GENERATIVE NETWORKS FOR EMULATING SYNTHETIC SKY IMAGES

Emulating astronomical images using Deep Generative models conditioned with Gaussian Processes or Autoregressive flows.

Claire Guilloteau
Université de Toulouse

Nesar Ramachandra
Argonne National Laboratory

François Lanusse
UC Berkeley

Sultan Hassan
New Mexico State University

Yuan-Sen Ting
Princeton University

Marc Huertas-Company
Observatoire de Paris & U. Paris Diderot

Sunny Cheng
University of Nottingham

Brant Robertson
University of California Santa Cruz

Alexie Leauthaud
University of California Santa Cruz

Song Huang
University of California Santa Cruz

September, 2019

ABSTRACT

Robust generation of high-fidelity data is an essential component of large astronomical survey analysis. While expensive simulations are typically used for creating synthetic data, recent development of fast emulators using machine learning techniques like Generative models – Variational Autoencoders[1], Generative Adversarial Networks[2] or Gaussian Processes – have made high precision predictions of cosmological functions possible. This projects deals with a natural progression of the above for emulating astronomical images using Deep Generative models. The project involves creating training images using GalSim, a software library for generating images of stars and galaxies, tuning a relatively few number of physical parameters on a space-filling scheme. A second dataset, made of real galaxy images from Cosmic Evolution Survey – COSMOS[3, 4] –, is also considered in this work. Ensemble of generative networks, Principal Component Analysis, Gaussian Processes, Variational Autoencoders and Masked Autoregressive Flow[5], is trained at interpolating values of input parameters. Independent measurement pipelines are applied to validate the emulated images beyond visual confirmation and hold-out tests. The network will be made more physics-aware based on this preliminary results and is extended to emulate realistic synthetic images tuned for other surveys and catalogs. Generative models designed in this work are compared at generating galaxy images and Deep generative models as Variational Autoencoders outperform machine learning techniques such as Principal Component Analysis for analytical and real images. An effort will be made in future work to improve current models performances and interpolate physical parameters to extend emulation to other surveys, as Illustris/TNG simulations.

***K*eywords** Galaxy emulation · Neural Networks · Generative models · Variational Autoencoder

1 Introduction

The generation of high-fidelity data is a crucial challenge in large astronomical analyses, either for creation of mock sky catalogs in anticipation of future surveys, or in parameter inference pipelines for confrontations with observation. While modeling the complex non-linear physics as gravitational or Magneto-hydrodynamical interactions can be very precise, they cannot be described analytically and generally require expensive numerical simulations. Application of these in traditional statistical inverse problems of astrophysical parameter inference such as the Markov Chain Monte Carlo (MCMC) Metropolis-Hastings algorithm[6] is prohibitively expensive since these Bayesian inference algorithms rely on hundreds of thousands forward model evaluations to estimate posterior probabilities of model parameters.

Recent developments of machine learning techniques enable model parameters inference from experimental data at low computational cost. Progress on emulation techniques based on deep generative models allow high-precision and inexpensive forward models evaluation. Emulation methods in astrophysics, for example [7, 8, 9], are presented as relevant to many problems when a large number of expensive model evaluations is needed and so are those presented in this work.

This project aims at creating a robust emulator based on generative models and neural networks to generate synthetic sky images, composed of galaxy postage stamps. Such emulator will be trained at generating images of galaxies from a number of physical parameters. Meticulous assessments and measurement pipelines will be performed to validate the emulated images beyond visual confirmation.

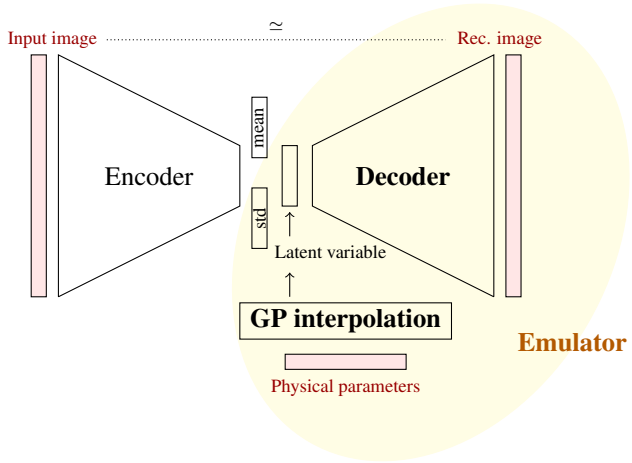

Figure 1: Emulation : an example with Variational Autoencoder and Gaussian Process

Given the complexity involved in real galaxy images, we first create a simulated dataset using GalSim [10], a library for generating images of stars and galaxies. This training set is made of elliptical exponential profile images generated by tuning relatively few parameters on an optimal space-filling sampling scheme [11]. However, deep learning techniques are known to be sensitive to input noise and real galaxy images are much more complex than analytical profiles. Therefore, an emulator trained on this first data wouldn't be able to generate high-fidelity data. Therefore, we define a second dataset made of real galaxy postage stamps from the Cosmic Evolution Survey (COSMOS) [3] taken by Hubble Space Telescope (HST) and related physical parameters given by GalSim and by Leauthaud et al. [4]. Several emulators are then trained and experienced on these datasets. Emulators in this work are designed with two components and trained in two steps, as illustrated Fig.1. First, an unsupervised generative model is trained to reproduce input images through dimensionality reduction and reconstruction (respectively denoted as encoder and decoder Fig. 1), learning a low-dimensional representation of the inputs, known as latent variables. Once trained, the dimensionality reduction network is removed and an interpolator (designated as Gaussian Processes – GP – interpolation Fig. 1) or a conditioning process is trained to map from physical parameters to latent variables. Finally, emulating synthetic galaxy images from physical parameters is presented here as mapping the latter to latent variables and decoding them to images. In this work, we propose three emulator architectures. The first one uses Principal Components Analysis (PCA) as generative model and GP as an interpolator. The second one, as shown in Fig. 1, is composed of a Variational AutoEncoder (VAE) [1] and GP interpolation. The third emulator architecture also consists of a VAE as a generative model but a conditional density estimation network (Masked Autoregressive Flow – MAF [5]) is used as a conditioning process. To assess these emulators performance, we consider several measurement and validation pipelines. We first evaluate reconstruction quality through first order metrics and compare pixel intensities between original and emulated images, compute reconstruction errors as Mean Square Error (MSE) and R-squared error (R^2). Then, we propose higher order metrics to confirm that the emulator is capable of realistic data synthesis. To do so, we measure, on original and emulated images independently, shear parameters using GalSim. The distribution of these parameters should be similar.

This report is organized as follows : in Sect. 2, we describe the 2 datasets used in this work. Emulation methods are detailed in Sect. 3 and results are evaluated in Sect. 4. Finally, perspectives and future work are discussed and conclusions of this work are presented in Sect. 5.

2 Data

2.1 GalSim images

Galaxy images emulated in this work are generated using GalSim [10]. This open-source toolkit aims at providing images of galaxies and stars, including several transformations handling such as dilation, rotation, shear or convolution with Point spread function (PSF) profiles. GalSim allows one to generate and transform a number of parametric profiles or images of galaxies from Cosmic Evolution Survey (COSMOS) [3], a real astrophysical catalog, associated to a PSF model and a parametric representation.

2.1.1 Analytical profiles

Given some well-chosen physical parameters, GalSim is able to generate galaxy profile images. Therefore, as a first simple dataset, we define five physical parameters to tune for galaxy images generation with GalSim. We initially generate a set of parametric exponential profiles. The surface brightness varies along with r as

$$I(r) = \frac{F}{2\pi r_0^2} \exp -\frac{r}{r_0},$$

with respect to F, total flux and r_0, scale radius. Such generated profiles are circularly symmetric and centered on the coordinate origin. Then we introduce two additional parameters describing a shear distortion to transform a circle into an ellipse with a minor-to-major axis ratio $q = \frac{b}{a}$ and a position angle β. In practice, we use as parameters $g_1 = \frac{a-b}{a+b} \cos 2\beta$ and $g_2 = \frac{a-b}{a+b} \sin 2\beta$, under the constraint $\sqrt{g_1^2 + g_2^2} < 1$, to define this transformation. The g_1 and g_2 are called "reduced shear" parameters. Finally, these elliptical exponential profiles are convolved with a Gaussian PSF, characterized by its Full Width at Half Maximum (FWHM), linearly related to the variance σ^2 of the related 2-D Gaussian distribution. The values of these 5 physical parameters (F, r_0, g_1, g_2, σ^2) are generated on a latin hypercube design for optimal space filling. This space-filling scheme is explained in detail in Appendix A.

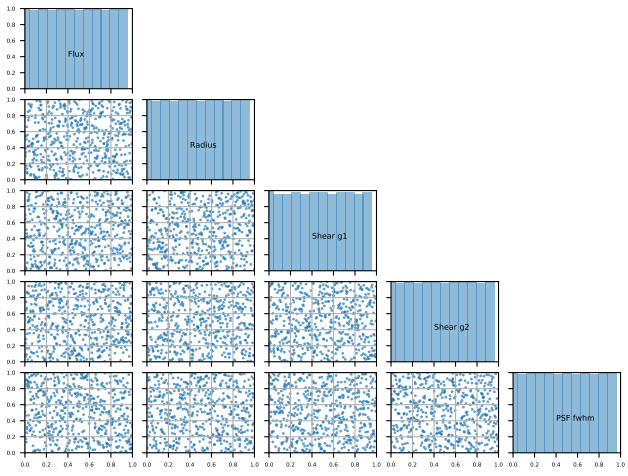

Figure 2: Optimal space filling scheme: latin hypercube for 512 evaluations.

A realization of latin hypercube sampling performed for 512 sampled points is presented Fig. 2. Each graph in the diagonal plots the distribution histogram of each parameter, which is, as expected, uniformly distributed. Each scatter plot represents the normalized value of 2 physical parameters associated to each sampled point. Using these physical parameters samples, we generate with GalSim a representative analytical profiles dataset. Fig. 3 shows a subset of these parametric galaxy images and emphasizes the large diversity of generated elliptical profiles.

2.1.2 COSMOS-type images

The COSMOS survey is the largest survey taken by the Hubble Space Telescope (HST). A publicly available subset of this survey, the COSMOS HST Advanced Camera for Survey (ACS) is used by GalSim as postage stamps provided by [add ref Leauthaud et al. 2007] with a PSF model based on the optical properties of HST. The catalog used here is a catalog of $\sim$ 82000 galaxies taken by the Wide Field Channel (WFC),

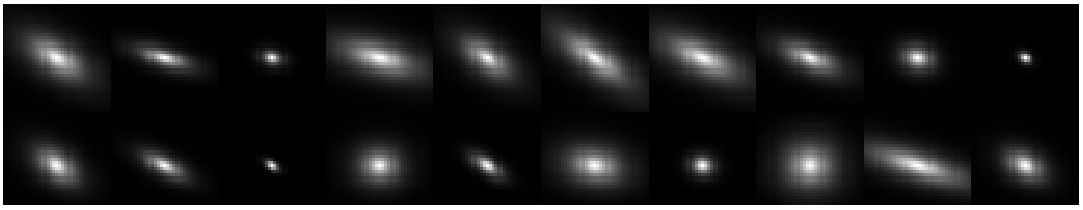

Figure 3: 20 GalSim profiles generated with 5 parameters: galaxy flux, radius, shear profile parameters g_1 and g_2, Gaussian PSF fwhm.

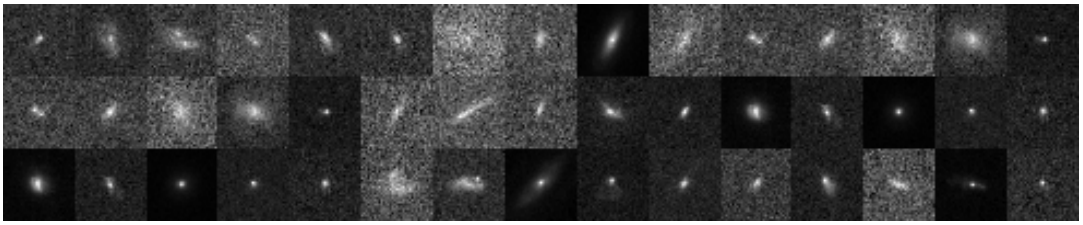

Figure 4: 45 postage stamps (64 $\times$ 64 pixels) of real galaxy images from COSMOS with an absolute depth magnitude lower than $I_{AB} = 25.2$, provided by GalSim.

through the F814W filter with an absolute depth magnitude lower than $I_{AB} = 25.2$. fig. 4 shows 45 postage stamps of size 64 $\times$ 64 pixels of such real galaxies in COSMOS provided by GalSim. The noisy images of the galaxies depart significantly from the from analytical profiles from Fig. 3, due to the reduction in brightness and features with decreased signal-to-noise ratio. This underlines the fact that analytical profiles are highly over-simplified to understand their complexities of real galaxies.

Every postage stamp in GalSim is associated with a parametric representation. Galaxies have been fitted either with a sersic profile or a bulge and disk profile. The surface brightness of a sersic profile varies as

$$I(r) = \frac{F}{a(n) r_e^2} \exp -b(n) \left(r/r_e\right)^{\frac{1}{n}},$$

where r_e is the half-light radius (the radius within half of the galaxy light is contained), $a(n)$ is a normalization function and $b(n)$ computed by GalSim following [add ref Ciotti et al. 1999] with respect to n, the sersic index. The bulge profile is fitted with a DeVaucouleurs profile whose surface brightness is

$$I(r) = \frac{F}{0.010584 r_e^2} \exp -7.66925 \left(r/r_e\right)^{\frac{1}{4}},$$

and the disk profile is finally fitted with an exponential profile. Similar to elliptical exponential profiles generation (Sect. 2.1.1), the two shear parameters $q = \frac{b}{a}$ and β are added to describe each galaxy. GalSim also provides an estimate of the redshift z of each galaxy. Hence, these 6 parameters (F, r_e, n, q, β, z) represent the physical parameters provided by GalSim. We also include some additional parameters introduced in [add ref Leauthaud et al. 07] : absolute magnitude I_{AB} in the AB system, star formation rate SFR, specific star formation rate $sSFR$ and stellar mass M.

In this set of real galaxy images, the distribution of each parameter is not uniform as in analytical profiles generated in the previous section with an optimal space-filling scheme. Fig. 5 presents the distribution histogram of 5 physical parameters associated to each image (in the diagonal) and scatter plots of normalized values of 2 physical parameters. These plots give an overview of galaxies in our COSMOS-type dataset. The distribution of these 5 physical parameters looks like Gamma or Normal distributions where normalized extreme values (near 0 or 1) are under-represented in the dataset. This might have an influence on the performance of our generative model, which will not be trained enough to generate outliers in the training set, as instance galaxies with a high or medium half-light radius and a high flux.

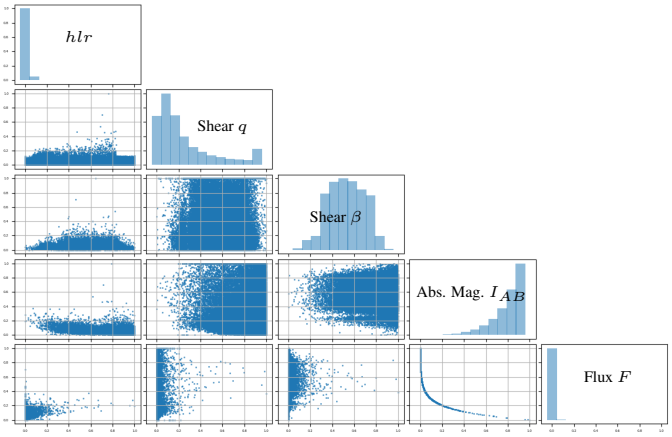

Figure 5: An overview of a few COSMOS galaxy physical parameters given by GalSim et Leauthaud et al. [4] : half-light radius, shear parameters q and β, absolute magnitude and galaxy flux. In the diagonal: the distribution histogram of physical parameters associated to each image. Other plots : scatter plots of normalized values of 2 physical parameters associated to each image.

3 Emulation methods

This section develops different emulation methods build up with a generator (Principal Components Analysis - PCA or Variational AutoEncoder - VAE) and an interpolating (Gaussian Processes - GP) or a conditioning (Masked Autoregressive Flow - MAF) process, their main characteristics and architecture. These emulators are trained and tested on the various datasets we introduced in Sect. 2. Visual results on each dataset and for each method are shown and discussed.

3.1 Emulation with PCA and GP

PCA is known as an orthogonal and linear dimensionality reduction procedure to transform a set of N observations of dimension p expected to be correlated into a smaller set of linearly independent M principal components of dimension p. This process performs a M-dimensional ellipsoid fitting to the data. Th principal components are then the axes of the ellipsoid. More formally, a PCA allows to write each observation $\mathbf{X} \in \mathbb{R}^p$ as

$$\mathbf{X} \simeq \mathbf{B}\mathbf{z}, \tag{1}$$

with $\mathbf{B} \in \mathbb{R}^{p \times M}$ is the orthogonal truncated basis (i.e. the principal components basis) and $\mathbf{z} \in \mathbb{R}^M$ are the weights associated to each principal component. The encoding part of the PCA generator is finding the best ellipsoid fit and thus the principal components matrix $\mathbf{B}$. In practice, the number of principal components M is an hyperparameter and is often set as the smallest number of components that statistically explain at least a certain percentage (99.5 % in Fig. 6.**a**)) of the variance of the dataset. The decoding part of the PCA generator is basically multiplying this basis matrix $\mathbf{B}$ with a latent vector $\mathbf{z}$. As PCA relies on linear assumptions, if the data is not linearly correlated, or if data doesn't fit in an ellipsoid manifold, PCA could fail to find a relevant basis of principal components. Therefore, we expect PCA to be a powerful generator on the simple analytical profiles dataset and quickly be over-performed by other generators for complex COSMOS images.

Once a dimensionality reduction has been performed on the training set and an efficient generator model such as PCA has been trained, we aim at linking the related physical parameters to the low-dimensionality representation of the data, i.e. the latent space represented in Eq. 1 by $\mathbf{z}$. To this aim, we need to observe and ensure a dependency between the observed physical parameters of the training set and the related latent variables. This can be achieved by simply plotting point-wise dependency graphs between 2 dimensions of the latent space and assigning to each point a color corresponding to the value of a specific physical parameter. Such a graph is given Fig. 7 and will be discussed later. If any dependency is noticed, mapping physical parameters $\mathbf{p}_i \in \mathbb{R}^d$ into a latent variable $\mathbf{z}_i \in \mathbb{R}^M$ can be performed with a Gaussian Process (GP) interpolation. GP interpolation process is detailed in Appendix B.

As shown in Fig. 6, we performed a PCA on 512 analytical profiles generated with GalSim in Sect 2.1.1, setting the number M of components to 12 such that the model explains $\sim 99.5\%$ of the variance of the data. The resultant 12 components are shown in Fig. 6.**b)**. Fig 7 shows an example of a point-wise dependency plot of the first 2 dimensions of the latent space, i.e. the values of the two first principal components, where a light yellow point corresponds to a large-radius galaxy and a dark blue point to a small-radius galaxy. There is here a very clear dependency between the physical parameter r_0 and the value of the first two components. Similar plots can be derived to ensure that dependencies are observed with every physical parameter and that an interpolation between physical parameters and latent variables is possible. A Gaussian Process is then trained to perform such interpolation and the emulator finally the system {GP, inverse PCA}, with GP interpolating from physical parameters to a latent variable and inverse PCA generating a galaxy image from the interpolated latent variable. Results of this emulator on 10 test-analytical profiles (which are not included in the training set) are shown in Fig. 8. The top row shows the original analytical profiles whereas the bottom row shows the emulated image from the physical parameters associated to each profile in the first row. These results confirm that, visually, our emulator is able to generate realistic simple profiles from physical parameters. However, very sheared galaxies aren't visually well emulated as an artifact appears in the emulated images such that the emulated profile doesn't look like an exponential profile anymore. The last column of Fig. 8 is an example of this phenomenon. Such artefacts occur during image generation by inverse PCA, we train another generative model on this dataset in Sect. 3.2 to fix this issue.

The same PCA pipeline is then applied on the COSMOS-type dataset. The number of components is set to $M = 20$ to explain 97.7% of the variance of the data. The threshold of explained variance ratio is reduced here because a small part of the total variance of the data is related to the noise in training set. We aim here at recovering only the principal components only related to the galaxies in our dataset. Before controlling dependency between physical parameters and latent variable, we need to verify that is able to reproduce more complex images than analytical profiles, Fig. 9. Similarly the first row shows the original COSMOS images. The second row plots the reconstructed image after basis truncation and last row images are noisy versions of second row images, to confirm that reconstructed images with Gaussian noise are visually similar to original COSMOS images. PCA seems to be able to generate realistic galaxies when they are close to simple profiles but fails to reproduce more sophisticated profiles as galaxies flagged with a red star Fig. 9. Moreover, dependencies between physical parameters and latent variable are more difficult to notice in point-wise dependency plots, as illustrated in Fig. 10. Training a Gaussian Process to interpolate physical parameters in such a latent space isn't relevant and emulating COSMOS images with a {GP, inverse PCA} system fails to capture more complex structures. We next explore other methods that might overcome PCA-GP limitation.

3.2 Emulation with VAE and GP

In this section, we explore another generative model, a Variational Autoencoder, associated with Gaussian Processes to emulate more realistic analytical profiles and COSMOS images. VAE and GP emulator is expected to fix or improve PCA issues highlighted in the previous section. A Variational AutoEncoder (VAE) is an unsupervised generative model designed to, given a training data, generate new samples from the same distribution. VAEs are a subset

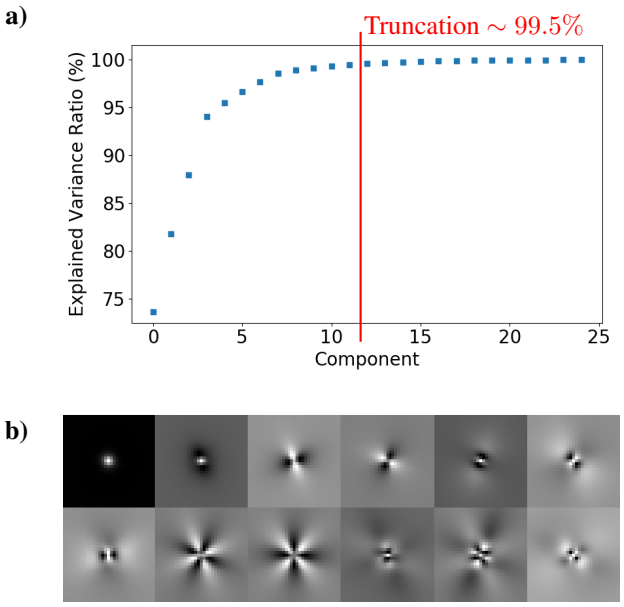

Figure 6: **a)** Explained variance ratio (%) versus number of components in truncated basis for a PCA performed on 512 analytical profiles generated with GalSim in Sect. 2.1.1. The truncation (in red) is performed such that linear combinations of the first principal components statistically explain $\sim 99.5\%$ of the variance of the data. **b)** The 12 first components (from highest to lowest individual explained variance ratio) composing the truncated basis $\mathbf{B}$ in Eq. 1.

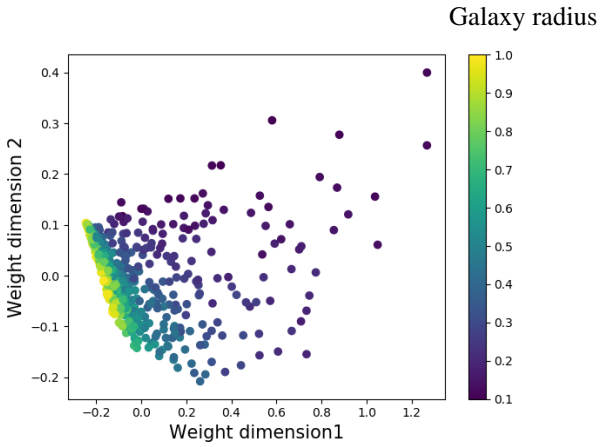

Figure 7: Point-wise dependency scatter plot for the first 2 dimensions of the latent vector $\mathbf{z}$ in Eq. 1 calculated on the 512 analytical profiles dataset generated with GalSim in Sect. 2.1.1. A color is assigned to each point according to the radius of the related simulated galaxy.

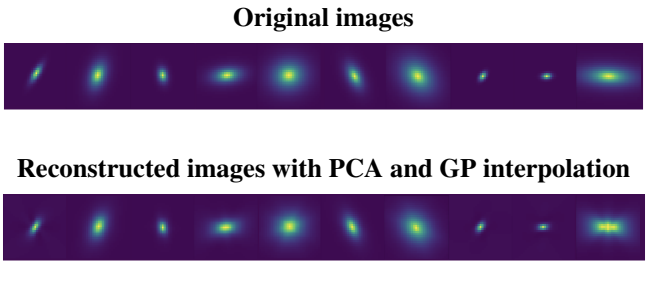

Figure 8: Visualisation of PCA-GP emulation of analytical profile images. Top row: Original COSMOS images to be reconstructed. Bottom row: Reconstructed COSMOS images with PCA and GP interpolation.

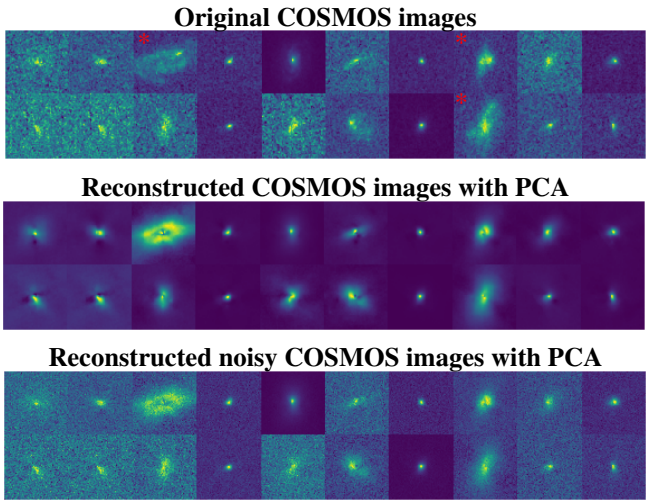

Figure 9: Visualisation of PCA emulation of COSMOS images. Top row: Original COSMOS images to be reconstructed. Middle row: Reconstructed COSMOS images with PCA. Bottom row: Reconstructed COSMOS images with PCA and Gaussian noise added.

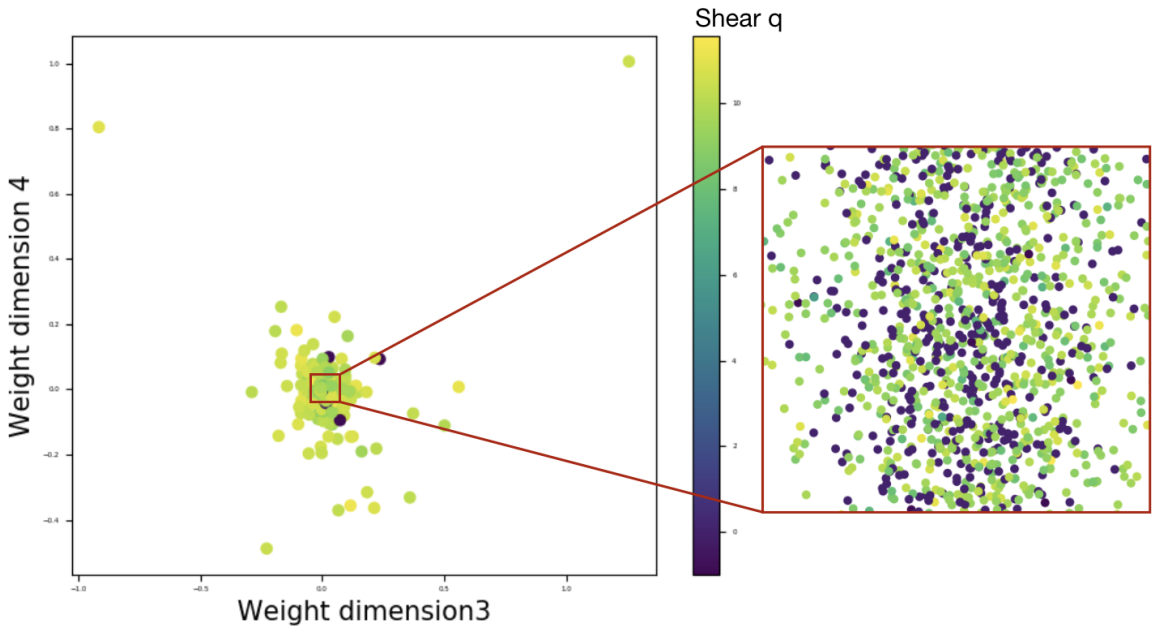

Figure 10: Point-wise dependency scatter plot for the first 2 dimensions of the latent vector $\mathbf{z}$ in Eq. 1 calculated on the COSMOS-type dataset presented in Sect. 2.1.2. A color is assigned to each point according to the shear parameters q of the related galaxy.

of generative models with approximate (but explicit) density estimation. Given a training data $\mathbf{x}$, they define an untractable density function with latent variable $\mathbf{z}$, $\mathrm{p}_\theta(\mathbf{x}) = \int \mathrm{p}_\theta(\mathbf{z})\mathrm{p}_\theta(\mathbf{x}|\mathbf{z})$. This density function cannot be optimized toward the network parameters θ directly. In practice, it is derived and a lower bound of the likelihood is optimized.

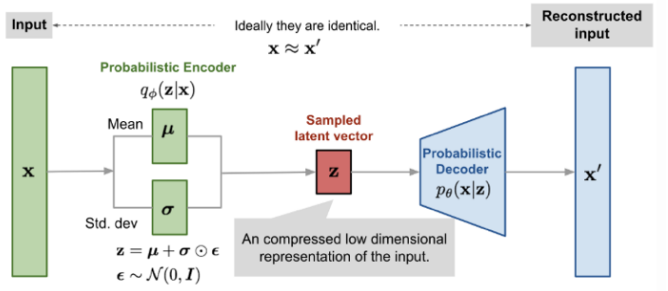

Figure 11: Probabilistic model of Variational Autoencoders: illustration.

VAEs, unlike Autoencoders, allow to sample from the model to generate data. Following the general architecture of a VAE detailed in Fig. 11, the generative model named here as probabilistic decoder, should be able from a sampled latent vector $\mathbf{z}$. This latent variable is a low-dimensional representation capturing the main features of our dataset, to sample from a true conditional $\mathrm{p}_\theta(\mathbf{x}|\mathbf{z})$ a realistic generated output $\mathbf{x}'$. That latent vector $\mathbf{z}$ is sampled from a true prior $\mathrm{p}_\theta(\mathbf{z})$ usually chosen Gaussian such that $\mathbf{z} \sim \mathcal{N}(\mu, \mathbf{\Sigma})$, where $\mathbf{\Sigma}$ is a diagonal matrix of diagonal σ. The training aims at estimating the true parameters θ of the model. The intractability of the likelihood and the posterior density leads to introduce an encoder network which takes as input $\mathbf{x}$ and estimates a posterior $\mathrm{q}_\phi(\mathbf{z}|\mathbf{x})$, with ϕ parameters of the encoder, which approximates the posterior $\mathrm{p}_\theta(\mathbf{z}|\mathbf{x})$. Both encoder and decoder produce distributions over $\mathbf{z}$ and $\mathbf{x}$ and sample from this distribution to obtain $\mathbf{z}$ and $\mathbf{x}'$. The loss function $l(\phi, \theta)$ detailed in Eq. 2 used to optimize parameters ϕ and θ is a lower bound of the log-likelihood $\log \mathrm{p}_\theta(\mathbf{x})$.

$$l(\phi, \theta) = -\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right] + \mathbb{KL}\left[q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\right] \quad (2)$$

The training is thus maximizing this lower bound composed of two terms : the negative KL-divergence between the posterior $\mathrm{q}_\theta(\mathbf{z}|\mathbf{x})$ and the prior $\mathrm{p}_\theta(\mathbf{z})$ and the log-likelihood of $\mathrm{p}_\theta(\mathbf{x}|\mathbf{z})$. This means that the training is optimizing toward ϕ and θ in order to make approximate posterior close to prior and reconstruct input data.

In practice, encoder and decoder parts of VAEs are designed as neural networks with different kind and number of layers depending on the task. Our VAE consists of convolutional layers (4×4 kernels and a ReLU activation function), that are expected to efficiently extract features from images. The general architecture of VAEs we work with in this work is detailed in Fig. 11. As the architecture of a VAE can rely on non-linear transformations, it is more likely to extract complex features from the data than PCA and thus provides a non-linear low dimensional representation of the data.

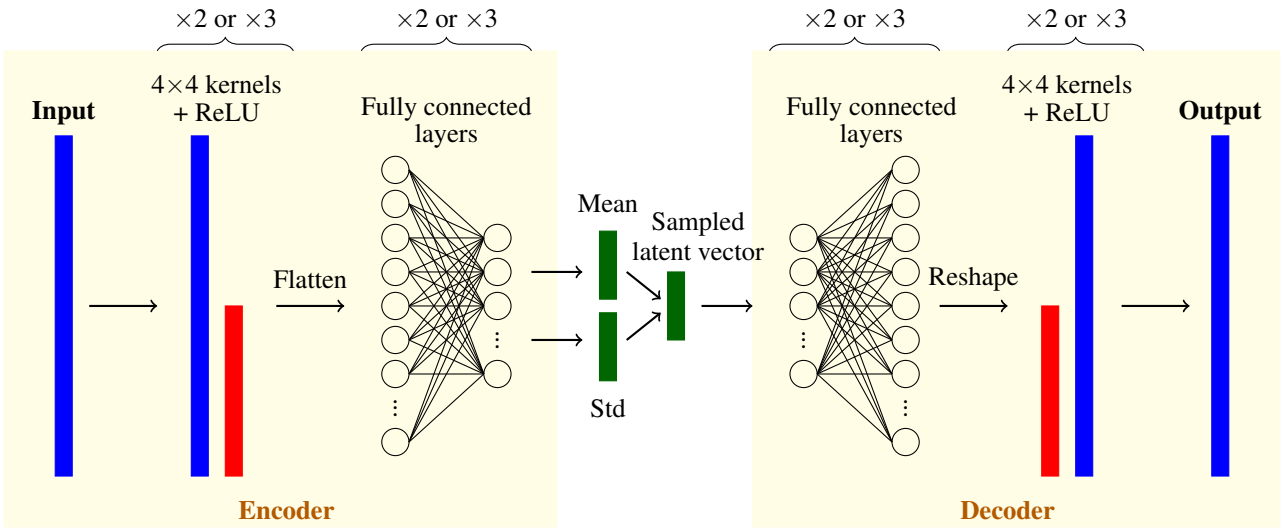

Figure 12: Overall architecture of Variational Autoencoders used as generative models in this work, either trained with analytical profile images or COSMOS images.

Once the network is well trained to reproduce each input data, the encoder part is no longer used and the proper generative model is the probabilistic decoder. As in Sect. 3.1, if possible, a GP is trained to interpolate a set physical parameters (related to training images) to the latent space established by the training of a VAE. New galaxy images are then emulated from physical parameters to a latent variable by GP interpolation and from a latent variable to a realistic image by VAE-decoding.

We apply the above pipeline to a training set composed of 512 analytical profiles generated in Sect. 2.1.1 and we create a 64 images testing set from physical parameters sampled with a latin hypercube scheme. We set the dimension of the latent space to 20, a trade-off between the number of components used in Sect. 3.1 and the need of a low-dimensional latent space. The VAE training converges very fast after about a hundred epochs. Before any GP training, Fig. 13 illustrates a dependency between physical parameters and latent variables through a point-wise dependency plot of the first two dimensions of the latent space, where a color related to the galaxy has been assigned to each point. The first colorbar matches galaxies from the testing set and the second one galaxies from the training set.

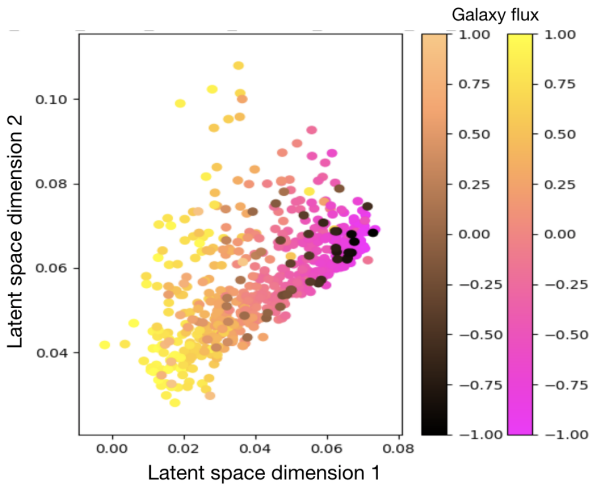

Figure 13: Point-wise dependency scatter plot for the first 2 dimensions of the latent vector $\mathbf{z}$ calculated on the analytical profiles dataset presented in Sect. 2.1.1. A color (copper colorscale for testing set images and yellow colorscale for training set images) is assigned to each point according to the flux of the related simulated galaxy.

As mentioned earlier in Fig. 7, a clear dependency is observed between fluxes of galaxies and the first two dimensions of the low-dimensional representation of the data established by VAE training. Such dependency is similarly noticed for other physical parameters associated to our dataset. After GP training, we visually control Fig. 14 that our emulator, composed of GP interpolation and VAE decoding, is able to reproduce, from the same physical parameters, similar profiles (2^{nd} row) than GalSim simulation (1^{st} row). Almost no difference can be detected between simulations and emulations, even for very sheared galaxies (5^{th} and 7^{th} images) that are badly reconstructed with PCA and GP emulation. The major visual issue of emulating analytical profiles with PCA and GP is here solved by replacing the generative model by a VAE. Further validation tests are performed in Sect. 4.

Next, a similar VAE is trained on reproducing a subset of COSMOS postage stamps introduced in Sect. 2.1.2. The training is first performed on ~ 20000 COSMOS-type images dataset to reduce computational time but still assess our emulator performances on a reasonably trained model. A training on the whole dataset will be performed later. As COSMOS images are more complex images than analytical profiles, we suppose there are more features to extract from this dataset and we set the latent space dimension to 32. VAE training converges after ~ 4000 epochs. Before connecting the interpolator, we assess

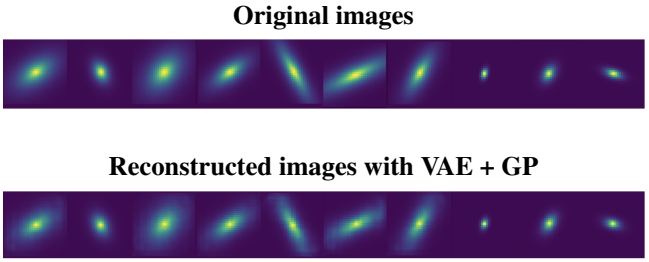

Figure 14: Visualisation of VAE-GP emulation of analytical profile images. Top row: Original images to be reconstructed. Bottom row: Emulated images with VAE-GP.

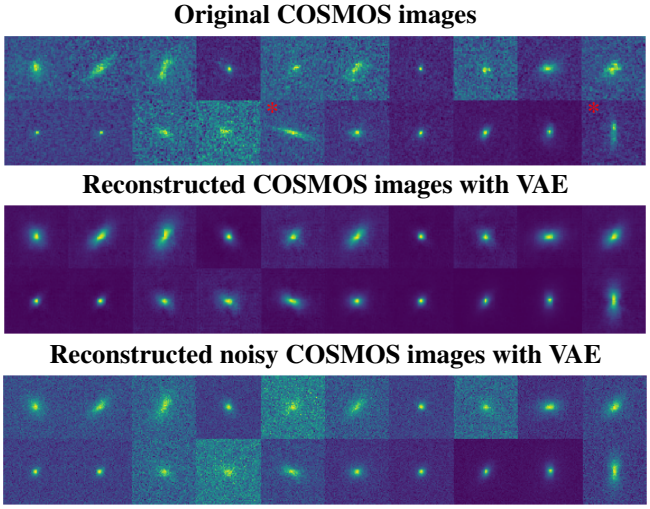

Figure 15: Visualisation of VAE generation of COSMOS images. Top row: Original COSMOS images to be reconstructed. Middle row: Reconstructed COSMOS images with VAE trained on COSMOS images. Bottom row: Reconstructed COSMOS images with VAE and Gaussian noise added.

the viability of the VAE in reproducing galaxies as shown in Fig. 15. The first row shows the original COSMOS images, the second row the reconstructed images with VAE and last row images are noisy versions of second row images. Third row galaxies should be visually close to original COSMOS images. Given the higher complexity of these COSMOS-type galaxy images generating problem than analytical profiles generation, we cannot expect as good results as those shown Fig. 14 mainly due to noise corruption and large heterogeneity of profiles in COSMOS images. Galaxies seem to be well reproduced by the encoding and decoding process, expect for very flat galaxies (flagged galaxies in Fig. 15), that are reconstructed shrunk but thicker. This may be because of under-representation of very flat galaxies in the dataset. This problem is not solved yet but improvements to deal with under-representation are proposed Sect. 5. Then, we try to visualize some dependency between physical parameters and latent variables of training and testing sets Fig. 16.**a)**, through point-wise dependency plots of 2 dimensions of the latent space where each point in gray-scale is related to a galaxy image in the training set and each point in orange-scale to a galaxy image in the testing set. Each point varies in gray or orange level according to the half-light radius of the matching galaxy. In this plot in particular and in the majority of point-wise dependency plots, dependency between physical parameters and latent variable is difficult to observe. Therefore, we use UMAP (Uniform Manifold Approximation and Projection), a non-linear dimensionality reduction method for visualization based on geometry and topology. In this case, the idea is to project the dimension of the latent space to a 2-dimension space to visualize this projection and the projected value of latent variables. This can be a better visual indicator of the dependency between latent variables and physical parameters than point-wise dependency plots, that doesn't take into account every dimension of the latent space. Fig. 16.**b)** shows the UMAP plot of the latent space related to the VAE trained with COSMOS images, with colors corresponding to galaxy half-light radius. A dependency is indeed clearly noticed in this plot, especially for high half-light radius (hlr) galaxies. However, the difference between very low and low hlr galaxies, in the bottom-right part of the plot, is hardly distinguishable. As a matter of fact, no GP we trained was able to effectively interpolate from physical parameters to this latent space. The reason could be that the conditional distribution of latent variables given physical parameters is not as simple as a Gaussian and a more complex interpolator or conditional density estimator is needed here.

3.3 Emulation with VAE and MAF

This section explores the option of considering a density estimator neural network instead of a Gaussian Process interpolator to map from physical parameters to latent variables of the VAE trained on COSMOS images previously. Masked Autoregressive Flows (MAFs) provide an estimation of a joint density $\mathrm{p}(\mathbf{x})$ of a set of variables $\mathbf{x}$ based on a joint interpretation of autoregressive models, a neural-network like function that is invertible, and normalizing flows, that map the distribution to a normal distribution from which we can easily sample. This joint density $\mathrm{p}(\mathbf{x})$ can be decomposed into a product of 1-dimensional conditionals, parametrized in autoregressive models as Gaussian such that :

$$\mathrm{p}(\mathbf{x}) = \Pi_i \mathrm{p}(x_i|x_{1:i-1}), \text{ where } \mathrm{p}(x_i|x_{1:i-1}) = \mathcal{N}(x_i|\mu_i, (\exp \alpha_i)^2)$$

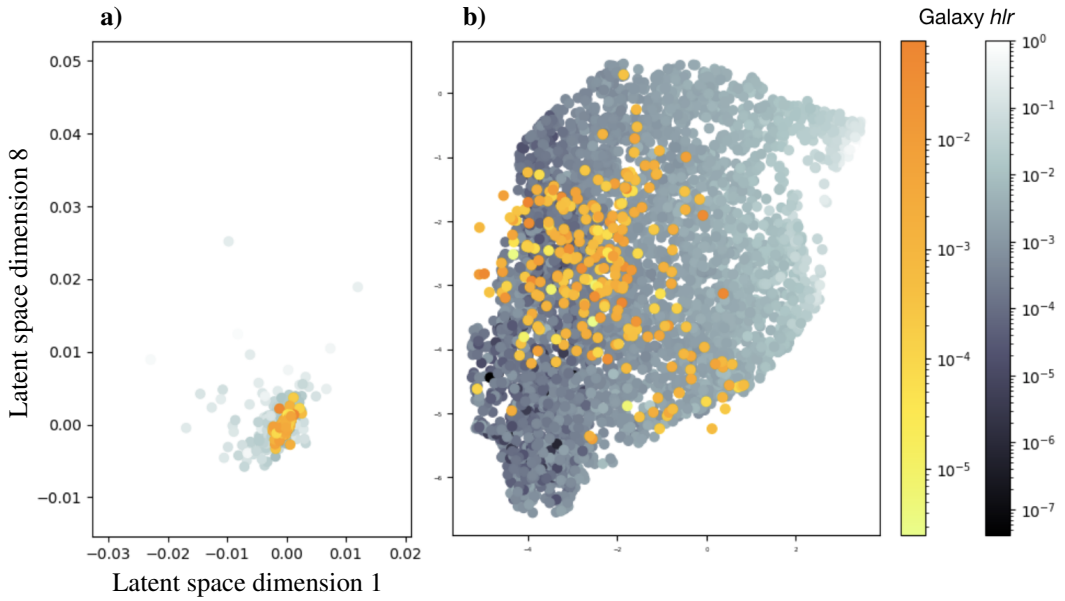

Figure 16: **a)** Point-wise dependency scatter plot for 2 dimensions of the latent vector $\mathbf{z}$ calculated on the COSMOS-type dataset presented in Sect. 2.1.2. A color (orange colorscale for testing set images and gray colorscale for training set) is assigned to each point according to the half-light radius of the related galaxy. **b)** Uniform Manifold Approximation and Projection plot of latent variables over training and testing sets. Colorbars stand for both plots.

$$\text{with} \quad \begin{array}{rcl} \mu_i & = & f_{\mu_i}(x_{1:i-1}) \\ \alpha_i & = & f_{\alpha_i}(x_{1:i-1}). \end{array}$$

This allows to write this data generation equation :

$$x_i = f(u_i), \text{ with } \mathbf{u}_i \sim \mathcal{N}(0, \mathbf{I}) \text{ and } f(u_i) = u_i \exp \alpha_i + \mu_i.$$

f is here easily invertible such that :

$$u_i = f^{-1}(x_i) \text{ where } f^{-1}(x_i) = (x_i - \mu_i) \exp -\alpha_i.$$

Thus, the determinant of the Jacobian of f^{-1} is easily computable : $\left|\det \frac{\partial f^{-1}}{\partial \mathbf{x}}\right| = \exp - \sum_i \alpha_i$, which allows this autoregressive model to be seen as a normalizing flow where $\mathrm{p}(\mathbf{x}) = \pi\left(f^{-1}(\mathbf{x})\right)\left|\det \frac{\partial f^{-1}}{\partial \mathbf{x}}\right|$ where $\pi(\cdot) \sim \mathcal{N}(0, \mathbf{I})$. The function f can be interpreted as an invertible and differentiable transformation of a standard normal density into the target density. This flow is implemented by stacking several Masked Autoencoders for Density Estimation (MADEs). MADE is an adaptation of Autoencoders to make them estimate a tractable distribution in a single pass through the network as shown in Fig. 17.

An alternative of MAFs used in this work is conditional MAFs which task is to estimate the conditional joint density $\mathrm{p}(\mathbf{x}|\mathbf{y}) = \prod_i \mathrm{p}(x_i|x_{1:i-1}, \mathbf{y})$ given a set of pairs $(\mathbf{x}, \mathbf{y})$. This is done by extending the set of input variables with $\mathbf{y}$ and only modeling conditional densities. $\mathbf{y}$ is then an additional input in each MADE layer of this conditional MAF and $\mathrm{p}(\mathbf{x}|\mathbf{y}) = \pi\left(f^{-1}(\mathbf{x}, \mathbf{y})\right)\left|\det \frac{\partial f^{-1}}{\partial \mathbf{x}}\right|$. In our physical parameters $\mathbf{p}$ to latent variables $\mathbf{z}$ mapping problem, the idea is to sample a new latent variable $\mathbf{z}$ from the estimated distribution $\mathrm{p}(\mathbf{z}|\mathbf{p})$ using a conditional MAF as described above taking $\mathbf{x}$ as a latent variable $\mathbf{z}$ and $\mathbf{y}$ as physical parameters $\mathbf{p}$. So far, unfortunately, this conditional density estimation with MAFs does not give acceptable results and this work is still in progress. Considering this generation problem and dependencies observed in the previous section between physical

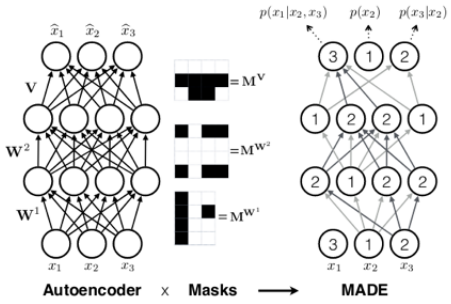

Figure 17: Masked Autoencoder for Density Estimation : illustration. The final density is calculated as follows: $\mathrm{p}(\mathbf{x}) = \mathrm{p}(x_2)\mathrm{p}(x_3|x_2)\mathrm{p}(x_1|x_2, x_3)$

parameters and latent variables, well designed and trained MAFs should, eventually, be able to estimate a relevant conditional distribution of latent variables given physical parameters.

4 Validation schemes

This section develops validation schemes we perform on emulation methods introduced previously to analyze and compare their performance. We first evaluate these methods with first order validation tests, which essentially are pixel-to-pixel comparisons. Then, we assess performances on higher order metrics, designed to confirm that emulator are truly generating realistic galaxy images that can be used in survey analysis. Validation tests are applied, in this section, to the emulators PCA and GP, and VAE and GP trained on analytical profiles. Given visual performances of emulators trained on COSMOS images and the difficulty of mapping physical parameters to latent variables we are currently confronting, we only discuss performances of the VAE model trained on COSMOS images. Nevertheless, each validation pipeline shall apply on every emulator which generate realistic galaxy images.

4.1 First order validation

We first evaluate emulators performances on their ability to generate images for which pixel values are close to simulated or real COSMOS images. To do so, we first plot the histogram distribution of true and predicted pixel intensities. Ideally, these two distributions are equal. We also plot predicted pixel intensities versus true pixel intensities, showing good one-to-one correlation. Figs. 18 and 19 show these plots respectively for PCA-GP and VAE-GP emulation, both trained on the analytical profiles dataset described in Sect. 2.1.1. According to these plots, the distribution of PCA-GP reconstructed pixel intensities is closer to the true distribution than the distribution of VAE-GP reconstructed pixel intensities. Similarly, the scatter plot of predicted versus true pixel intensity related to PCA-GP emulation is closer to identity than VAE-GP emulation scatter plot. This result is expected given the relatively small size of the training set (512 images). We experienced, in this work, that VAE-GP emulator performance on this dataset increases with training set size, outperforming very fast PCA-GP emulator performance. Fig. 22 shows both plots for VAE image generation, trained on $\sim$ 20000 COSMOS-type images, described in Sect. 2.1.2. Predicted pixel intensity plotted here is non-noisy generated images pixel intensities. Therefore, predicted pixel intensity distribution and predicted versus true pixel intensity scatter plot are not expected to be as close as, respectively, true pixel intensity distribution and identity. In particular, the scatter plot is expected to be spread around identity with a deviation corresponding to the standard deviation of the noise in training images, without any bias. Fig. 22 shows that distributions of predicted and true pixel intensities are very close to each other and that the scatter plot of predicted versus true pixel intensity is spread around identity with a small bias for very small intensities, which might be due to bad reconstruction of low-flux flat galaxies.

Then, we assess emulators performance with first order metrics. These first order metrics are pixel-wise quantitative assessments that measure a reconstruction error based on differences between true and reconstructed pixel values. Let us denote by $\mathbf{x}$ original images to be reconstructed and $\mathbf{x}'$ emulated images, both composed of n pixels, where $\mathbf{x}_i$ stands for the i^{th} pixel of the image $\mathbf{x}$. We define the Mean Square Error (MSE) as :

$$\text{MSE}(\mathbf{x}, \mathbf{x}') = \frac{1}{n} \sum_{i=1}^{n} (\mathbf{x}_i - \mathbf{x}'_i)^2$$

Ideally, the difference between each pixel $\mathbf{x}_i - \mathbf{x}'_i$ is null such that the lower MSE the better. Thus, the distribution of Mean Square Errors among a set of images is, optimally, a Dirac distribution. Figs. 20 and 21 show MSE distributions, respectively over training and testing sets of analytical profiles, for both PCA-based and VAE-based emulators. For a 512 images training set and a 64 images testing set of analytical profiles, MSE distribution of the PCA-based emulator is closer to a Dirac distribution than the VAE-based emulator. This means that, for these datasets, PCA-GP emulator is more able to reconstruct, pixel-wisely, analytical profile images, than VAE-GP emulator. Again, as experienced, VAE-GP emulator median MSE values on this dataset decreases with training set size and outperforms PCA-GP median MSE emulator performance. Fig. 23 shows the MSE distribution related to COMSOS images reconstruction with VAE. This distribution is very close to a Dirac distribution and minimum, maximum and median MSE value, respectively equal to 8.5e-9, 1.2e-6, 1.3e-8, are very low, with, though, a few outlier values. It highlights the very good ability of VAE to reproduce COSMOS-type images.

These first validation tests are designed to provide first-order assessments on emulators described in Sect. 3. These tests and metrics only allow to verify a pixel-wise reconstruction quality. The next section extends validation tests to second and higher metrics that are more likely to confirm that emulators are able to generate physically realistic galaxy images.

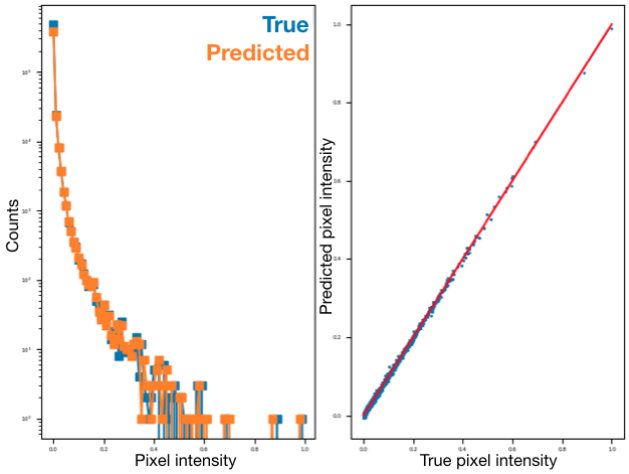

Figure 18: Left-hand plot: histogram distribution of true and predicted pixel intensities for PCA-GP emulated analytical profiles generated in Sect 2.1.1. Ideally, these two distributions are equal. Right-hand plot : predicted versus true pixel intensity scatter plot. It should fit or be similar to identity.

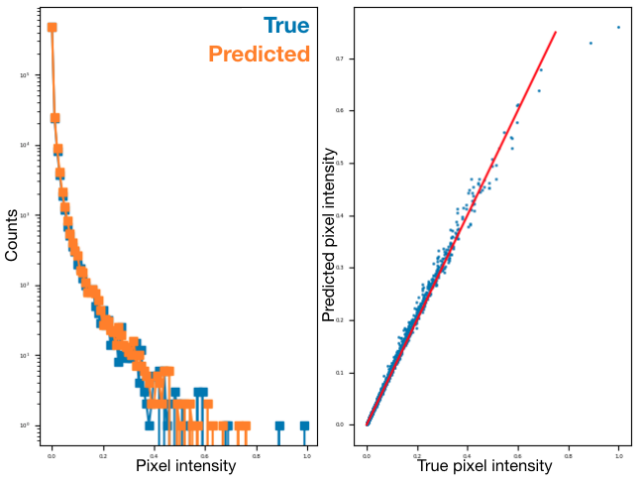

Figure 19: Left-hand plot: histogram distribution of true and predicted pixel intensities for VAE-GP emulated analytical profiles generated in Sect 2.1.1. Right-hand plot : predicted versus true pixel intensity scatter plot.

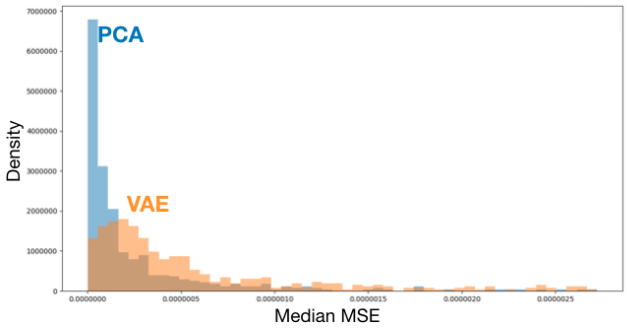

Figure 20: Mean Square Error distribution over the training set composed of 512 analytical profiles. The blue histogram is related to PCA-GP emulation and the orange histogram is related to VAE-GP emulation.

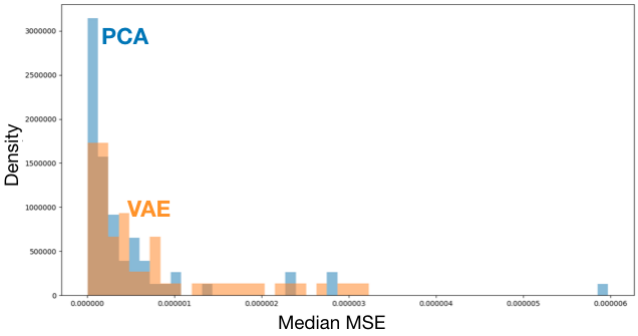

Figure 21: Mean Square Error distribution over the testing set composed of 64 analytical profiles. The blue histogram is related to PCA-GP emulation and the orange histogram is related to VAE-GP emulation.

4.2 Shear measurements, higher order metrics

To validate emulators performances with higher order metrics, we use the shape measurement module of GalSim. This module contains functions for second order moments evaluation of images following Hirata et al. method [12]. These functions find the best elliptical Gaussian that match every input image. These moments are represented as a shear distortion in GalSim and shear parameters estimates of the observed galaxy are returned. The idea here is to estimate shear parameters of emulated images and confirm that they are matching shear parameters of original images. This validation test, of higher order, allows to verify that our emulators are able to generate realistic galaxies for physical parameters recovering, beyond visual confirmation and pixel-to-pixel comparison. Fig. 24 top row shows true (in blue) and predicted (in orange) distributions of shear parameters g_1 and g_2. Ideally, these distribution are equal. Fig. 24 bottom row shows predicted shear parameters versus true shear parameters. This scatter plot should match the identity line. The two first left-hand columns plots stand for the PCA-GP emulator and the two last right-hand columns stand for VAE-GP emulator. Both emulators has been trained on the analytical profiles dataset. PCA-GP emulator seems to generate realistic low-shear galaxies, but, as visually observed in Sect. 3.1, high-shear galaxies are reconstructed with a lower shear than expected. Indeed, a bias is clearly noticed in bottom row scatter plot. This bias does not exist for VAE-GP emulation, for which true and predicted shear parameters distributions are very close and scatter plot of predicted versus true shear parameters matches almost perfectly identity. Despite first-order validation tests outperformed by PCA-GP emulation, VAE-GP seems to be able to generate more physically realistic galaxies, with no bias in shear parameters recovering. Fig. 25

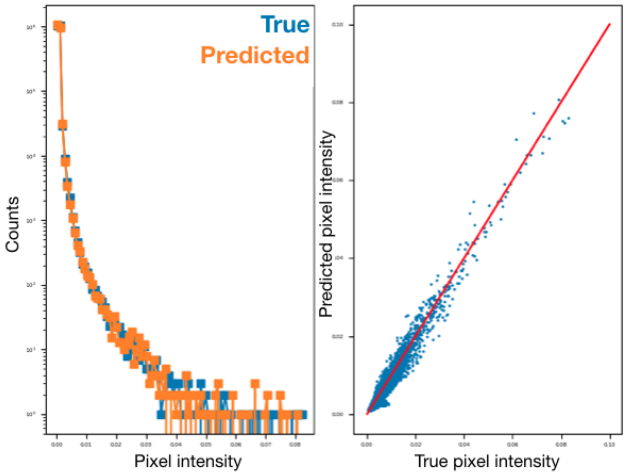

Figure 22: Left-hand plot: histogram distribution of true and predicted pixel intensities for VAE generated COSMOS profiles generated in Sect 2.1.2. Right-hand plot : predicted versus true pixel intensity scatter plot.

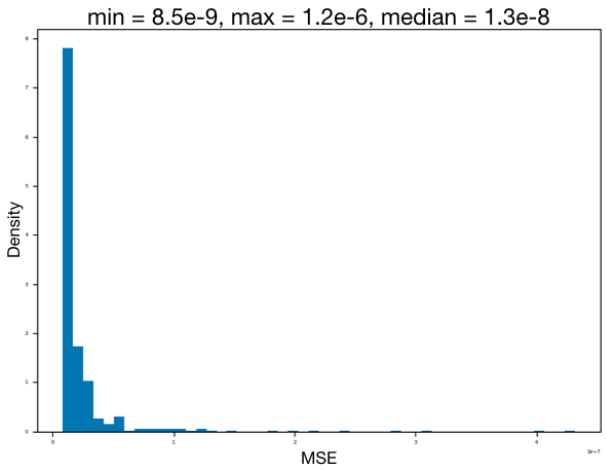

Figure 23: Mean Square Error distribution over the training set composed of analytical profiles. The blue histogram is related to PCA-GP emulation and the orange histogram is related to VAE-GP emulation.

Emulator	**Analytical profiles**	
	Small size	Large size
PCA-GP	++	++
VAE-GP	+/++	+++

Generative model	**COSMOS-type**
PCA	+
VAE	++

Table 1: Global performance qualitative assessments for every emulator validated in this section and trained on analytical profile images (left-hand table) and COSMOS images (right-hand images).

shows shows true (in blue) and predicted (in orange) distributions of shear parameters g_1 and g_2 (top row) and predicted shear parameters versus true shear parameters scatter plot (bottom row). Shear estimation has been done for images generated with a VAE trained on the COSMOS-type dataset. The two first left-hand columns plots stand training set images shear estimation and the two last right-hand columns stand for testing set images shear estimation. Again, the first two left-hand columns show that VAE is a very good model to generate realistic COSMOS-type images. Indeed, true and predicted shear parameters distributions are very close and scatter plot of predicted versus true shear parameters spreads around identity. However, results for testing set images (right-hand columns) are not as good as for training set images. This highlights an over-training problem. This problem makes the VAE generative model very good at reproducing training set images but not as good as reproducing image that have never been seen by the network in the training process. This situation should be solved with a bigger training set size, or a shorter training.

We also propose to compare emulated images given physical parameters with real or simulated datasets such as the Illustris/TNG simulation [13] or the Sloan Digital Sky Survey (SDSS) [14] through several indicators of dispersion between distributions of recovered parameters. An extension of shear measurement validation test could be more generally proving that the emulator is able to generate physically realistic images compared to other simulated or real datasets. Given a number of physical parameters, galaxy images are emulated. Then, these images are convolved with a specific PSF and are corrupted with a specific noise model, depending on the dataset we want our emulator to be compared with. As example, if we want to compare our dataset with images taken by the Hubble Space Telescope, we will convolve output images from our emulator with an HST PSF and add Poisson and specific HST noise. This process would generate realistic galaxy images as long as the PSF applied in post-processing is (non-strictly) larger than the PSF applied to images of the training set. Finally, we compare the distribution of some physical parameters recovered with extraction algorithms as well as Gini-m20 indexes [15] calculated for the emulated dataset and the simulated (TNG) or real dataset. This validation test will be performed as soon as a VAE-MAF emulator is working great and is validated with first and second order tests with a COSMOS-type training set.

Table 1 shows a global performance qualitative assessment for all emulators and training sets introduced in this work. Every validation scheme is expected to be performed on a well-trained VAE-MAF emulator and its performance are likely to outperform current COSMOS-type generation and emulate realistic galaxies for other simulated or real datasets.

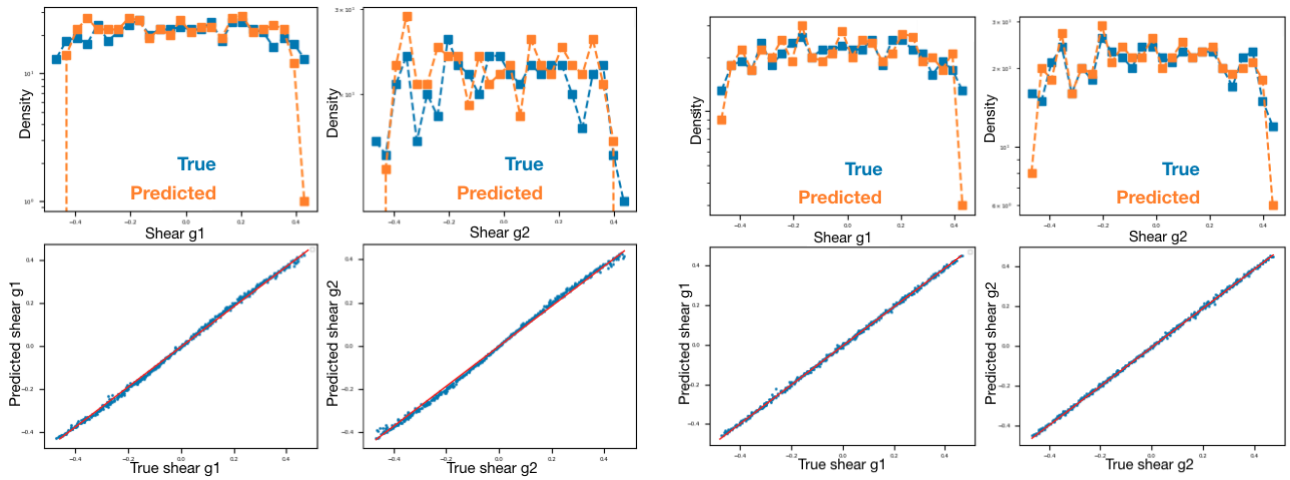

Figure 24: Top row: true (in blue) and predicted (in orange) distributions of shear parameters g_1 and g_2 (top row). Bottom row: predicted shear parameters versus true shear parameters scatter plot. Left-hand columns: shear estimation for PCA-GP emulated analytical profile images. Right-hand column: shear estimation for VAE-GP emulation of analytical profile images.

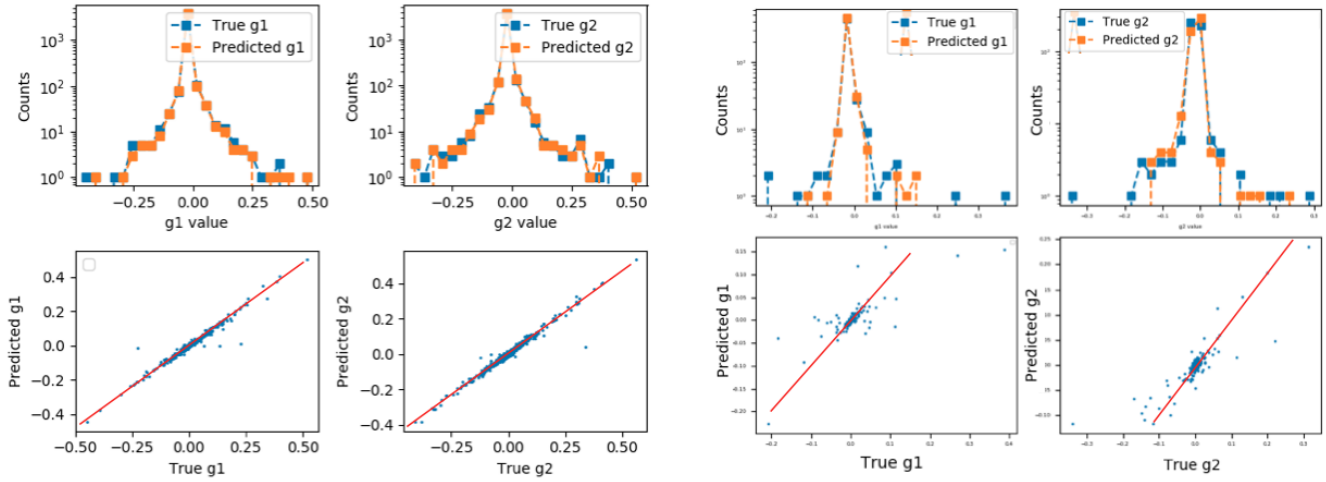

Figure 25: Top row: true (in blue) and predicted (in orange) distributions of shear parameters g_1 and g_2 (top row). Bottom row: predicted shear parameters versus true shear parameters scatter plot. Left-hand columns: shear estimation over the training set for VAE emulated analytical profile images. Right-hand column: shear estimation oveer the testing set for VAE emulation of analytical profile images.

5 Future work and conclusion

5.1 Perspectives

In the near future, efforts will be made on three main perspectives. VAE performances on generating realistic COSMOS images still need to be improved. Better performances of this VAE on testing set images could be achieved in carefully train the network to avoid over-fitting. Then, general performances could benefit from data augmentation. Indeed, we highlighted in Sect. 2.1.2 that the distribution of physical parameters on the COSMOS-type dataset in not uniform and this leads to bad generation of outliers that are under-represented in this dataset.

Once a better generator is trained as generating realistic images, we will train a conditioning process to interpolate from physical parameters to latent variable to be decoded using MAFs. As dependency between physical parameters and latent space has been shown in Sect. 3.2, we should be able to perform this conditioning, modifying MAF hyper-parameters.

Then, we will actually perform meticulous assessments with Illustris/TNG simulations to prove that the VAE-MAF emulator is able to generate physically realistic images compared with this simulated data. The general idea is to compare the distribution of some physical parameters, recovered with extraction algorithms, calculated for the emulated dataset and the simulated (TNG) or real dataset.

5.2 About other generative models and data

Another perspective in the near future could be training other generative models to figure out which one is the best for this kind of emulation and study whether or not any mapping from physical parameters is possible. An ensemble of generative models could be tested here : Vector Quantized VAE (VQ-VAE[16]), Generative Adversarial Networks (GAN[2]) or Pixel Convolutional Neural Networks (Pixel-CNN[17]).

VQ-VAEs[16] is a generative model for large scale image generation. It is composed of an encoder and a decoder, as a classic VAE. But the encoded variable is quantized based on its distance to prototype vectors in a codebook. The latent variable is then replaced by the index of the nearest prototype vector in the codebook. The decoder is trained to map the original image from this nearest prototype vector. The main advantages of such architecture is the fast training due to quantization and the broader diversity of trained images. GANs[2] are also generative models designed for large scale image generation but is build with two neural networks competing with each other. The first network is a generator neural network creating images by mapping random noise into an image, and the second network is a discriminator, which classifies input images as real or fake. The generator is trained at fooling the discriminator and the discriminator is trained at recognizing generated images. Larger scale GANs can generate high-quality and high-resolution images but are known to be challenging to train and evaluate. PixelCNNs[17] models the joint distribution of pixels of an image as a product of conditional distributions of a pixel given every previous pixel. Every conditional distribution is modelled by a convolutional neural network. The conditioning given every previous pixel is modelled as a mask for convolutional layers. Therefore, PixelCNN model is several masked convolutional layers stacked. Pixel are predicted sequentially, which means that each time a pixel is predicted, it is fed back into the neural network to predict the next pixel. The major drawback of PixelCNNs is their performance generating realistic images.

As emulating can be done for many applications and many kind of datasets, we would like to perform emulation on diverse astronomical datasets, for which simulations are very expensive. In the future, we will design a generative model for 3-D magneto-hydrodynamical (MHD) structures in the Interstellar Medium, 2-D radiative transfer images, and 1-D emission spectra of star-formation regions.

5.3 Conclusions

In this work, we proposed an ensemble of generative models trained at interpolating physical parameters onto galaxy images. The project involved creating two different training sets respectively composed of simulated profiles and real COSMOS-type images. Then, we designed three emulators, made of a generator that emulate images from a latent variable and an interpolator or conditioning process that map from physical parameters into a latent variable. The first emulator is composed of a linear generative model, Principal Components Analysis and a Gaussian Process interpolator, expected to be powerful for simple analytical profiles emulation but outperformed very fast by deep generative models. The second emulator is composed of a deep generative model, a Variational Autoencoder (VAE), and a Gaussian Process interpolator. As interpolation with Gaussian Processes can fail when this interpolation from physical parameters to a latent variable is not Gaussian, we proposed to use a deep conditional density estimator, Masked Autoregressive Flow (MAF). This third emulator made of VAE as a generative model and MAF as a conditioning process is not functional yet but expected to be in the very near future.

With several assessment pipelines, from first order to higher order, we demonstrated on analytical profiles the capability of deep neural networks to emulate physically realistic images compared to linear models as PCA. We then showed that VAE is also able to reproduce realistic COSMOS-type images. An effort in the near future will be put at conditioning this generative model given physical parameters and prove its ability to generate realistic images compared with other real or simulated datasets as Illustris/TNG.

Several generative models, as VQ-VAEs, GANs or PixelCNNs, will also be trained and assessed at emulating realistic images as detailed in Sect. 5.2. Then, we will experiment emulation on other astronomical surveys, such as MHD simulations or radiative transfer images.

Acknowledgements

This report describes the work conducted during the Kavli Summer Program in Astrophysics, hosted in University of California Santa Cruz in 2019. We would like to thank the Kalvi Foundation and all the organisers involved in this summer school for their generous support.

References

[1] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," *arXiv e-prints*, p. arXiv:1312.6114, Dec 2013.

[2] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Networks," *arXiv e-prints*, p. arXiv:1406.2661, Jun 2014.

[3] P. Capak, H. Aussel, M. Ajiki, H. J. McCracken, B. Mobasher, and N. e. a. Scoville, "The First Release COSMOS Optical and Near-IR Data and Catalog," , vol. 172, no. 1, pp. 99–116, Sep 2007.

[4] A. Leauthaud, R. Massey, J.-P. Kneib, J. Rhodes, D. E. Johnston, P. Capak, C. Heymans, R. S. Ellis, A. M. Koekemoer, O. Le Fèvre, Y. Mellier, A. Réfrégier, A. C. Robin, N. Scoville, L. Tasca, J. E. Taylor, and L. Van Waerbeke, "Weak Gravitational Lensing with COSMOS: Galaxy Selection and Shape Measurements," , vol. 172, no. 1, pp. 219–238, Sep 2007.

[5] G. Papamakarios, T. Pavlakou, and I. Murray, "Masked Autoregressive Flow for Density Estimation," *arXiv e-prints*, p. arXiv:1705.07057, May 2017.

[6] "Monte Carlo Sampling Methods using Markov Chains and their Applications," *Biometrika*, vol. 57, no. 1, pp. 97–109, Apr 1970.

[7] K. Heitmann, M. White, C. Wagner, S. Habib, and D. Higdon, "The Coyote Universe. I. Precision Determination of the Nonlinear Matter Power Spectrum," , vol. 715, no. 1, pp. 104–121, May 2010.

[8] M. Mustafa, D. Bard, W. Bhimji, Z. Lukić, R. Al-Rfou, and J. M. Kratochvil, "CosmoGAN: creating high-fidelity weak lensing convergence maps using Generative Adversarial Networks," *Computational Astrophysics and Cosmology*, vol. 6, no. 1, p. 1, May 2019.

[9] J. Caldeira, W. L. K. Wu, B. Nord, C. Avestruz, S. Trivedi, and K. T. Story, "DeepCMB: Lensing reconstruction of the cosmic microwave background with deep neural networks," *Astronomy and Computing*, vol. 28, p. 100307, Jul 2019.

[10] B. T. P. Rowe, M. Jarvis, R. Mandelbaum, G. M. Bernstein, J. Bosch, M. Simet, J. E. Meyers, T. Kacprzak, R. Nakajima, J. Zuntz, H. Miyatake, J. P. Dietrich, R. Armstrong, P. Melchior, and M. S. S. Gill, "GALSIM: The modular galaxy image simulation toolkit," *Astronomy and Computing*, vol. 10, pp. 121–150, Apr 2015.

[11] M. Mckay, R. Beckkman, and W. Conover, "Comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, vol. 21, pp. 266–294, 01 2000.

[12] C. Hirata and U. Seljak, "Shear calibration biases in weak-lensing surveys," , vol. 343, no. 2, pp. 459–480, Aug 2003.

[13] D. Nelson, A. Pillepich, V. Springel, R. Pakmor, R. Weinberger, S. Genel, P. Torrey, M. Vogelsberger, F. Marinacci, and L. Hernquist, "First Results from the TNG50 Simulation: Galactic outflows driven by supernovae and black hole feedback," *arXiv e-prints*, p. arXiv:1902.05554, Feb 2019.

[14] K. Abazajian, J. K. Adelman-McCarthy, M. A. Agüeros, S. S. Allam, S. F. Anderson, and J. e. a. Annis, "The First Data Release of the Sloan Digital Sky Survey," , vol. 126, no. 4, pp. 2081–2086, Oct 2003.

[15] J. M. Lotz, J. Primack, and P. Madau, "A New Nonparametric Approach to Galaxy Morphological Classification," , vol. 128, no. 1, pp. 163–182, Jul 2004.

[16] A. Razavi, A. van den Oord, and O. Vinyals, "Generating Diverse High-Fidelity Images with VQ-VAE-2," *arXiv e-prints*, p. arXiv:1906.00446, Jun 2019.

[17] A. van den Oord, N. Kalchbrenner, O. Vinyals, L. Espeholt, A. Graves, and K. Kavukcuoglu, "Conditional Image Generation with PixelCNN Decoders," *arXiv e-prints*, p. arXiv:1606.05328, Jun 2016.

A Latin hypercube sampling scheme

Latin hypercube sampling scheme is a statistical sampling method used to generate a near-random sample of values from a multi-dimensional distribution. The idea is to sample a function of N parameters dividing the range of each parameter into M equally probable subdivisions separated by hyperplanes. A hypercube is then a latin hypercube if and only if there is only one sample in each subdivision. In $N = 2$ dimensions, a sampled grid is a latin square if and only if there is only one sample in each row and each column, as shown in Fig.26.**a)**. The distribution is here a multivariate uniform distribution and thus, the $M = 5$ subdivisions are regularly distributed along each axis. Compared to a random space-filling scheme Fig 26.**b)**, which does not take into account the previous sampled points in a new sampled point generation, latin hypercube sampling ensures an optimal representation of the variability of parameters, given a number of samples. Given a number a of subdivisions, latin hypercube sampling ensures an optimal number of samples compared to an uniform space-filling scheme. Indeed, as highlighted in Fig.26.**c)**, the number of samples is equal to N^M in an uniform space-filling scheme versus M in latin hypercube sampling. Considering the significant cost of many data generation algorithms, this sampling method is particularly useful to generate a representative dataset at low cost.

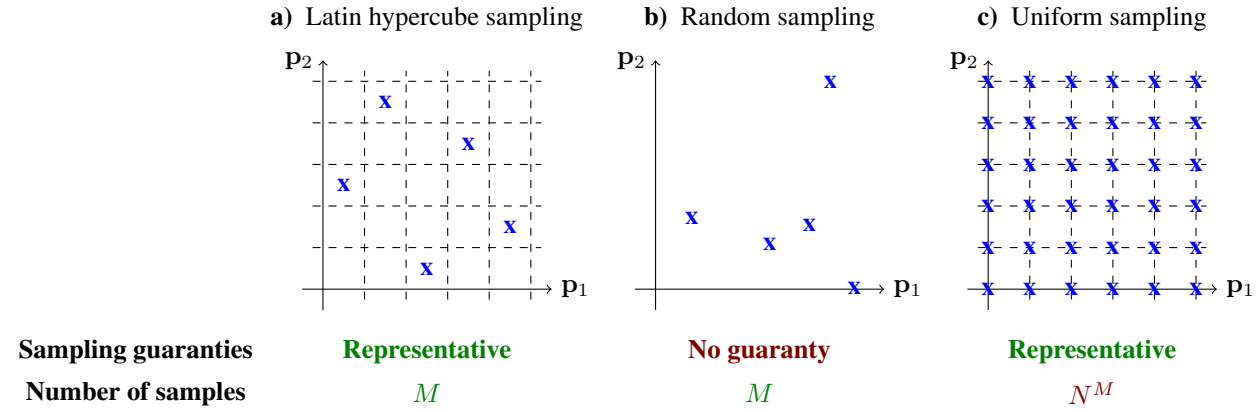

Figure 26: **a)** Latin hypercube, **b)** random, and **c)** uniform sampling in $N = 2$ dimensions with $M = 5$ subdivisons for parameters $\mathbf{p}_1$ and $\mathbf{p}_2$.

B Gaussian Processes

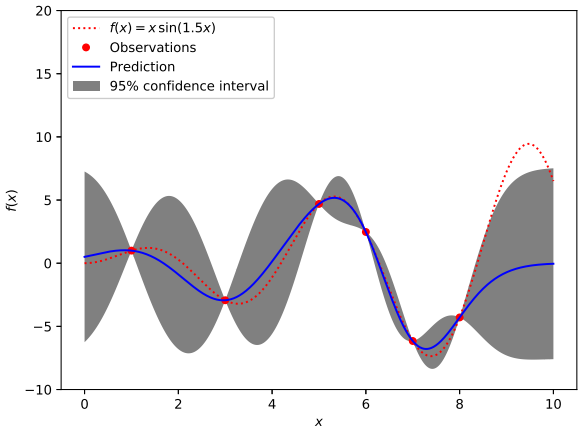

Figure 27: Gaussian Processes illustration in 1 dimension.

Gaussian processes offer a non-linear alternative to commonly used regression problems. The idea is to find a multivariate Gaussian distribution over any function f such that $\mathbf{z} = f(\mathbf{p}) + \epsilon$, where ϵ is the error induced by the model. In other words, a GP assumes that the probability $p(\mathbf{z}_1, ..., \mathbf{z}_n) = p(f(\mathbf{p}_1), ..., f(\mathbf{p}_n))$ is jointly Gaussian with some mean $\mu(\mathbf{p})$ and some covariance $\mathbf{K}(\mathbf{p}, \lambda)$ where $\mathbf{K}_{ij} = k(\mathbf{p}_i, \mathbf{p}_j, \phi)$, k is the positive definite kernel function, which constrains $f(\mathbf{p}_i)$ and $f(\mathbf{p}_j)$ to be similar if $\mathbf{p}_i$ and $\mathbf{p}_j$ are similar, and λ hyperparameters. The distribution of functions, from a prior knowledge i.e. the choice of the kernel function, is updated with observed samples from that function f by maximizing the log-likelihood $\log p(f(\mathbf{p})|\mathbf{p}, \lambda)$. An example of Gaussian Processes in 1 dimension is given Fig. 27. The original function to approximate is $f(x) = x \sin(1.5x)$ and the observed data is the set of red dots. A GP learned a distribution of functions that fits observations. The area in grey is the area for which there is a 95% chance that a realization of this distribution is contained inside it, and the prediction function in blue is the mean of these realizations that interpolates from the red dots. The choice of a kernel here would have had an influence on the width of the 95% confidence interval and the smoothness of realization functions, depending on the prior knowledge available on the function to approximate. The main drawback of Gaussian Processes is their computational cost. Indeed, As a GP is a non-parametric method, the computational cost of its training can be very expensive and grows cubicly with the training

set size and dimension. The dimensionality reduction performed by any generative model described in this work is thus a key process to reduce computational costs and training times.