (19), $p(\mathbf{I}; \Lambda_K, S_K)$ is updated.
  Apply Gibbs sampler to flip $\mathbf{I}^{\mathrm{syn}}$ for w *sweeps* under $p(\mathbf{I}; \Lambda_K, S_K)$
Until $\frac{1}{2}\sum_i^L |H_i^{\mathrm{obs}(\alpha)} - H_i^{\mathrm{syn}(\alpha)}| \leq \epsilon$ for $\alpha = 1, 2, \ldots, K$.

FRAME: Gibbs sampler

Extract from [Zhu Wu Mumford 1998]:

Algorithm 2. The Gibbs Sampler for *w* Sweeps

Given image $\mathbf{I}(\vec{v})$, flip_counter $\leftarrow 0$

Repeat

    Randomly pick a location $\vec{v}$ under the uniform distribution.

    For val $= 0, \ldots, G-1$ with G being the number of grey levels of $\mathbf{I}$

        Calculate $p(\mathbf{I}(\vec{v}) = \text{val} \mid \mathbf{I}(-\vec{v}))$ by $p(\mathbf{I}; \Lambda_K, S_K)$.

    Randomly flip $\mathbf{I}(\vec{v}) \leftarrow \text{val}$ under $p(\text{val} \mid \mathbf{I}(-\vec{v}))$.

    flip_counter $\leftarrow$ flip_counter $+ 1$

Until flip_counter $= w \times M \times N$.

Remark: Thanks to the Markov property, computing $p(I(v) = \text{val}|I(-v)) = p(I(v) = \text{val}|I(\mathcal{N}_v))$ only depends on the local neighborhood of v.

- [Zhu Wu Mumford 1998] also proposes a filter selection algorithm.

FRAME: Synthesis results

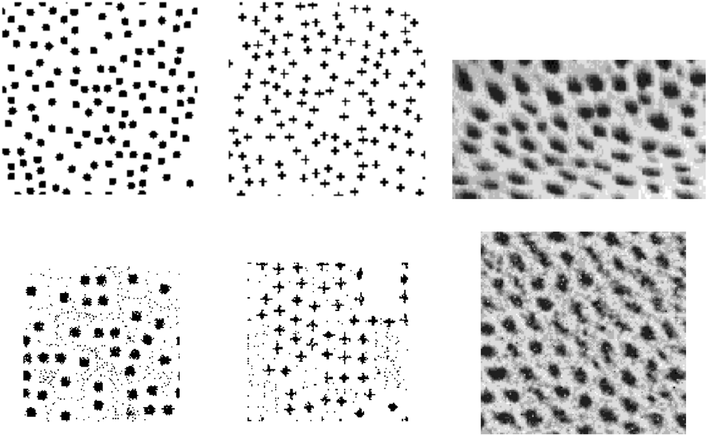


Figure 4.19: Three textures shown on the top are synthesized on the bottom, using exponential models and the learning model described in the text to fit the potentials ψ. The first two images use only one filter, the cheetah fur uses 6.

- FRAME model is limited to quantized images (8 greylevels)
- Reference measure μ is the uniform distribution on $\{0, \ldots, 7\}^{\Omega}$.
- Aboud 1 day for a 128×128 texture at the time !

FRAME: Pros and cons

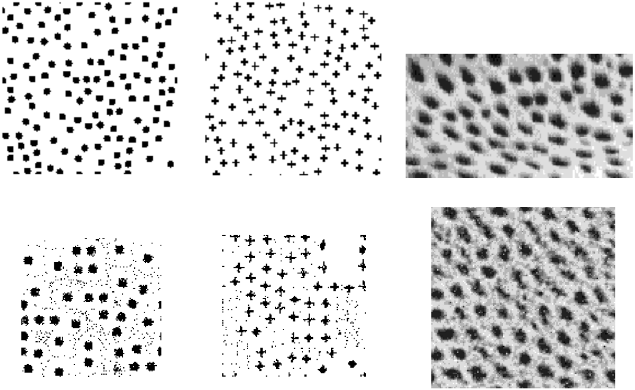
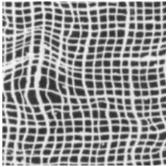


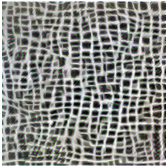
Figure 4.19: Three textures shown on the top are synthesized on the bottom, using exponential models and the learning model described in the text to fit the potentials ψ. The first two images use only one filter, the cheetah fur uses 6.

- Very nice mathematical modeling
- Very heavy computational cost
- Limited to (highly) quantized images

Exponential Models for Textures



Original



Synthesis

Original

Synthesis

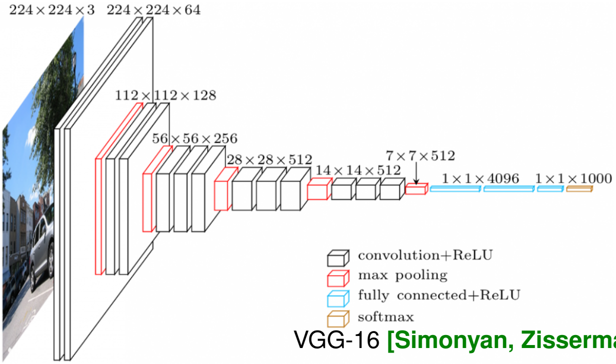
- DeepFRAME: Model using CNN **[Lu, Zhu, Wu, 2016]**
- The features extract the **spatial average** of responses to a given layer of a **pre-learned** convolutional neural network (CNN)

$$f_k(x) = \frac{1}{|\Omega|} \sum_{i \in \Omega} \mathcal{F}_k(x)(i)$$

where $(\mathcal{F}_k(x))_{1 \leqslant k \leqslant p}$ is the response at one particular layer of a CNN.

Statistics used in DeepFrame

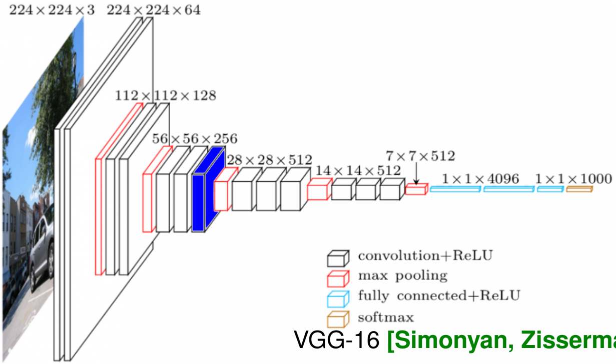
They use the CNN designed by the Visual Geometry Group (VGG) in Oxford.



VGG-16 **[Simonyan, Zisserman, 2015]**

Statistics used in DeepFrame

They use the CNN designed by the Visual Geometry Group (VGG) in Oxford.



VGG-16 **[Simonyan, Zisserman, 2015]**

Neural Network Features

- Let us consider the feature responses of each layer 1,...,p :

$$\forall x \in \mathbb{R}^d, \quad \mathcal{F}(x) = (\mathcal{F}_1(x), \ldots, \mathcal{F}_p(x)) \in \prod_{k=1}^{p} \mathbb{R}^{d_k}$$

  where $\mathcal{F}_k(x)$ is one response to a layer of a CNN with a non-linear unit $\varphi \in \mathscr{C}^1(\mathbb{R})$.
- More precisely,

$$\mathcal{F}_j(x) = \varphi(A_j(\mathcal{F}_{j-1}(x))) = (\varphi \circ A_j \circ \varphi \circ A_{j-1} \circ \ldots \circ \varphi \circ A_1)(x)$$

  where $A_j : \mathbb{R}^{n_j} \to \mathbb{R}^{n_{j+1}}$ are affine maps, and $\varphi : \mathbb{R} \to \mathbb{R}$ is a non-linear unit applied on each component.
- **Example:** For a convolutional neural network,

$$A_j(y) = k_j * y + b_j$$

  where $k_j : \Omega_j \to \mathbb{R}^{n_{j+1} \times n_j}$ is a matrix convolution kernel (with small support, e.g. 3×3 for VGG).
- (Empirical) combination of Langevin dynamics and exponential weights updates.

Experiments for SOUL for texture synthesis

Next results and experiments are from [De Bortoli et al 2021] :
Maximum entropy methods for texture synthesis: theory and practice,
V. De Bortoli, A. Desolneux, A. Durmus, B. Galerne, A. Leclaire, SIAM Journal on Mathematics of Data Science (SIMODS), 2021

Neural Network Features

- We consider as texture statistics the spatial average of the feature responses of each layer:

$$f(x) = \left(\frac{1}{d_1} \sum_{i=1}^{d_1} \mathcal{F}_1(x)(i), \dots, \frac{1}{d_p} \sum_{i=1}^{d_p} \mathcal{F}_p(x)(i) \right) .$$

- The corresponding macrocanonical model is stationary (because f is translation invariant).

Proposition ([De Bortoli et al 2019])

Let $x_0 \in \mathbb{R}^d$ and assume that $df(x_0)$ has rank $\min(d,p) = p$.
Assume that $\varphi \in \mathscr{C}^1(\mathbb{R})$ and that

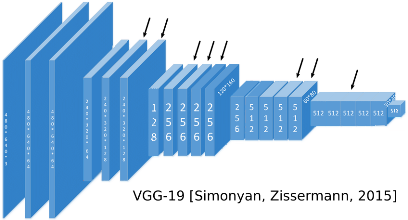
$$\exists c > 0,\ \forall x \in \mathbb{R}, \quad |\varphi(x)| \leqslant c(1 + |x|).$$

Then the maximum entropy principle holds with $J(x) = \frac{\varepsilon}{2}\|x\|^2$ for any $\varepsilon > 0$.

- Similar result with non-smooth RELU $\phi(t) = \max(t, 0)$.
  [De Bortoli et al 2021].

Experimental setup

- $f(x)$: spatially averaged reponses to *differentiable* VGG-19 at layers $3, 4, 5, 6, 7, 11, 12, 14$.
- Initialization: Gaussian random field with correct second-order statistics.
- $\delta_n = \mathcal{O}(\frac{1}{n})$, $\gamma_n = \mathcal{O}(\frac{1}{n})$, $m_n = 1$
- $\varepsilon = 0.1$ i.e. $\mu(dx) \propto e^{-0.05\|x\|^2}$
- $\Theta = \mathbb{R}^p$ (no projection)
- The color distribution is reimposed by adding mean color and covariance matrix feature.



VGG-19 [Simonyan, Zissermann, 2015]

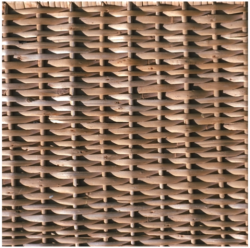
Synthesis Results

Original (256×256)

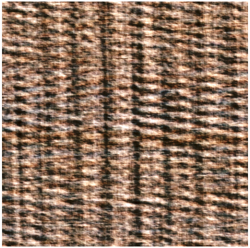
Initialization (Gaussian)

After 5000 iterations

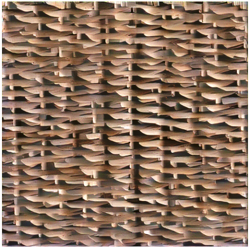
Synthesis Results



Original (512 $\times$ 512)

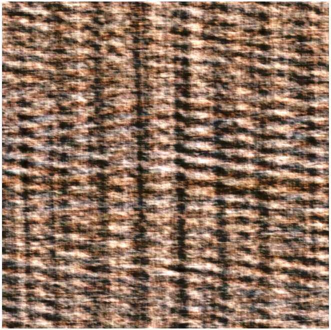


Initialization (Gaussian)



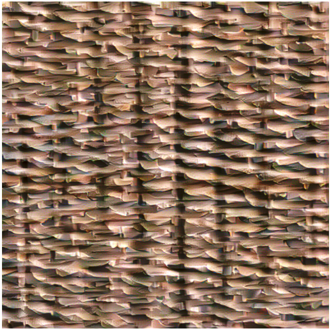
After 5000 iterations

Empirical Convergence



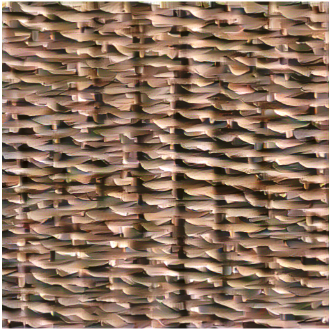
Iteration 0

Empirical Convergence



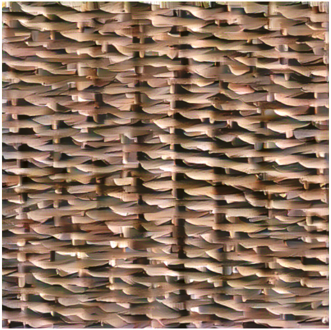
Iteration 100

Empirical Convergence



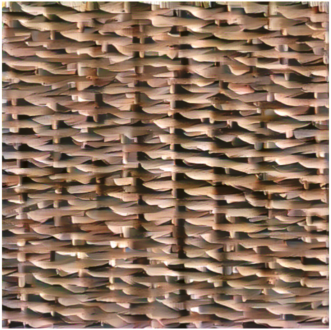
Iteration 200

Empirical Convergence



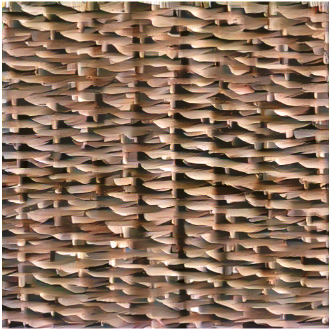
Iteration 300

Empirical Convergence



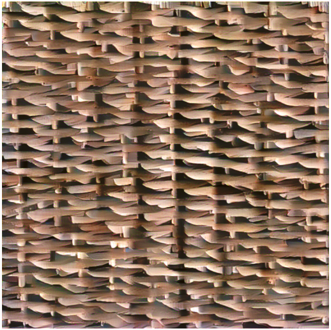
Iteration 400

Empirical Convergence



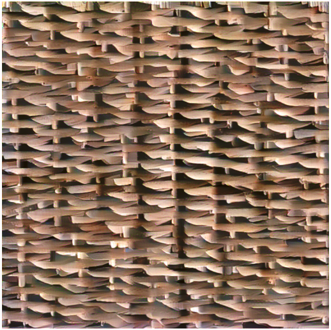
Iteration 500

Empirical Convergence



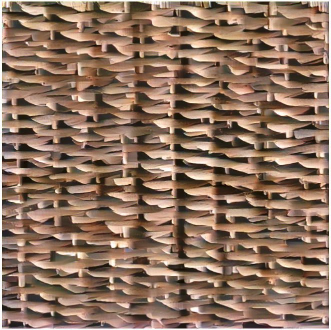
Iteration 600

Empirical Convergence



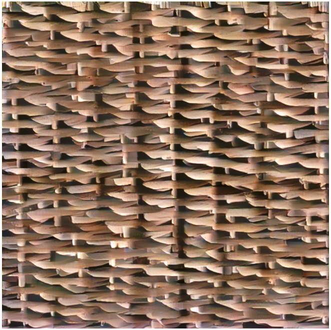
Iteration 700

Empirical Convergence



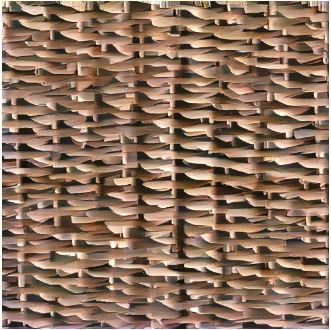
Iteration 800

Empirical Convergence



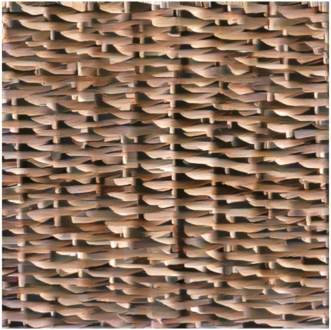
Iteration 900

Empirical Convergence



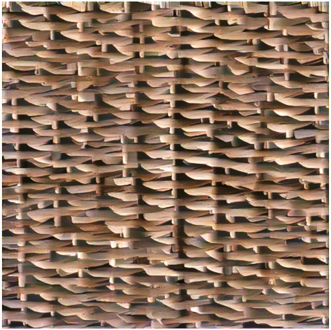
Iteration 1000

Empirical Convergence



Iteration 2000

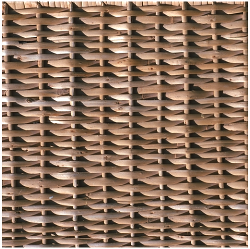
Empirical Convergence



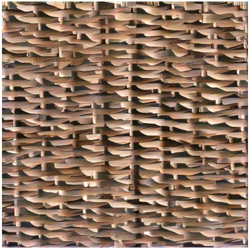
Iteration 4000

Synthesis Results

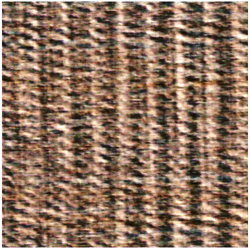
- Need to use pre-learn VGG-19 ?



Original (512×512)

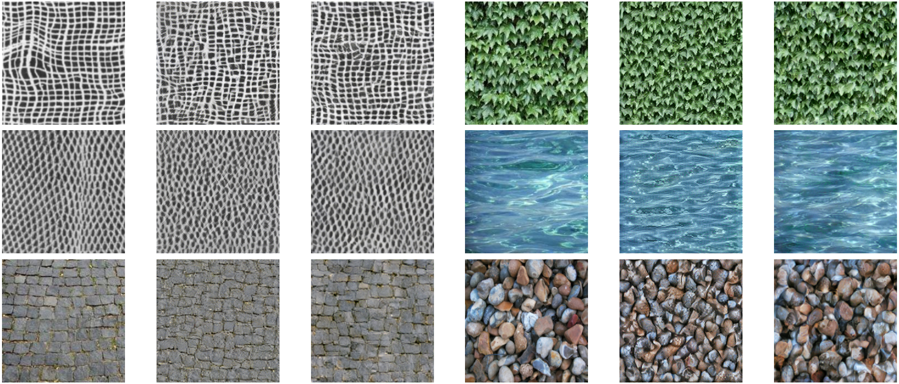


VGG weights



Random weights

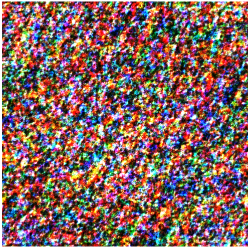
Comparison with DeepFrame



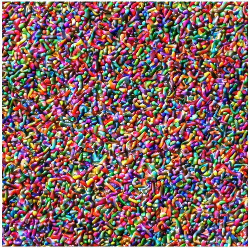
Original | DeepFrame **[Lu et al.]** | Our result | Original | DeepFrame **[Lu et al.]** | Our result

Synthesis Results

Original (512 $\times$ 512)



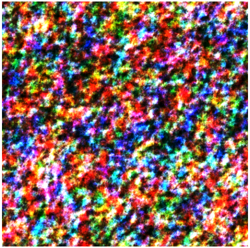
Initialization (Gaussian)



After 5000 iterations

Synthesis Results

Original (512 $\times$ 512)



Initialization (Gaussian)

After 5000 iterations

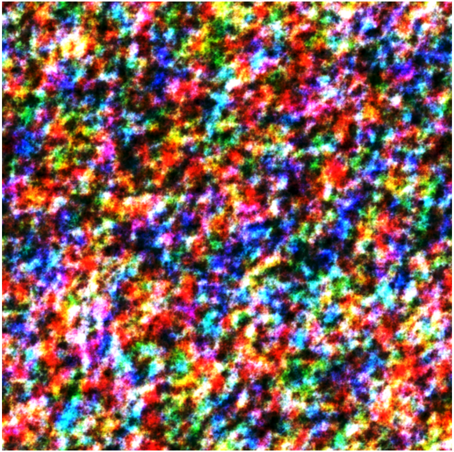
Synthesis Results: Mixing Issue

Original (512×512)

Synthesis Results: Mixing Issue

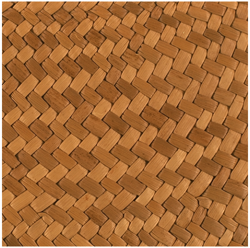
After 5000 iterations

Synthesis Results: Mixing Issue

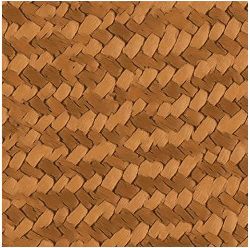


Initialization (Gaussian)

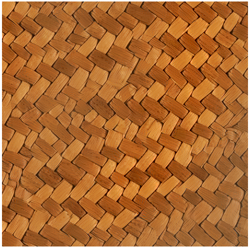
Comparison



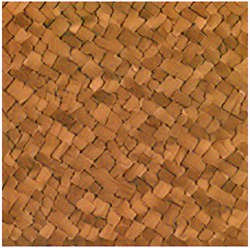
Original (512 $\times$ 512)



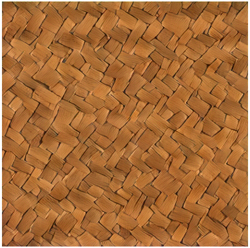
[Galerne et al.]



[Gatys et al.]



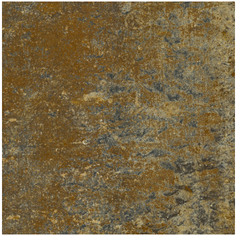
DeepFrame **[Lu et al.]**
(resolution /2)

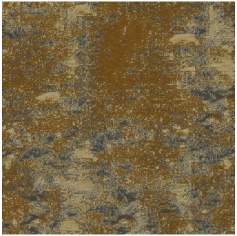


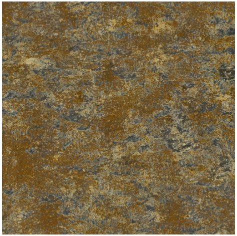
Our Result

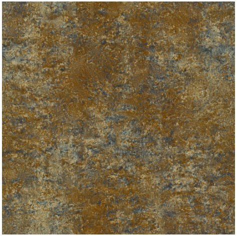
GAN **[Jetchev et al.]**

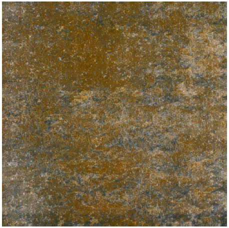
Comparison


Original


[Galerne et al.]


[Gatys et al.]


[Portilla & Simoncelli]


Our result


SGAN **[Jetchev et al.]**

Comparison

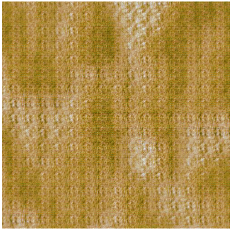
Original

[Galerne et al.]

[Gatys et al.]

[Portilla & Simoncelli]

Our result



SGAN [Jetchev et al.]

Conclusion - Perspectives

- Langevin sampling allows to design a generalization of FRAME
  $\longrightarrow$ with a continuous state-space
  $\longrightarrow$ only needs to differentiate the features (Auto-Diff)
- Provably convergent sampling and estimation algorithms (under hypotheses), even for CNN features !
- Able to synthesize textures using VGG features (although mixing time is large).
- A model with only 2560 parameters.
- Non mixing issue: When running with limited time, not mixing, so not that different from approximate microcanonical models/algorithms.

PERSPECTIVES/OPEN QUESTIONS:

- Microcanonical and macrocanonical models asymptotically coincide when $\Omega \to \mathbb{Z}^2$?
- Improve mixing time of Markov chain ?

Bibliographic references (not complete) I

- Macrocanonical Models for Texture Synthesis, V. De Bortoli, A. Desolneux, B. Galerne, A. Leclaire, SSVM 2019
- Maximum entropy methods for texture synthesis: theory and practice, V. De Bortoli, A. Desolneux, A. Durmus, B. Galerne, A. Leclaire, SIAM Journal on Mathematics of Data Science (SIMODS), 2021
- L. Gatys, A. S. Ecker, and M. Bethge, *Texture synthesis using convolutional neural networks*, in Advances in Neural Information Processing Systems, 2015
- D. J. Heeger and J. R. Bergen, *Pyramid-based texture analysis/synthesis*, SIGGRAPH ’95, 1995
- Y. Lu, S.-C. Zhu, and Y. N. Wu, *Learning frame models using CNN filters*, in 31th conference on artificial intelligence, 2016.
- D. Mumford and A. Desolneux, *Pattern Theory: The Stochastic Analysis of Real-World Signals*, Ak Peters Series, 2010
- J. Portilla and E. Simoncelli, *A parametric texture model based on joint statistics of complex wavelet coefficients*, IJCV, 40 (2000)

Bibliographic references (not complete) II

- K. Simonyan and A. Zisserman, *Very deep convolutional networks for large-scale image recognition*, tech report, 2014
- S. Zhu, Y. Wu, and D. Mumford, *Filters, random fields and maximum entropy (FRAME): Towards a unified theory for texture modeling*, International Journal of Computer Vision, 27 (1998)