The Astronomical Journal, 156:268 (14pp), 2018 December https://doi.org/10.3847/1538-3881/aae77c
© 2018. The American Astronomical Society.
OPEN ACCESS

ExoGAN: Retrieving Exoplanetary Atmospheres Using Deep Convolutional Generative Adversarial Networks

Tiziano Zingales[1,2] and Ingo P. Waldmann[1]
[1] University College London Gower Street, Bloomsbury, London WC1E 6BT, UK; tiziano.zingales.15@ucl.ac.uk
[2] INAF—Osservatorio Astronomico di Palermo Piazza del Parlamento 1, 90134 Palermo, Italy
Received 2018 June 7; revised 2018 October 8; accepted 2018 October 8; published 2018 November 15

Abstract

Atmospheric retrievals on exoplanets usually involve computationally intensive Bayesian sampling methods. Large parameter spaces and increasingly complex atmospheric models create a computational bottleneck forcing a trade-off between statistical sampling accuracy and model complexity. It is especially true for upcoming *JWST* and ARIEL observations. We introduce ExoGAN, the Exoplanet Generative Adversarial Network, a new deep-learning algorithm able to recognize molecular features, atmospheric trace-gas abundances, and planetary parameters using unsupervised learning. Once trained, ExoGAN is widely applicable to a large number of instruments and planetary types. The ExoGAN retrievals constitute a significant speed improvement over traditional retrievals and can be used either as a final atmospheric analysis or provide prior constraints to subsequent retrieval.

Key words: methods: statistical – planets and satellites: atmospheres – radiative transfer – techniques: spectroscopic

1. Introduction

The modeling of exoplanetary atmospheric spectroscopy through so-called atmospheric retrieval algorithms has become the accepted standard in the interpretation of transmission and emission spectroscopic measurements (e.g., Rocchetto et al. 2016; Barstow et al. 2017; Sheppard et al. 2017; Bruno et al. 2018; Kreidberg et al. 2018; Mansfield et al. 2018; Spake et al. 2018; Tsiaras et al. 2018). These retrieval algorithms are designed to solve the often ill-posed inverse problem of determining atmospheric parameters (such as trace gas abundances) from the measured spectra and their corresponding measurement uncertainties (e.g., Irwin et al. 2008; Madhusudhan & Seager 2009; Benneke & Seager 2013; Line et al. 2013; Cubillos et al. 2016; Lavie et al. 2017; Gandhi & Madhusudhan 2018). The associated atmospheric forward model to be fitted varies in complexity from retrieval to retrieval, but most times encompass a high dimensional likelihood space to be sampled. In the era of *JWST* (Gardner et al. 2006) and ARIEL (Tinetti et al. 2016) observations, said model complexity will have to increase significantly. To date, the most commonly adopted statistical sampling methods are Nested Sampling (Skilling 2004; Feroz & Hobson 2008; Feroz et al. 2009) and Markov Chain Monte Carlo (e.g., Gregory 2011). These approaches typically require of the order of 10^5–10^6 forward model realizations until convergence. The traditional analysis method, which uses Bayesian statistics, creates a precarious bottleneck: to achieve convergence within reasonable time frames (hours to days), we require the atmospheric forward model to be fast and consequently overly simplistic. The inclusion of disequilibrium chemistry, self-consistent cloud models, and the move from one-dimensional (1D) to two- or three-dimensional (2D, 3D) radiative transfer, are largely precluded by this constraint. In this paper, we present the first deep-learning architecture for exoplanetary atmospheric retrievals and discuss a path toward solving the computational bottleneck using atmospheric retrievals assisted by deep learning.

Artificial Intelligence has been used extensively to understand and describe complex structures and behavior in a wide variety of data sets across a plethora of research fields.

In recent years, the field of exoplanets has seen pioneering deep-learning papers on planet detection (Pearson et al. 2018; Shallue & Vanderburg 2018), exoplanet transit prediction (Kipping & Lam 2017), and atmospheric spectral identification Waldmann (2016). In Waldmann (2016), we applied a deep-belief neural network (DBN) to recognize the atmospheric features of an exoplanetary emission spectrum. This approach provided a qualitative understanding of the atmospheric trace gases likely to be present in a planetary emission spectrum, to then be included in our atmospheric retrieval framework TauREx (Waldmann et al. 2015a, 2015b). In this paper, we introduce a generative adversarial network (GAN; Goodfellow et al. 2014) to predict the maximum likelihood (ML) of the full retrieval solution given the observed spectrum. As shown in the following sections, this can be used as a stand-alone solution to retrieval or used to constrain the prior parameter ranges for a more standard atmospheric retrieval later.

We designed our algorithm following four guiding principles:

1. Once trained, the deep- or machine-learning algorithm should apply to the widest possible range of planet types.
2. Once trained, the algorithm should apply to a wide range of instruments.
3. The algorithm should be robust in the presence of unknown "un-trained" features and be able to generalize to parameter regimes outside its formal training set.
4. The design of the algorithm and data format should be modular and easily modifiable and expandable.

In the following sections, we present the Exoplanet Generative Adversarial Network (ExoGAN) algorithm and demonstrate it on a variety of retrieval scenarios. We provide the ExoGAN algorithm freely to the community (see end of this paper).

2. Method

In the following sections, we will introduce GANs and deep convolutional generative adversarial networks (DCGANs), followed by a discussion on how we adopt DCGANs for exoplanetary retrievals.

2.1. Generative Adversarial Networks

Generative Adversarial Networks first introduced by Goodfellow et al. (2014) belong to the class of unsupervised deep generative neural networks (Goodfellow et al. 2016). Deep generative models can learn the arbitrarily complex probability distribution of a data set, $p_{\rm data}$, and can generate new data sets drawn from $p_{\rm data}$. Similarly, they can also be used to fill in missing information in an incomplete data set, so-called inpainting. In this work, we use the data inpainting properties of the GAN to perform retrievals of the atmospheric forward model parameters.

The most common analogy for a GAN architecture is that of a counterfeit operation. The neural network is given a training data set, $\boldsymbol{x}$, in our case combinations of atmospheric spectra with their associated forward model parameters. We refer to the training set as the "real" data with the probability distribution $p_{\rm data}$. Now two deep neural networks are pitted against each other in a `minmax` game. One network, the generator network (G), will try to create a "fake" data set (p_g), indistinguishable from the "real" data. In a second step, a second neural network, the discriminator (D), tries to classify "fake" from "real" data correctly. The training phase of the GAN is completed when a Nash equilibrium is reached, and the discriminator cannot identify real from fake any longer. At this stage, the generator network will have learned a good representation of the data probability distribution and $p_g \simeq p_{\rm data}$. Figure 1 shows a schematic of our GAN implementation. Unlike for variational inference methods, such as variational autoencoders (VAE; Kingma & Welling 2013; Jimenez Rezende et al. 2014), the functional form of the data likelihood does not need to be specified but is learned by the Generator. Such implicit latent variable models or likelihood-free networks allow the learning of arbitrarily complex probability distributions in an unsupervised manner while assuming minimal prior assumptions on the data distribution.

GANs have been applied to multiple problems, such as semi-supervised learning, stabilizing sequence learning methods for speech and language, and 3D modeling (Denton et al. 2015; Radford et al. 2015; Lamb et al. 2016; Salimans et al. 2016; Wu et al. 2016). Notable examples of GANs applied in an astrophysical context are given by Rodriguez et al. (2018), Stark et al. (2018), and Schawinski et al. (2017), who used GANs trained on existing N-body simulations to efficiently generate new, physically realistic realizations of the cosmic web, learn Point Spread Function from data or de-noise ground-based observations of galaxies.

In the field of exoplanets, the use of GANs or similar deep architectures has not yet been explored. In this work, we base ExoGAN on a Deep Convolutional Generative Adversarial Network (DCGAN; Radford et al. 2015).

DCGANs are an evolution from the classical GAN by replacing the multilayer perceptrons (MLPs; Rumelhart et al. 1986; Bengio 2009) in the Generator and Discriminator networks with all convolutional layers. Their characteristics makes DCGAN significantly more robust to discrete-mode and manifold model collapse (Metz et al. 2016; Arjovsky & Bottou 2017) and are found to be stable in most training scenarios (Radford et al. 2015). The use of batch normalization (Appendix B) further increases training speed and robustness. Besides, we note that convolutional networks are ideally suited to capturing the highly correlated signals of broad, roto-vibrational spectral bands in NIR and IR wavelengths.

2.2. Adversarial Training

As described in the previous section, both Generator and Discriminator networks are pitted against one another during training. The goal of the training phase is to reach a Nash Equilibrium, i.e., when neither player can improve by unilaterally changing one's strategy. Figure 1 shows a schematic of the ExoGAN setup.

In order to return the generator distribution p_g over the data $\boldsymbol{x}$, we start from a prior distribution of Gaussian-distributed latent variables $p(z)$ and define $G(z; \boldsymbol{\theta}_G)$ as the mapping from latent variable space to generated data. Here $\boldsymbol{\theta}_G$ are the hyperparameters of the Generator network (see Appendix A).

Let $D(\boldsymbol{x})$ be the probability that $\boldsymbol{x}$ came from the data rather than p_g. Hence, in the state of convergence, we have $p_g = p_{\rm data}$ and $D(\boldsymbol{x}) = \frac{1}{2}$. In the training phase, we need D to maximize the probability of assigning the correct label to both training examples and samples from G. At the same time, we want G to minimize the probability $\log(1 - D(G(z)))$. We can now define the cross-entropy cost function of the Discriminator as:

$$J^{(D)} = -[\log D(\boldsymbol{x}) + \log(1 - D(G(z)))]. \tag{1}$$

During training, we employ batch training, with the cost function of a batch of n data samples being

$$J^{(D)} = -\left\{\sum_{i=1}^{n} \log D(x_i) + \right. \\ \left. + \sum_{i=1}^{n} \log(1 - D(G(z_i)))\right\} \tag{2}$$

which can be written as the expectation values over the data and generated samples:

$$J^{(D)} = -\{\mathbb{E}_{\boldsymbol{x}\sim p_{\rm data}}[\log D(\boldsymbol{x})] \\ + \mathbb{E}_{z\sim p_z}[\log(1 - D(G(z)))]\}. \tag{3}$$

Since the discriminator wants to minimize the cost function and the generator wants to maximize it, we can summarize the training as a zero-sum game where the cost function for the generator is given by: $J^{(G)} = -J^{(D)}$. Hence, to capture the entire game, we only need to specify the loss function of the Discriminator since it encompasses both $\boldsymbol{\theta}^{(D)}$ and $\boldsymbol{\theta}^{(G)}$ hyperparameters. We then optimize the value function $V(\boldsymbol{\theta}^{(D)}, \boldsymbol{\theta}^{(G)}) = -J^{(D)}(\boldsymbol{\theta}^{(D)}, \boldsymbol{\theta}^{(G)})$,

$$\min_G \max_D V(D, G) = \mathbb{E}_{x\sim p_{\rm data}}[\log D(\boldsymbol{x})] \\ + \mathbb{E}_{z\sim p_z}[\log(1 - D(G(z)))]. \tag{4}$$

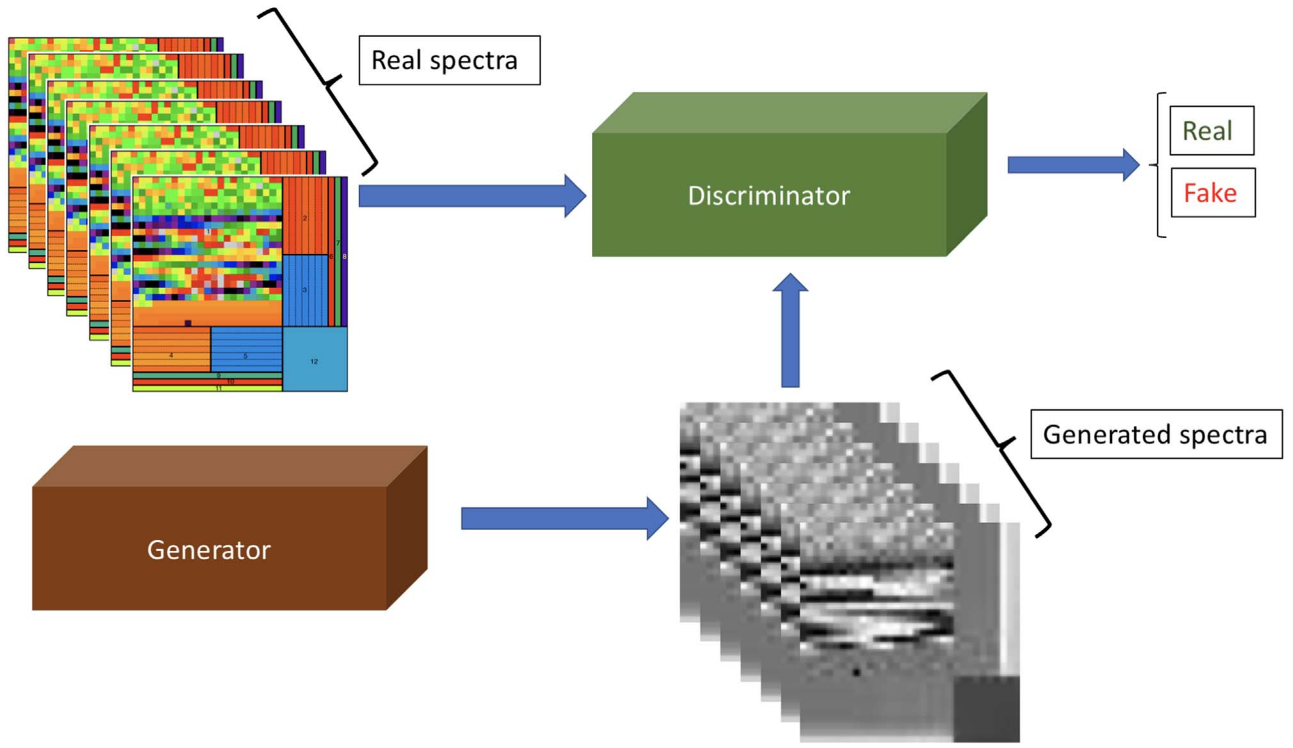

Figure 1. The ExoGAN scheme. The Generator produces data sets sampling from a latent variable space z. The Discriminator compares the generated data set with data drawn from the training set (top left). The network has converged when the Discriminator cannot differentiate Real spectra from Generated Spectra any longer.

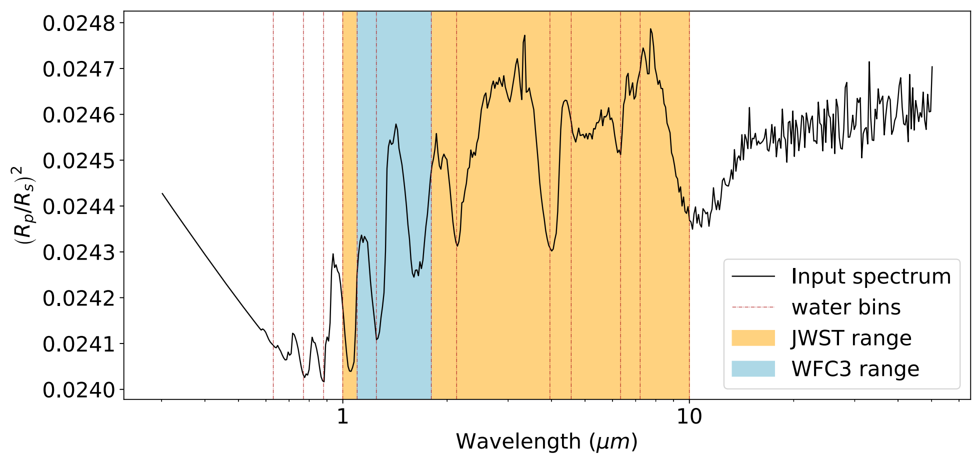

Figure 2. Spectral binning used in this work. The black line is a simulated spectrum of the hot-Jupiter HD 189733b. The red vertical lines represent the bin edges of prominent water bands. The blue and orange areas are the *Hubble*/WFC3 and *JWST* band-passes considered in this paper, respectively.

As stated earlier, Equation (4) constitutes a `minmax` game since it involves minimizing over G in an outer loop and maximizing over D in an inner loop.

2.3. Application to Exoplanet Spectra

Here we explain the data format of the input and training data. In Figure 2 we show an example a transmission spectrum of a cloud-free hot-Jupiter with water as the only trace gas at $3 \cdot 10^{-4}$ volume mixing ratio at a constant resolution of $\frac{\Delta\lambda}{\lambda} = 100$. We train ExoGAN on a wavelength range of 0.3 -50 μm. For this paper, we restrict our sampling resolution to be $R = 100$ for every spectrum. This choice, however, does not preclude training with higher-resolution data in the future.

2.3.1. Normalization

For the neural network to learn efficiently, we must normalize the data to lie between zero and unity. We have experimented with various normalization schemes. The most obvious scheme is a “global” normalization, where we normalize the full training set by its global maximum and minimum values. This approach proved problematic as spectral signatures for planets with low trace-gas abundances and small atmospheric scale heights would be too weak/flat to be recognizable by the neural network for reasonable training times. We have therefore opted to normalize each training spectrum to amplify the spectral features. Assuming that the most common broadband absorber is water in an exoplanetary atmosphere, we divide the spectral range along its major water bands in the IR, see dashed red lines in Figure 2. Note that this does not mean that water-free atmospheres cannot be

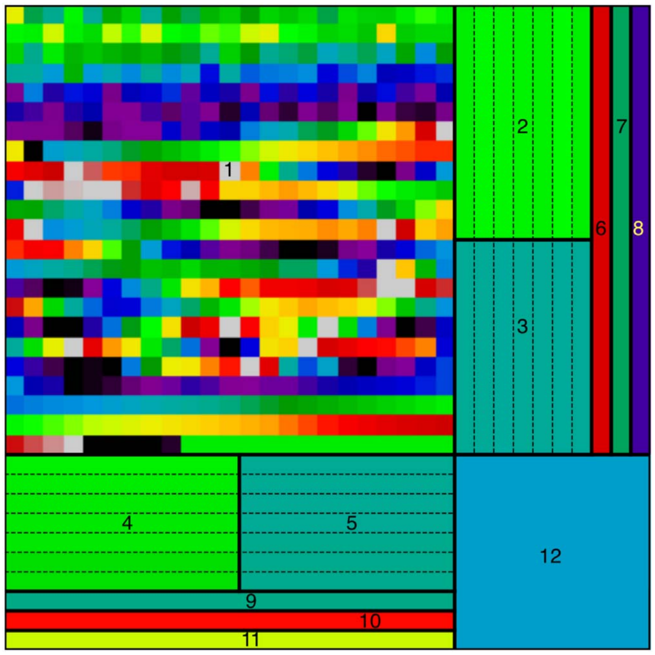

Figure 3. The Atmospheric Spectra and Parameters Array (ASPA). Each area is dedicated to a particular atmospheric characteristic: Area 1 is the spectrum between 1 and 50μm at resolution 100 normalized between 0 and 1 in each spectral bin. Areas 2 to 5 give information about the normalization factors used in the different section of the spectrum, clear and dark area give, respectively, information about the maximum values and the minimum values. In areas 6 to 8, we encode the atmospheric trace-gas volume mixing ratios of CO_2, CO, and CH_4 respectively. Areas 9 to 11 are, respectively M_p, R_p, and T_p. Area 12 gives information on the H_2O trace-gas volume mixing ratio.

detected. Additionally, we divide the spectrum by the pass-bands of the *JWST*/NIRISS, NIRCam, and MIRI instruments (Kalirai 2018) and the *Hubble*/WFC3 instrument passband. In total, we have 14 spectral bands. We now normalize each spectral band between 0 and 1 and record the minimum and maximum normalization factors for each. This normalization scheme ensures a maximum amplification of the spectral features while retaining reversibility.

2.3.2. The Atmospheric Spectrum and Parameters Array (ASPA)

To store all aspects of an atmospheric transmission spectrum, we define the Atmospheric Spectrum and Parameters Array (ASPA). It is a 2D array encoding the 1D normalized spectral bands, each band's minimum and maximum normalization factors and the associated forward model parameter values. We parameterize each training spectrum with seven forward model parameters, ϕ, namely: H_2O, CO_2, CH_4 and CO volume mixing ratios, the mass of the planet M_p, the radius R_p, and its isothermal temperature T_p at the terminator. Figure 3 shows a false-color ASPA. For this paper, the ASPA is a 33 × 33 pixel array, with the main part (Section 1) encoding the spectral information. Sections 2–5 encode the normalization factors and 6–12 the atmospheric parameters. By design, the planet's water abundance takes a significantly large range area of the ASPA, reflecting the relative importance of water in forming the spectral continuum. The ASPA format is adaptable to other configurations in the future.

2.4. The Training

To train ExoGAN on a wide range of possible exoplanetary atmospheres, we generated a very comprehensive training set

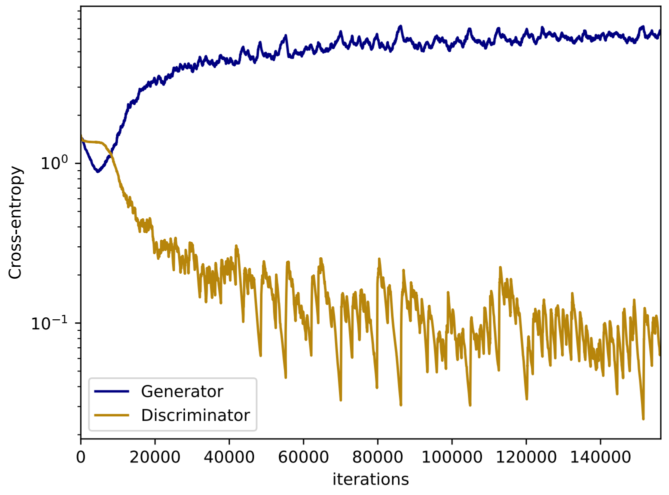

Figure 4. Discriminator (golden) and Generator (blue) cross-entropies as function of the iteration steps.

Table 1
Parameters Boundary Condition Used to Generate the Training Set

Training Set Parameters		
Variable	Lower Bound	Upper Bound
H_2O	10^{-8}	10^{-1}
CO_2	10^{-8}	10^{-1}
CO	10^{-8}	10^{-1}
CH_4	10^{-8}	10^{-1}
M_p	0.8 M_J	2.0 M_J
R_p	0.8 R_J	1.5 R_J
T_p	1000 K	2000 K

Note. Each parameter has been divided into 10 parts and used to model 10^7 different spectra.

of atmospheric forward models using the TauREx retrieval code (Waldmann et al. 2015a, 2015b). We sampled each of the seven previously mentioned forward model parameters (H_2O, CO_2, CH_4, and CO abundances, the mass of the planet M_p, the radius R_p, and the temperature T_p) 10 times within the parameter ranges denoted in Table 1. This configuration yields 10^7 forward models, which are split into 90% training set and 10% test set. The test set is used to validate the accuracy of the network on previously unseen data. As discussed later on, we find this training set to be over-complete and only require a smaller sub-set of the full training set for convergence.

During the training, we perform two training iterations of the discriminator to every training step of the generator. We used an NVIDIA TESLA V100 GPU with minibatch sizes of 64 training ASPAs. We required ~9 hr per epoch on the V100 GPU and comparatively about three days on 20 CPU cores in parallel. The convergences of the loss functions during the training phase are shown in Figure 4. The full model setup can be found in Appendix C (Tables 6 and 7). We tested three different sizes of our latent variable space z, with $z_{\rm dim} = 50$, 100, and 200. We found $z_{\rm dim} = 50$ to yield significantly noisier reconstructions at the end of one epoch of training, whereas no discernible differences between $z_{\rm dim} = 100$ and $z_{\rm dim} = 200$ could be observed. We hence settled on $z_{\rm dim} = 100$. We have adopted a training minibatch size of 64 ASPAs and found no significant effect of larger training batch sizes on network convergence.

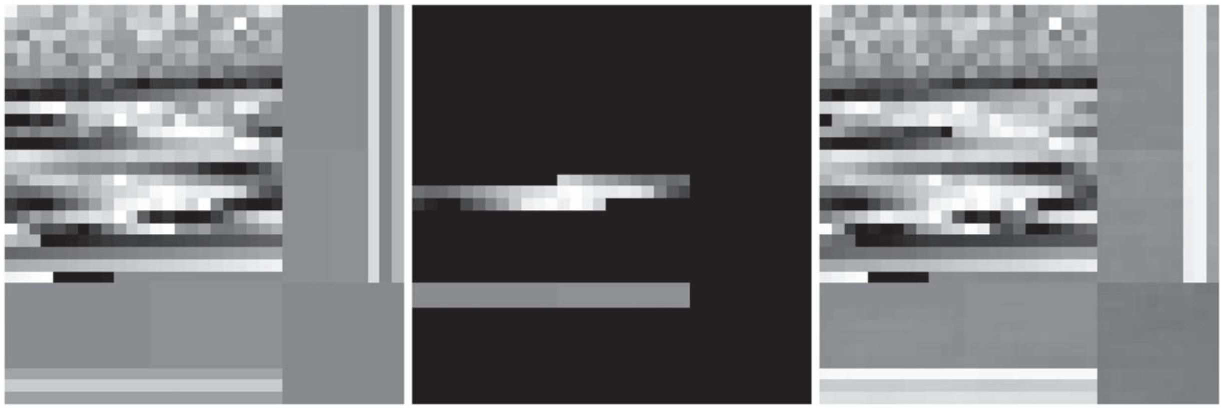

Figure 5. Left: input spectrum together with the parameters pixels. Center: masked ASPA leaving *Hubble*/WFC3 wavelengths only. Right: ExoGAN completed ASPA given the middle ASPA.

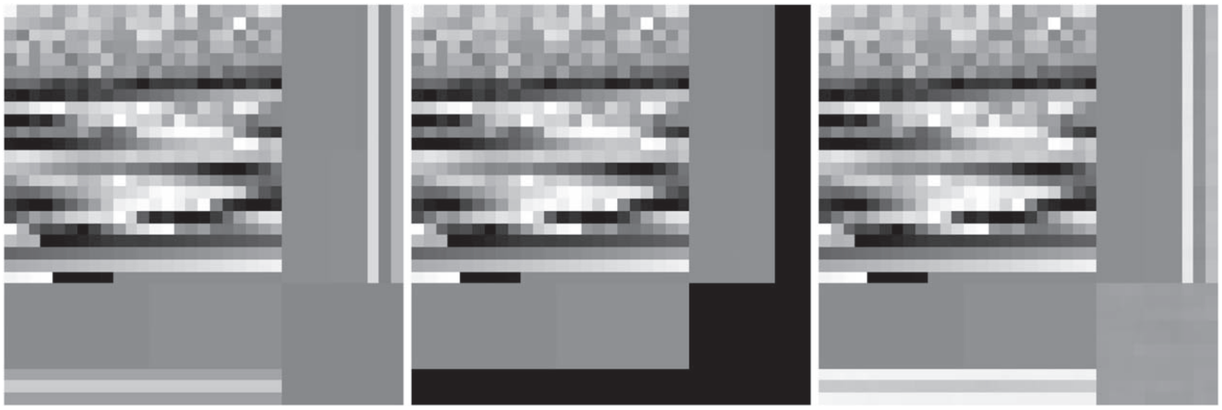

Figure 6. Same as Figure 5 but only masking the atmospheric forward model parameters.

During minibatch training, the algorithm is presented with a sub-set of the full training data (in this case 64 ASPAs) rather than the full training set (or batch). This eases memory requirements of large training set, in particular for memory-limited devices such as GPUs. By only considering a sub-set of training data at a time, a gradient descent optimizer, such as ADAM, is still able to perform well, despite the increase in variance on the gradient estimated. In order to avoid biased estimations and convergence to local minima, minibatches must be selected randomly from the training set at each iteration.

2.5. Data Reconstruction

Once we have trained ExoGAN, we can now define our "retrieval" model. As alluded to above, we use the inpainting properties of a GAN to complete the missing data, in this case, the forward model parameters, in our ASPA. In other words, we convert our observed spectrum into the ASPA format and keep unknown values (parameters and missing wavelength ranges) masked. Given the information available, the ExoGAN will then attempt to fill in the missing information to complete the full ASPA. Here we follow the semantic inpainting algorithm by Yeh et al. (2016).

We can define our reconstructed data, $\boldsymbol{x}_{\text{recon}}$, from the incomplete observed data, $\boldsymbol{y}$, using

$$\boldsymbol{x}_{\text{recon}} = M \odot \boldsymbol{y} + (1 - M) \odot G(\hat{z}) \tag{5}$$

where M is a binary mask set to zero for missing values in $\boldsymbol{y}$, i.e., forward model parameter values and, possibly, missing wavelength ranges. Here, $\odot$ constitutes the Hadamard product and $G(\hat{z})$ is the GAN generated data. We note that after the ExoGAN has been trained, z represents an encoding manifold of p_{data} and we denote the closest match of $(M \odot G(z))$ to $(M \odot \boldsymbol{y})$ with $\hat{z}$, where $\hat{z} \subseteq z$. The aim is now to obtain $\hat{z}$ that accurately completes $\boldsymbol{x}_{\text{recon}}$.

Let us define the following optimization.

$$\hat{z} = \arg\min_{z} \mathcal{L}(z). \tag{6}$$

where $\mathcal{L}$ is a loss function of z that finds its minimum when $\hat{z}$ is reached. Following Yeh et al. (2016), we define the loss function to be comprised of two parts, contextual loss and perceptual loss,

$$\mathcal{L} = \mathcal{L}_{\text{cont}}(z) + \lambda \mathcal{L}_{\text{perc}}(z). \tag{7}$$

The contextual loss, $\mathcal{L}_{\text{cont}}(z)$ is the difference between the observed data and the generated data. Here we follow the definition by Amos (2016):

$$\mathcal{L}_{\text{cont}}(z) = \| M \odot G(z) - M \odot y \|_1. \tag{8}$$

Empirically, Yeh et al. (2016) find the l_1 norm to yield slightly better results, though the l_2 norm can equally be used. Whereas the conceptual loss compares the generated data with the observed data directly, the perceptual loss, $\mathcal{L}_{\text{perc}}(z)$, uses the

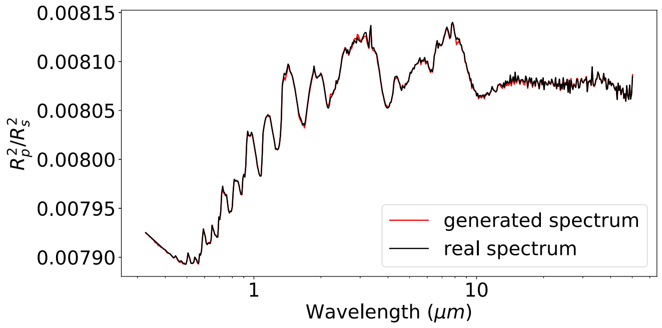

Figure 7. Spectral reconstruction of ExoGAN of a water-dominated *Hubble*/WFC3 spectrum. Black: the ground-truth spectrum; Red: the ExoGAN reconstructed spectrum across all wavelengths giving as input only the *Hubble*/WFC3 bandpass.

discriminator network to verify the validity of the generated data given the training set.

$$\mathcal{L}_{\rm perc}(z) = \log(1 - D(G(z))). \tag{9}$$

To solve Equation (6), we use the ADAM optimizer (Kingma & Ba 2014) with a learning rate of 0.1. For a deeper discussion about the ADAM optimizer, see Appendix C.

We investigated the ratio of perceptual loss (Equation (9)) to contextual loss (Equation (8)) and found $\lambda = 0.1$ to be optimal but note that $\lambda > 0.1$ gives too much emphasis to the perceptual loss term and yielded less reliable results.

In Figures 5 and 6, we show the three phases associated to a prediction: Left, the ground truth; Middle: the masked spectrum/parameters; Right: the reconstructed ASPA. Figure 7 shows a water-dominated atmosphere of a test-set hot-Jupiter (black) and the ExoGAN reconstructed spectrum based on the *Hubble*/WFC3 bandpass only (red). We find a very good agreement between reconstructed and ground-truth spectra.

3. Atmospheric Parameter Retrieval

To retrieve the atmospheric forward model parameters, we assume the observational uncertainties on the spectrum to be Gaussian distributed. We then generate 1000 noisy instances of the observed spectrum, $x_i(\lambda)$, by sampling from a normal distribution with a mean of $x(\lambda)$ and standard deviation σ_λ. From these noisy spectrum instances, we generate 1000 corresponding ASPAs with missing information (may they be parameters, spectral ranges or both) masked. We now let ExoGAN predict and inpaint these ASPAs. Finally, we collect all parameter predictions and calculate the mean and standard deviation of the resulting distribution. Hence, the resulting distributions are not posterior distributions derived from a Nested or MCMC sampling atmospheric retrieval, but are conceptually more similar to running a retrieval based on optimal-estimation multiple times and collecting the distribution of results.

4. Accuracy Tests

We defined the accuracy of the retrieved parameter, A, as the function of the ground-truth parameter value, ϕ, the retrieved value, $\phi_{\rm recon}$, and its corresponding error σ_ϕ,

$$A(\phi, \sigma_\phi) = \frac{1}{N}\sum_i^N \frac{(\phi_{i,\rm recon} - \phi_i)^2}{\phi_i^2 + \sigma_{\phi_i}^2}. \tag{10}$$

where N is the number of reconstructed ASPA instances.

Table 2
ExoGAN Prediction Accuracies Associated to Each Parameters for the Training Set

Training Set Parameters			
Variable	$A(0\sigma_\phi)$	$A(1\sigma_\phi)$	$A(2\sigma_\phi)$
CO	64.4%	74.9%	80.8%
CO_2	93.7%	96.4%	97.3%
H_2O	86.3%	92.9%	94.8%
CH_4	80.3%	88.4%	91.9%
R_p	99.8%	99.8%	99.8%
M_p	88.8%	90.5%	91.6%
T_p	89.4%	91.9%	93.1%

Note. The $A(0\sigma_\phi)$ column represents the absolute accuracy of the prediction without taking into account the error bar of the retrieval. The second and third columns are taking into account the 1σ and 2σ retrieved errors following Equation (10), respectively.

Table 3
Same as Table 2 but for the Test Set

Test Set Parameters			
Variable	$A(0\sigma_\phi)$	$A(1\sigma_\phi)$	$A(2\sigma_\phi)$
CO	62.8%	72.6%	78.2%
CO_2	94.2%	96.6%	97.4%
H_2O	89.6%	92.8%	93.9%
CH_4	80.3%	88.2%	91.6%
R_p	100.0%	100.0%	100.0%
M_p	88.0%	89.7%	90.8%
T_p	90.4%	92.2%	93.2%

We compute the reconstruction accuracies for 1000 randomly selected planets for each, the test and training sets. The accuracies are summarized in Tables 2 and 3 for 0σ (an exact match), 1σ, and 2σ confidence intervals. Figure 8 shows an example of the parameter distributions retrieved for a test-case planet.

4.1. Comparison with a Classical Retrieval Model

In this section, we compare the ExoGAN results with a "classical" retrieval result obtained with the TauREx retrieval code. For this comparison and tests in subsequent sections, we used as example the hot-Jupiter HD 189733b with planetary/orbital parameters taken from Torres et al. (2008), Butler et al. (2006) and atmospheric chemistry based on Venot et al. (2012), see Table 4.

We now retrieve the forward model parameters for both TauREx and ExoGAN for spectra across the *Hubble*/WFC3 only band and a broad (0.3–15 μm) wavelength band. Here, the *Hubble*/WFC3 spectrum was taken from Tsiaras et al. (2018) and interpolated to the ExoGAN resolution using a quadratic interpolation (Figure 9). The large wavelength range spectrum is synthetic, based on Table 4.

In Figure 10 we compare both sets of results. The *Hubble*/WFC3 and large wavelength retrievals are shown with square and circular markers, respectively. In both cases, the ExoGAN predictions are consistent with the TauREx retrievals within the error bars. We note that in the case of CO in the *Hubble*/WFC3 data, neither TauREx nor ExoGAN feature detections as expected.

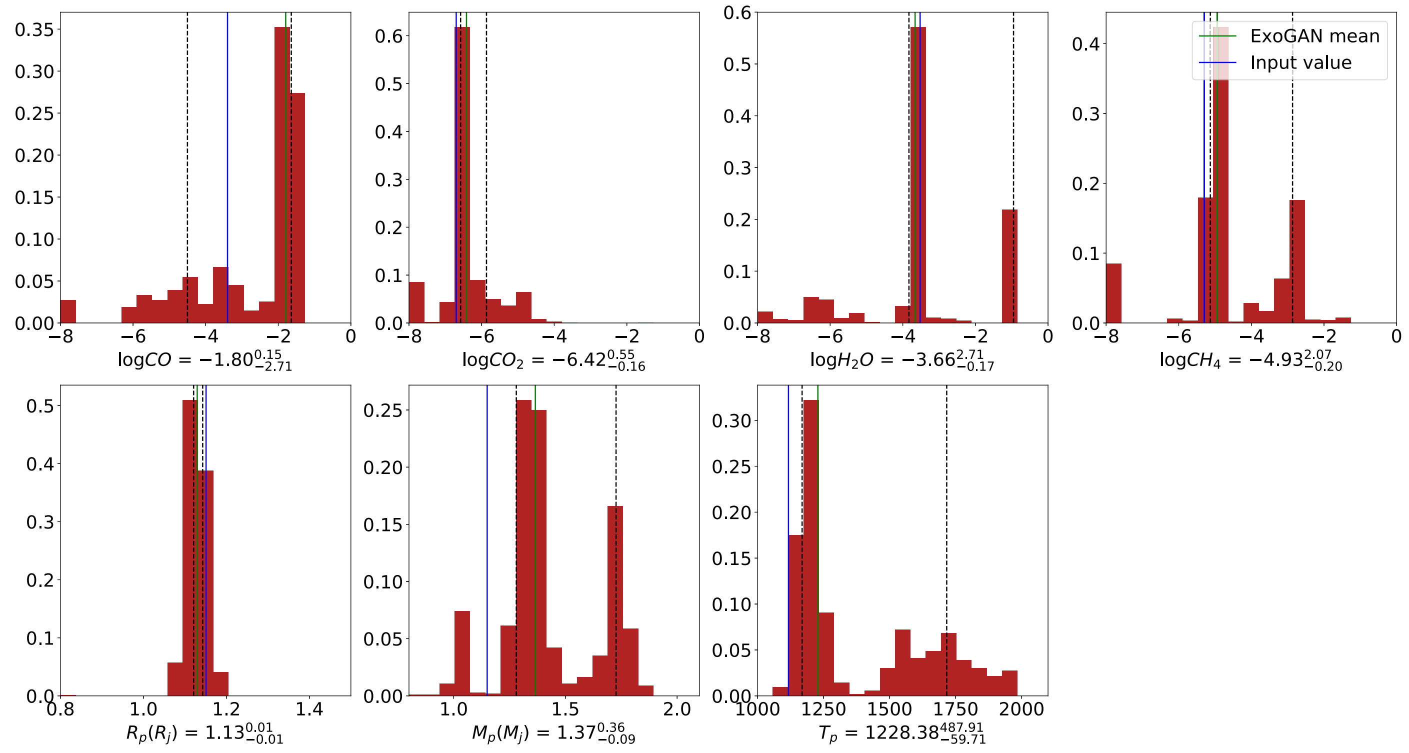

Figure 8. ExoGAN parameter distribution of the default test planet. Blue vertical line: Mean predicted value; green vertical line: ground truth value; vertical dotted lines: 1σ bounds estimated by ExoGAN.

Table 4
Test-case Atmospheric and Planetary Parameters Used Based on HD 189733b

Test Planet Parameters	
Parameter	Value
R_*	0.752 $R_\odot$
R_p	1.151 R_J
M_p	1.150 M_J
T_p	1117 K
H_2O	$3 \cdot 10^{-4}$
CO	$4 \cdot 10^{-4}$
CO_2	$2 \cdot 10^{-7}$
CH_4	$5 \cdot 10^{-6}$

Note. The molecular abundances are given in volume mixing ratios.

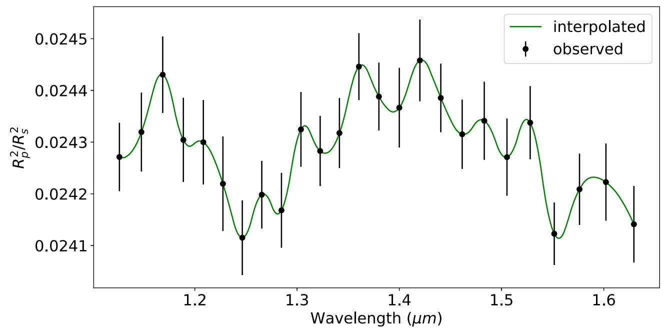

Figure 9. Real HD 189733b observation with the *Hubble* WFC3 camera (Tsiaras et al. 2018). The black points are the observed data and the green line is the interpolated spectrum to the ExoGAN resolution.

Table 5
Summary of All the Robustness Test Results

Robustness Results						
Variable	Clouds		Unknown Gases		T Offscale	
	Input	ExoGAN	Input	ExoGAN	Input	ExoGAN
log(CO)	−3.4	$-4.1^{3.1}_{2.5}$	<−8	$-5.7^{1.8}_{1.4}$	−3.4	$-3.1^{0.4}_{3.8}$
log(CO_2)	−6.7	$-6.0^{2.3}_{1.7}$	<−8	$-5.5^{3.9}_{1.8}$	−6.7	$-5.6^{4.4}_{0.2}$
log(H_2O)	−3.5	$-3.6^{1.1}_{3.0}$	−3.5	$-3.3^{0.7}_{3.5}$	−3.5	$-2.9^{0.2}_{4.1}$
log(CH_4)	−5.3	$-6.7^{1.6}_{1.1}$	<−8	$-5.5^{2.0}_{1.9}$	−5.3	$-5.1^{2.1}_{1.1}$
R_p (R_J)	1.15	$1.18^{0.01}_{0.01}$	1.15	$1.14^{0.01}_{0.01}$	1.15	$1.16^{0.02}_{0.01}$
M_p (M_J)	1.15	$1.23^{0.59}_{0.42}$	1.15	$1.39^{0.43}_{0.49}$	1.15	$1.60^{0.2}_{0.7}$
T_p (K)	1117	1681^{153}_{208}	1117	1689^{179}_{506}	2500	$1744^{157}_{6.4}$

Note. For each value we show the input value used for the spectrum and the predicted result from ExoGAN. For the unknown gases test, we used ammonia with a volume mixing ratio of 10^{-4}.

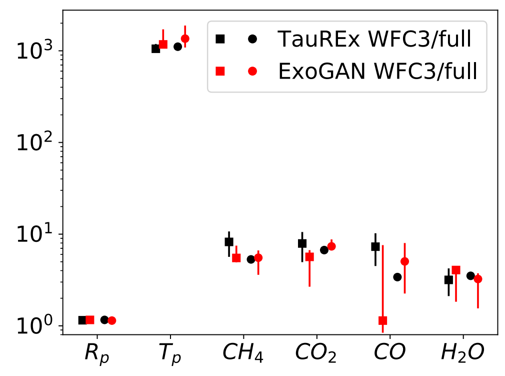

Figure 10. Comparison between the ExoGAN predictions (red points) and TauREx (black points). For the molecules, we show the value −log(mixing ratio). The squared points show the results for a real spectrum of HD 189733b using *Hubble*/WFC3. The round points are the results for a synthetic model of HD 189733b between 0.3 and 15 μm. The results from the two retrievals are in both cases consistent with each other within the error bars.

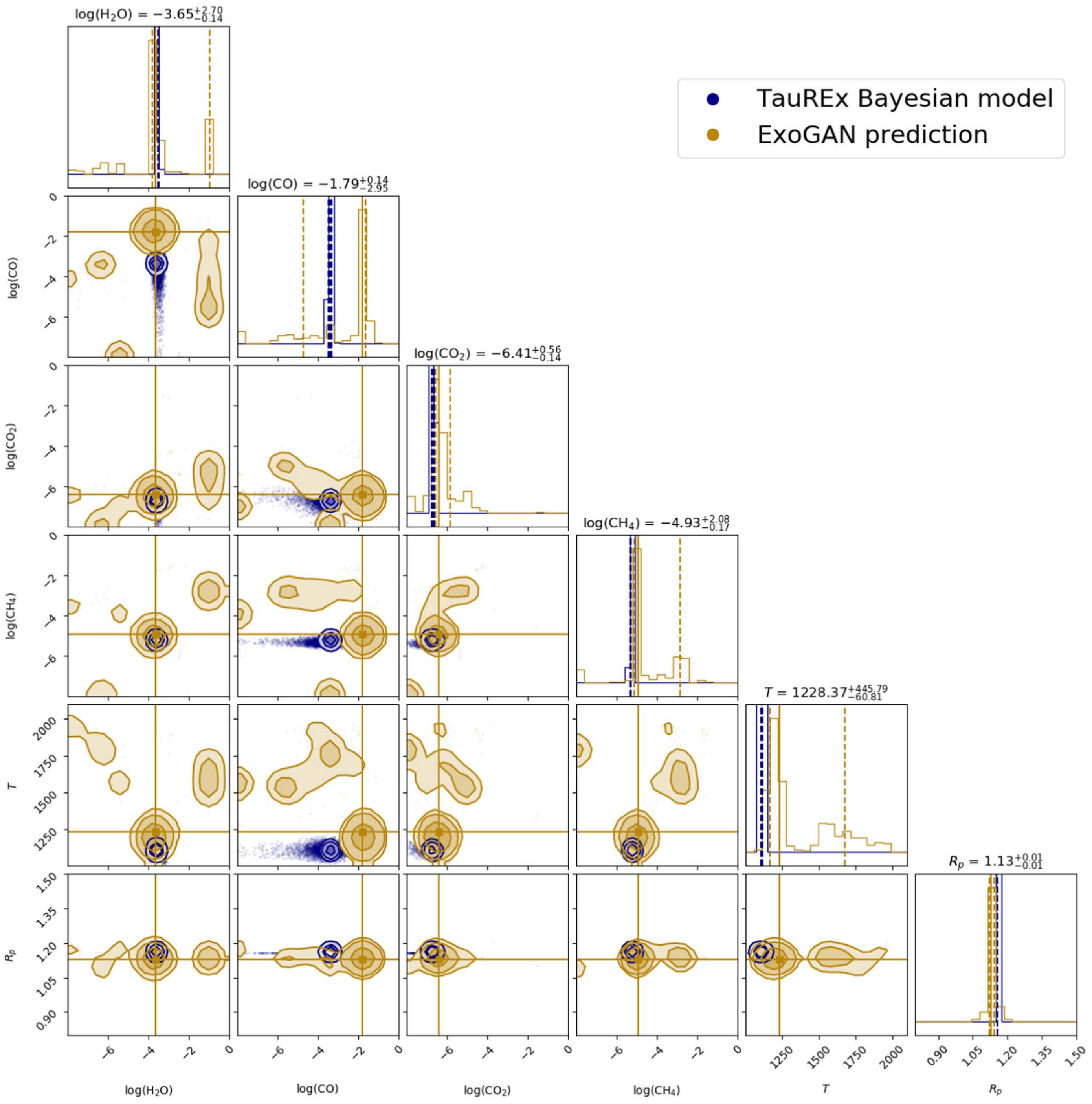

Figure 11. TauREx posterior distributions (in blue) compared to a ExoGAN prediction (in golden). As input spectrum, we used a synthetic spectrum of HD 189733b with planetary and atmospheric parameters from Venot et al. (2012) and a wavelength range of 0.3–15 μm. The two results are in agreement with each other.

We then generated a second synthetic spectrum of HD 189733b between 0.3 and 15 μm, using the parameters of Venot et al. (2012) and overplotted the TauREx retrieved posterior distributions with those derived by ExoGAN, Figure 11. Both algorithms converge to the same solution with the ExoGAN results showing a broader distribution. The only significant difference is the CO abundance, where the ExoGAN abundances are higher. Note that both TauREx and ExoGAN show tails in their CO abundance posteriors indicating the difficulties of retrieving CO even for classical retrieval algorithms.

Comparisons of runtime are remarkable. Using the TauREx Retrieval code with seven free parameters, a standard nested-sampling analysis takes ~10 hr on 24 CPU cores using absorption cross-sections at a resolution of $R = 15000$ and spanning a large (0.3–15 μm) wavelength range. The trained ExoGAN requires ~2 minutes for the same analysis. This result constitutes a speed up of ~300 times and is independent of the number of free parameters and of the resolution of the input spectrum. Similarly, training ExoGAN on higher-resolution data does not significantly impact its runtime after

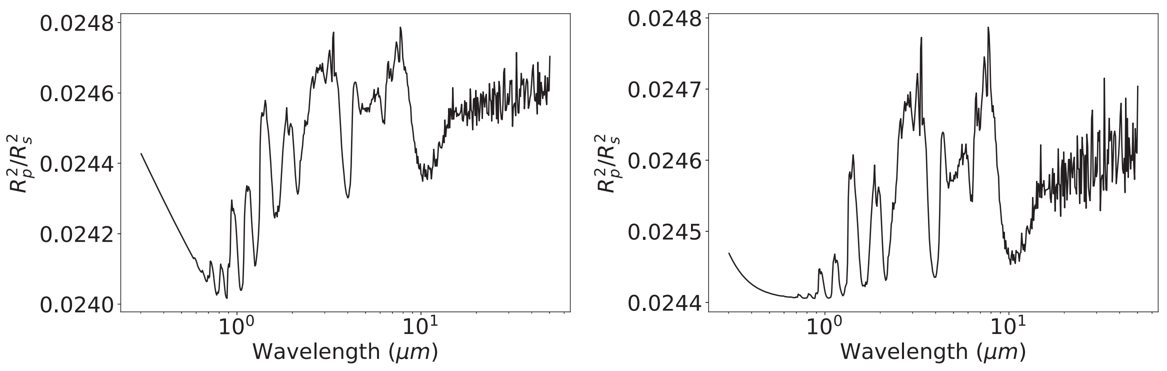

Figure 12. Simulated spectra of the default test planet HD 189733b without clouds (left) and with gray clouds at 10 mbar cloud top pressure (right).

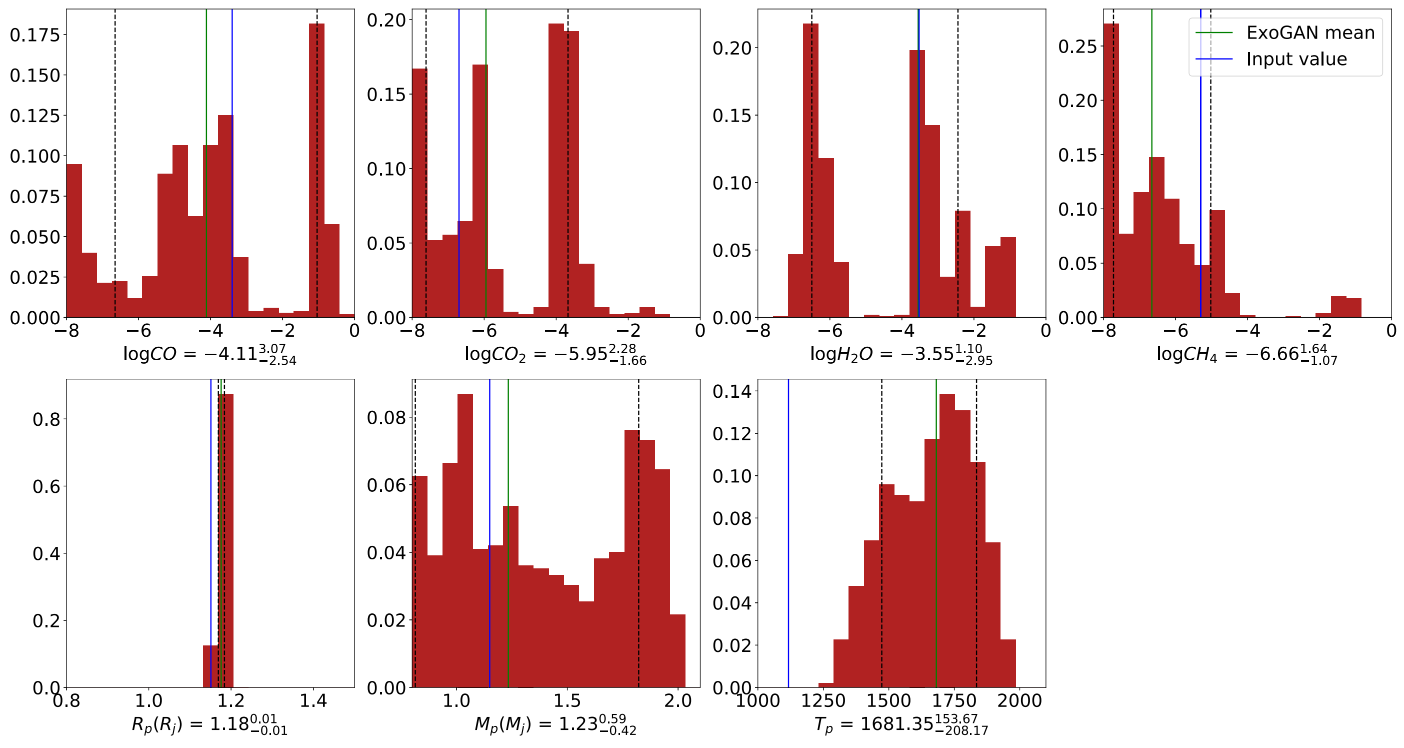

Figure 13. Same as Figure 8 but for the clouds' robustness test for the default test planet, Section 5.1.

training as both the size and architecture of the underlying network remain unchanged.

5. Robustness Tests

To test the limits of ExoGAN, we simulate three conditions previously encountered by the network. We use the same example planet as in the previous section (Table 4) and simulate the following three scenarios unseen by ExoGAN during training phase:

1. the presence of clouds;
2. the addition of a trace gas unknown to the network;
3. atmospheric temperatures outside the training range.

Each test is discussed below, and the ExoGAN predicted abundances versus the ground-truth are summarized in Table 5. Furthermore, we test the ExoGAN's robustness against varying signal-to-noise ratio (S/N) levels of the observed spectrum.

5.1. Presence of Clouds

Here we test the response of ExoGAN to the presence of clouds in the atmospheric spectrum. We simulate a gray cloud deck at 10 mbar pressure (Figure 12) and let ExoGAN reconstruct the atmospheric parameters (see Figure 13). The lack of information due to the clouds' presence results in a wider distribution of parameters. However, ExoGAN is still able to retrieve all trace-gas abundances within 1σ confidence. We find that temperature estimates can be overestimated. This result is likely a consequence of the normalization procedure used in the presence of clouds.

5.2. Presence of Molecules Outside of the Training Set

In this test, we simulate the impact of unknown features on the retrievability of known trace gases. We here consider a spectrum containing water at the default test value and NH_3 with a mixing ratio of 10^{-4}. Though Venot et al. (2012) estimated an NH_3 mixing ratio of 10^{-6}, we use an unrealistically high value

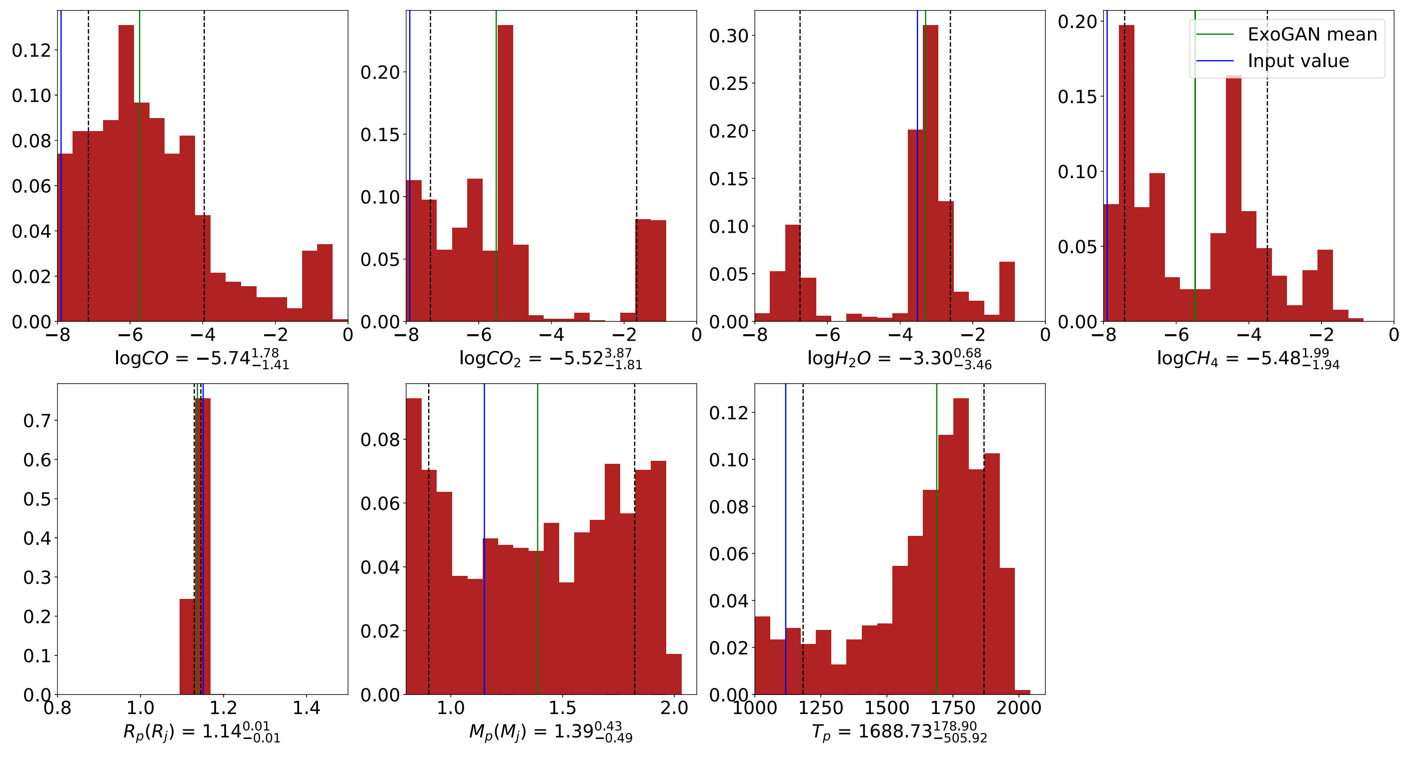

Figure 14. Same as Figure 8 but for the ExoGAN analysis for a spectrum with only water and NH_3, Section 5.2.

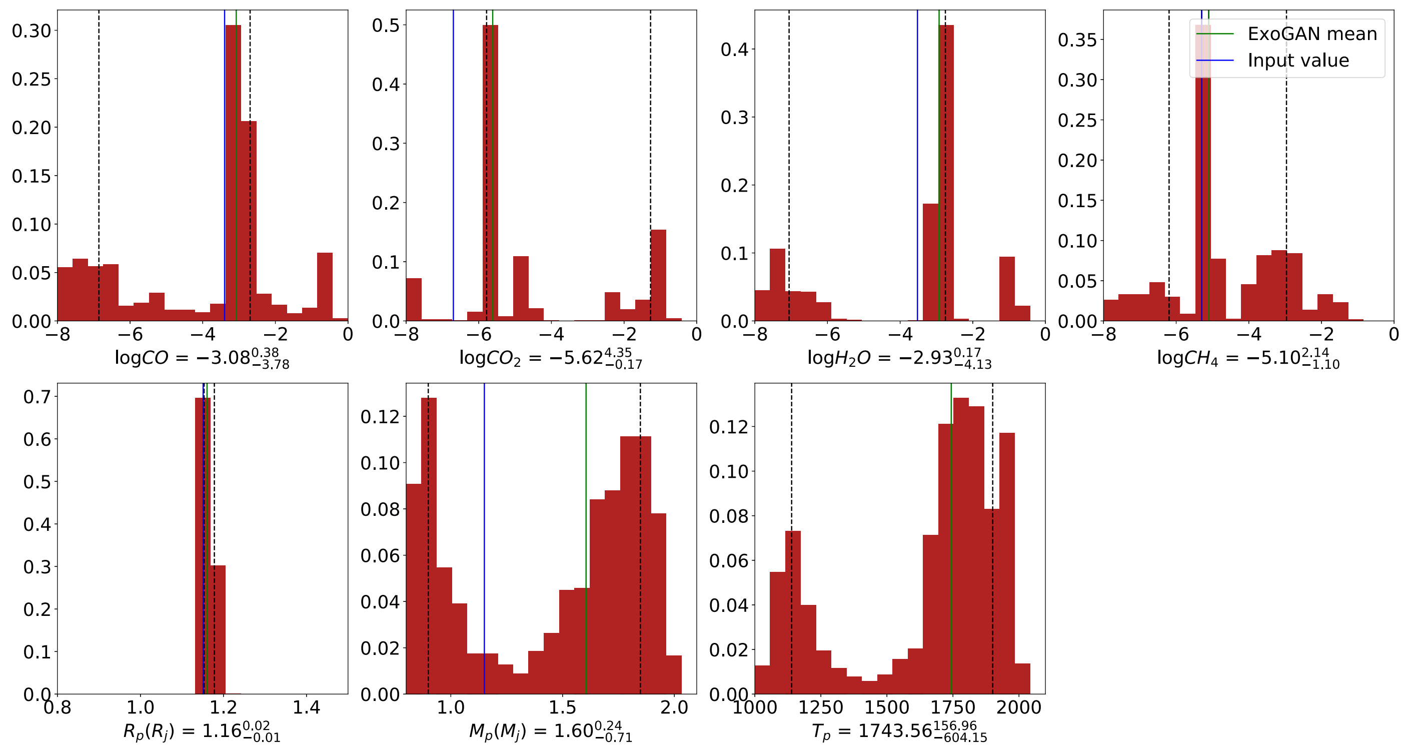

Figure 15. Same as Figure 8 but for the ExoGAN analysis for a planetary temperature at 2500 K—500 K outside the training range, see Section 5.3.

as a worst-case scenario. By removing all other trained trace gases but water, we also test for spurious detections in non-existing trace gases. Figure 14 shows the ExoGAN parameter distributions. We find the network to recognize the absence of trace gases, and it does not detect "false positives", while still recovering the exact mixing ratio of H_2O.

5.3. Parameters Outside the Training Range

In the third robustness test, we simulated a default planetary atmosphere but an effective temperature of 2500 K—500 K above the temperature training range. In this test, as shown in Figure 15, all parameters converge toward the real solution within 1σ, except for the planetary temperature. Here, the

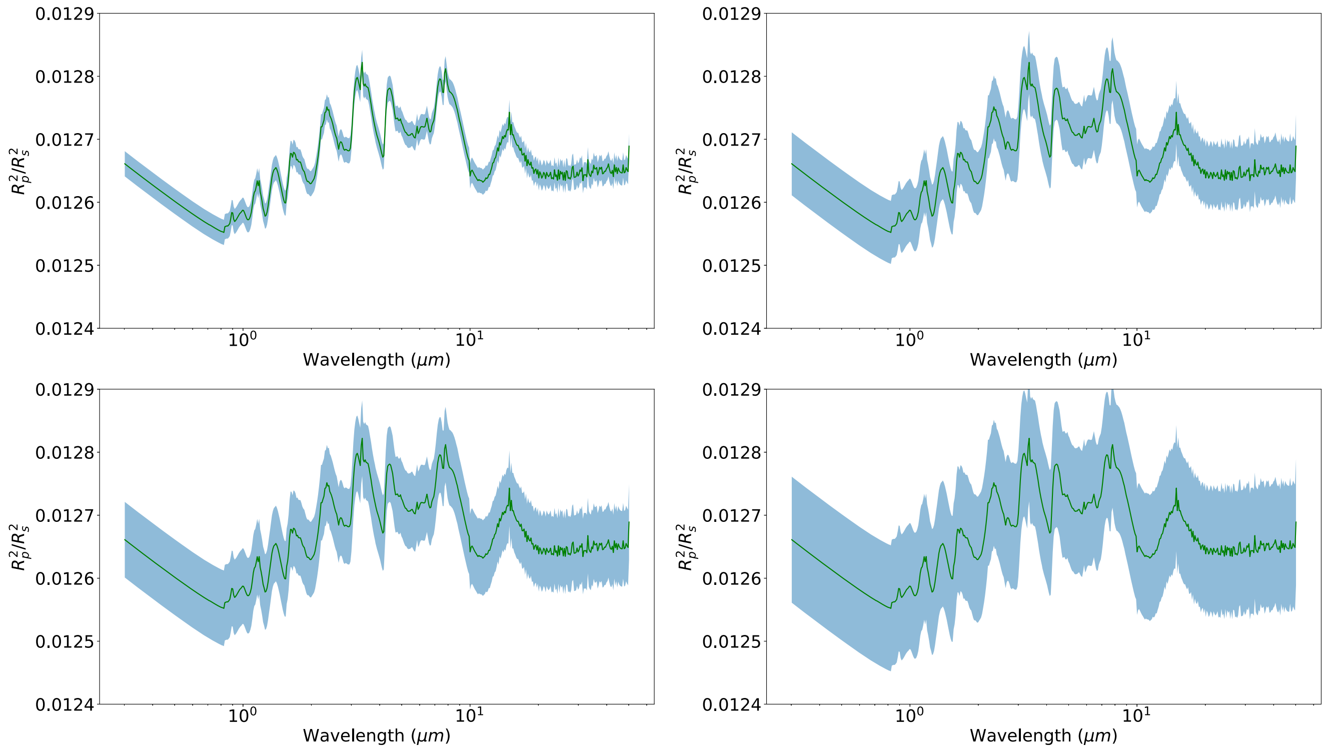

Figure 16. Four examples of spectra used to calculate the accuracy of the ExoGAN. The green line represents the input spectrum and the blue part is the area representing the error bars, σ_λ in which we varied the input signal to simulate a noisy spectrum. In the top left panel, we see the 20 ppm error bars, and in the top right panel, we see those for 50 ppm. In the bottom left panel we see the 60 ppm error bars, and in the bottom right, those for 100 ppm.

network does not retrieve the correct temperature but assigns a large error bar, suggesting that the temperature value is unconstrained if the input value is not contained in the domain range of ExoGAN.

5.4. Impact of Spectral Signal-to-noise

We test ExoGAN for varying levels of observational noise. Here we take the default planet (Table 4) and add noise in steps of 10 ppm in the range [0, 100] ppm. In Figure 16 we show examples of spectra at σ_λ: 20, 50, 60, and 100 ppm noise levels.

For each noise level, we calculated the accuracy of the prediction following Equation (10), but setting $A(\sigma_\phi = 0)$, Figure 17. We note that Figure 17 only shows the difference between the predicted value and an exact match, and prediction accuracies increase when retrieval error bars are taken into account. Here we want to demonstrate the relative degradation of the prediction accuracy as a function of σ_λ.

As intuitively expected, the noisier the spectra, the less accurate the model. The radius of the planet can be easily recognized by ExoGAN in the entire error range tested. The most difficult parameters to identify are the CO abundance and the mass of the planet.

6. Discussion

6.1. Training and Training Data

In this work, we used 10^7 forward models over seven atmospheric forward model parameters. We found that this training set is significantly over-complete and the ExoGAN training can be completed successfully with $\sim$50 % of the existing training set. Optimizing training in future iterations will allow for the inclusion of more complex atmospheric forward models.

One of the main difficulties for training neural networks with transmission spectra is the normalization of the spectra in R_p/R_*. A consistent normalization across a broad range of possible atmospheres is required during the training process, but difficult to achieve in reality given strongly varying atmospheric scale heights and trace-gas abundances. In this work, we adopted a normalization based on instrument pass-bands as well as water bands. Though in practice this approach works for most scenarios, it can introduce biases when high-altitude clouds are present. In these cases, we find that the normalization procedure stretches the observed spectrum too much, leading the network to identify higher atmospheric temperatures than it otherwise would. In future work, we plan to mitigate this effect by including gray clouds in the training set as well as further refining the normalization scheme. We note that for emission spectroscopy, a consistent normalization is more readily achieved if the planetary and stellar equilibrium temperatures are assumed to be known (Waldmann 2016).

ExoGAN has been trained on a large set of simulated forward models. By including ExoGAN as an integral part in the TauREx retrieval framework, we will be able to use forward models created during a standard retrieval run (of the order of 10^5–10^6 models per retrieval) to perform online learning and continuously improve the accuracy of ExoGAN over time.

6.2. Comparison with Other Machine-learning Architectures

In the previous sections, we have explored the use of DCGANs to retrieve atmospheric parameters from observations. GANs belong to the class of semi-supervised and unsupervised generative models, and since their inception, they

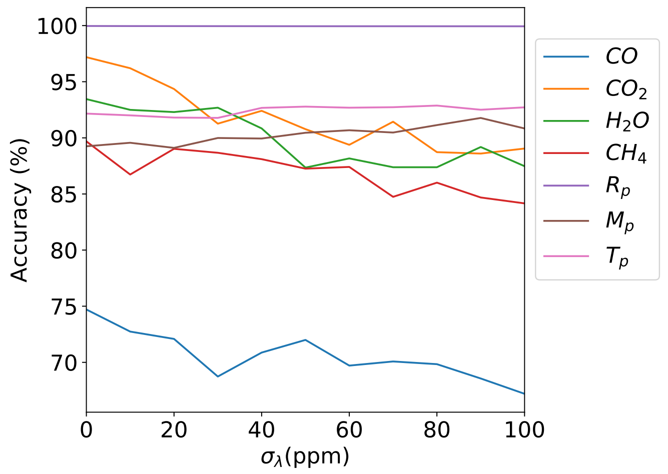

Figure 17. Accuracy as a function of spectral error bars, σ_λ. As discussed in the text, we note that this figure does not take into account the retrieval error bar, i.e., $A(\sigma_\phi = 0)$ following Equation (10).

have been subject to significant research. In this paper, our use of DCGAN is unsupervised as we provide the parameters together with the data to be modeled. Such an approach allows for a high degree of flexibility in using ExoGAN, as we only need to re-define the ASPA array to train on new problem sets.

Most other generative models require the likelihood function of the data to be defined, something we do not intrinsically know for many exoplanet observations, whereas GAN-based models are likelihood-free methods and $p_\theta(\boldsymbol{x})$ does not need to be computed during training. This characteristic has obvious advantages over pure variational autoencoders, which require a parameterized form of the probability space from which they draw their latent variables.

While we have explored the use of GANs in the scope of this paper, we note that other neural network architectures, such as simpler deep believe networks or VAEs, may yield comparable results. In fact, recent work by the 2018 NASA Frontier Development Lab[3] has explored various deep-learning architecture in the context of atmospheric retrievals with promising results. Similarly, other machine-learning frameworks may also be successfully used to model exoplanetary spectra. For example, Márquez-Neila et al. (2018) recently presented an atmospheric retrieval algorithm based on random forests regression (Breiman 2001) and demonstrated the algorithm on *Hubble*/WFC3 observations.

6.3. Future Work

In this work, we have used the "vanilla" DCGAN as underlying algorithm. Since its inception, various interesting additions to the classical GAN have been proposed, which we intend to explore in future work. Notable among them are the VAE-GAN hybrids, random forest and GAN hybrids, and Bayesian-GAN models. The VAE-GAN models (e.g., Makhzani et al. 2015; Dosovitskiy & Brox 2016; Rosca et al. 2017; Ulyanov et al. 2017) allow direct inference using GANs—something that is not possible using purely generative models. To further guard against model collapse, Zuo et al. (2018) have recently proposed a random forest and GAN hybrid algorithm, GAF, where the fully connected layer of the GAN's discriminator is replaced by a random forest classifier. Saatchi & Wilson (2017) proposed a Bayesian-GAN by drawing probability distributions over $\boldsymbol{\theta}^{(D)}$ and $\boldsymbol{\theta}^{(G)}$, allowing for fully Bayesian predictive models and further guarding against model collapse.

[3] frontierdevelopmentlab.org

7. Conclusion

In the era of *JWST* and ARIEL observations, next-generation atmospheric retrieval algorithms must reflect the higher information content of the observation with an increase in atmospheric model complexity. Complex models are computationally heavy, creating potential bottlenecks given current state-of-the-art sampling schemes. Artificial intelligence approaches will provide essential tools to mitigate the increase in computational burden while maintaining retrieval accuracies.

In this work, we introduced the first deep-learning approach to solving the inverse retrieval of exoplanetary atmospheres. We trained a deep convolutional generative adversarial network on an extensive library of atmospheric forward models and their associated model parameters. The training set spans a broad range of atmospheric chemistries and planet types. Once trained, the ExoGAN algorithm achieves comparable performances to more traditional statistical sampling based retrievals, and the ExoGAN results can be used to constrain the prior ranges of subsequent retrievals (to significantly cut computation times) or be used as stand-alone results. We found ExoGAN to be up to 300 times faster than a standard retrieval for large spectral ranges. ExoGAN is designed to be universally applicable to a wide range of instruments and wavelength ranges without additional training.

All codes used in this publication are open-access and their latest versions are hosted at https://github.com/orgs/ucl-exoplanets. Manuals and links to the training sets can be found at http://exoai.eu.

Furthermore, the training data and the corresponding ExoGAN software (at the time of paper acceptance) have been assigned the DOI:10.17605/OSF.IO/6DXPS and are permanently archived at https://osf.io/6dxps/. All data/software pertaining to the ExoAI project (inc. TauREx) is archived here: https://osf.io/tfyn6/.

The authors thank the anonymous referee for improving the clarity of the paper. The authors also thank the UCL Exoplanet team for insightful discussions. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programmes (grant agreement No. 758892/ExoAI and No. 776403/ExoplANETS A) and the European Union's Seventh Framework Programme (FP7/2007-2013) ERC grant agreement No. 617119/ExoLights. The authors further acknowledge funding from Microsoft Azure for Research and the STFC grants ST/K502406/1 and ST/P000282/1. T.Z. also acknowledges the contribution from INAF trough the "Progetti Premiali" funding scheme of the Italian Ministry of Education, University, and Research

Appendix A
ExoGAN Architecture and Parameters

ExoGAN is made up of two neural networks, the generator and the discriminator, whose parameters are shown in Table 6.

Table 6
Architecture of ExoGAN Listing the Hyperparameters $\boldsymbol{\theta}^{(D)}$ and $\boldsymbol{\theta}^{(G)}$

Layer	Operation	Output	Dimension
Discriminator ($\boldsymbol{\theta}^{(D)}$)			
$\boldsymbol{X}$			$m \cdot 33 \cdot 33 \cdot 1$
h_0	conv	leaky relu—batch norm	$m \cdot 17 \cdot 17 \cdot 64$
h_1	conv	leaky relu—batch norm	$m \cdot 9 \cdot 9 \cdot 128$
h_2	conv	leaky relu—batch norm	$m \cdot 5 \cdot 5 \cdot 256$
h_3	conv	leaky relu—batch norm	$m \cdot 3 \cdot 3 \cdot 512$
h_4	linear	sigmoid	$m \cdot 1$
Generator ($\boldsymbol{\theta}^{(G)}$)			
z			$m \cdot 100$
h_0	linear	relu—batch norm	$m \cdot 3 \cdot 3 \cdot 512$
h_1	deconv	relu—batch norm	$m \cdot 5 \cdot 5 \cdot 256$
h_2	deconv	relu—batch norm	$m \cdot 9 \cdot 9 \cdot 128$
h_3	deconv	relu—batch norm	$m \cdot 17 \cdot 17 \cdot 64$
h_4	deconv	sigmoid	$m \cdot 33 \cdot 33 \cdot 1$

Note. We used five-layer deep networks for both generator and discriminators. m is the batch size fixed to 64 during training.

Appendix B Batch Normalization

A characteristic of DCGANs is the use of batch normalization (BN; Ioffe & Szegedy 2015; Xiang & Li 2017). BN is now a common technique in deep-learning applications to accelerate the training of neural networks. DCGAN networks (Radford et al. 2015) use BN for both the Discriminator and the Generator nets. Nevertheless, GAN architectures started using BN just for the generator net with the LAPGAN networks (Denton et al. 2015). Nowadays, many GAN architectures use BN. The idea behind BN is using a batch of samples $\{x_1, x_2, \ldots, x_m\}$ and computing, keeping the notation of Xiang & Li (2017), the following:

$$y_i = \frac{x_i - \mu_B}{\sigma_B} \cdot \gamma + \beta, \tag{11}$$

where μ_B and σ_B are, respectively, the mean and the standard deviation of the batch, and γ and β are the learned parameters. BN allows one to have an output with a mean μ and a standard deviation σ independent of the input distribution.

Appendix C The ADAM Optimizer

The Adaptive Moment Estimation (ADAM) is a very popular algorithm in deep learning and it computes an adaptive learning rate for the parameters of a neural network. It stores the exponentially decaying average of past squared gradients v_t together with the exponentially decaying average of the past gradients m_t. Keeping the notation of Ruder (2016), the past and the squared past gradients, m_t and v_t are defined as

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \tag{12}$$

and,

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2, \tag{13}$$

with β_1 and β_2 being the decay rates, and $g_t = \nabla_z \mathcal{L}(z_t)$ the gradient of the $\mathcal{L}$ function defined in Equation (7)

Equations (12) and (13) estimate, respectively, the mean (or first moment) and the variance (or second moment) of the gradients. Since the two moments are initialized as vectors of 0's, they are biased toward zero, particularly during the first time steps or when the decay rates β_1 and β_2 are small. To correct the biases, Kingma & Ba (2014) defined the bias-corrected moments as:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \tag{14}$$

and,

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}. \tag{15}$$

At this point, it is possible to update the z variable using the Adam update rule:

$$z_{t+1} = z_t - \frac{\eta}{\sqrt{\hat{v}_t} + \varepsilon} \hat{m}_t. \tag{16}$$

We used the values suggested by Kingma & Ba (2014) for the hyperparameters, shown in Table 7.

Table 7
Hyperparameters Used in ExoGAN

Hyper-parameter	*Stage*		Description
	Training	Prediction	
batch size	64	1024	Number of spectral samples used at each training/prediction iteration for both networks
z	100	100	Generator Gaussian prior distribution
η	$2 \cdot 10^{-4}$	$1 \cdot 10^{-1}$	Learning rate for the Adam optimizer
β_1	0.5	0.9	Exponential decay rate for the first moment estimates in the Adam optimizer.
β_2	⋯	0.999	Exponential decay rate for the second moment estimates in the Adam optimizer.
λ	⋯	0.1	Hyper-parameter that controls the importance of the contextual loss compared to the perceptual loss
ε	⋯	10^{-8}	Constant that prevents the denominator in Equation (16) to be zero

ORCID iDs

Tiziano Zingales https://orcid.org/0000-0001-6880-5356
Ingo P. Waldmann https://orcid.org/0000-0002-4205-5267

References

Amos, B. 2016, Image Completion with Deep Learning in TensorFlow, http://bamos.github.io/2016/08/09/deep-completion, accessed: 12/05/2018
Arjovsky, M., & Bottou, L. 2017, arXiv:1701.04862
Barstow, J. K., Aigrain, S., Irwin, P. G. J., & Sing, D. K. 2017, ApJ, 834, 50
Bengio, Y. 2009, Learning Deep Architectures for AI, Vol. 2 (Hanover, MA: Now Publishers Inc.)
Benneke, B., & Seager, S. 2013, APJ, 778, 153
Breiman, L. 2001, Machine Learning, 45, 5
Bruno, G., Lewis, N. K., Stevenson, K. B., et al. 2018, AJ, 155, 55
Butler, R. P., Wright, J. T., Marcy, G. W., et al. 2006, ApJ, 646, 505
Cubillos, P., Blecic, J., Harrington, J., et al. 2016, BART: Bayesian Atmospheric Radiative Transfer Fitting Code, Astrophysics Source Code Library, ascl:1608.004
Denton, E., Chintala, S., Szlam, A., & Fergus, R. 2015, arXiv:1506.05751
Dosovitskiy, A., & Brox, T. 2016, arXiv:1602.02644
Feroz, F., Gair, J. R., Hobson, M. P., & Porter, E. K. 2009, CQGra, 26, 215003
Feroz, F., & Hobson, M. P. 2008, MNRAS, 384, 449
Gandhi, S., & Madhusudhan, N. 2018, MNRAS, 474, 271
Gardner, J. P., Mather, J. C., Clampin, M., et al. 2006, SSRv, 123, 485
Goodfellow, I., Bengio, Y., & Courville, A. 2016, Deep Learning (Cambridge, MA: MIT Press)
Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., et al. 2014, arXiv:1406.2661
Gregory, P. C. 2011, MNRAS, 410, 94
Ioffe, S., & Szegedy, C. 2015, arXiv:1502.03167
Irwin, P. G. J., Teanby, N. A., De Kok, R., et al. 2008, JQSRT, 109, 1136
Jimenez Rezende, D., Mohamed, S., & Wierstra, D. 2014, arXiv:1401.4082
Kalirai, J. 2018, ConPh, 59, 251
Kingma, D. P., & Ba, J. 2014, arXiv:1412.6980
Kingma, D. P., & Welling, M. 2013, arXiv:1312.6114
Kipping, D. M., & Lam, C. 2017, MNRAS, 465, 3495
Kreidberg, L., Line, M. R., Parmentier, V., et al. 2018, arXiv:1805.00029
Lamb, A., Dumoulin, V., & Courville, A. 2016, arXiv:1602.03220
Lavie, B., Mendonça, J. M., Mordasini, C., et al. 2017, AJ, 154, 91
Line, M. R., Wolf, A. S., Zhang, X., et al. 2013, ApJ, 775, 137
Madhusudhan, N., & Seager, S. 2009, ApJ, 707, 24
Makhzani, A., Shlens, J., Jaitly, N., Goodfellow, I., & Frey, B. 2015, arXiv:1511.05644
Mansfield, M., Bean, J. L., Line, M. R., et al. 2018, AJ, 156, 10
Márquez-Neila, P., Fisher, C., Sznitman, R., & Heng, K. 2018, NatAs, 2, 719
Metz, L., Poole, B., Pfau, D., & Sohl-Dickstein, J. 2016, arXiv:1611.02163
Pearson, K. A., Palafox, L., & Griffith, C. A. 2018, MNRAS, 474, 478
Radford, A., Metz, L., & Chintala, S. 2015, arXiv:1511.06434
Rocchetto, M., Waldmann, I. P., Venot, O., Lagage, P. O., & Tinetti, G. 2016, ApJ, 833, 120
Rodriguez, A. C., Kacprzak, T., Lucchi, A., et al. 2018, arXiv:1801.09070
Rosca, M., Lakshminarayanan, B., Warde-Farley, D., & Mohamed, S. 2017, arXiv:1706.04987
Ruder, S. 2016, arXiv:1609.04747
Rumelhart, D. E., Hinton, G. E., & Williams, R. J. 1986, Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1 (Cambridge, MA: MIT Press)
Saatchi, Y., & Wilson, A. G. 2017, arXiv:1705.09558
Salimans, T., Goodfellow, I., Zaremba, W., et al. 2016, arXiv:1606.03498
Schawinski, K., Zhang, C., Zhang, H., Fowler, L., & Santhanam, G. K. 2017, MNRAS: Letters, 467, L110
Shallue, C. J., & Vanderburg, A. 2018, AJ, 155, 94
Sheppard, K. B., Mandell, A. M., Tamburo, P., et al. 2017, ApJL, 850, L32
Skilling, J. 2004, in XXIV Int. Workshop on Bayesian Inference and Maximum Entropy Methods in Science and Engineering 735, ed. R. Fischer, R. Preuss, & U. von Toussaint (Melville, NY: AIP), 395
Spake, J. J., Sing, D. K., Evans, T. M., et al. 2018, Natur, 557, 68
Stark, D., Launet, B., Schawinski, K., et al. 2018, MNRAS, 477, 2513
Tinetti, G., Drossart, P., Eccleston, P., et al. 2016, Proc. SPIE, 9904, 99041X
Torres, G., Winn, J. N., & Holman, M. J. 2008, ApJ, 677, 1324
Tsiaras, A., Waldmann, I. P., Zingales, T., et al. 2018, AJ, 155, 156
Ulyanov, D., Vedaldi, A., & Lempitsky, V. 2017, arXiv:1704.02304
Venot, O., Hébrard, E., Agúndez, M., et al. 2012, A&A, 546, A43
Waldmann, I. P. 2016, ApJ, 820, 107
Waldmann, I. P., Rocchetto, M., Tinetti, G., et al. 2015a, ApJ, 813, 13
Waldmann, I. P., Tinetti, G., Rocchetto, M., et al. 2015b, ApJ, 802, 107
Wu, Y., Schuster, M., Chen, Z., et al. 2016, arXiv:1609.08144
Xiang, S., & Li, H. 2017, arXiv:1704.03971
Yeh, R. A., Chen, C., Yian Lim, T., et al. 2016, arXiv:1607.07539
Zuo, Y., Avraham, G., & Drummond, T. 2018, arXiv:1805.05185