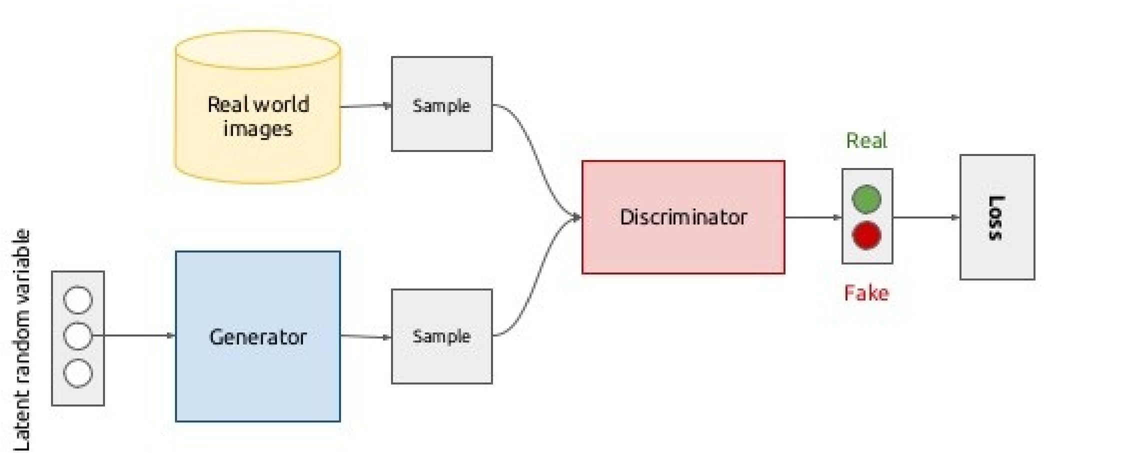
Generative Adversarial Networks for Cosmology Mass Maps
(Simulation Emulation)

Mustafa Mustafa
Evan Racah, Rami Alrfou

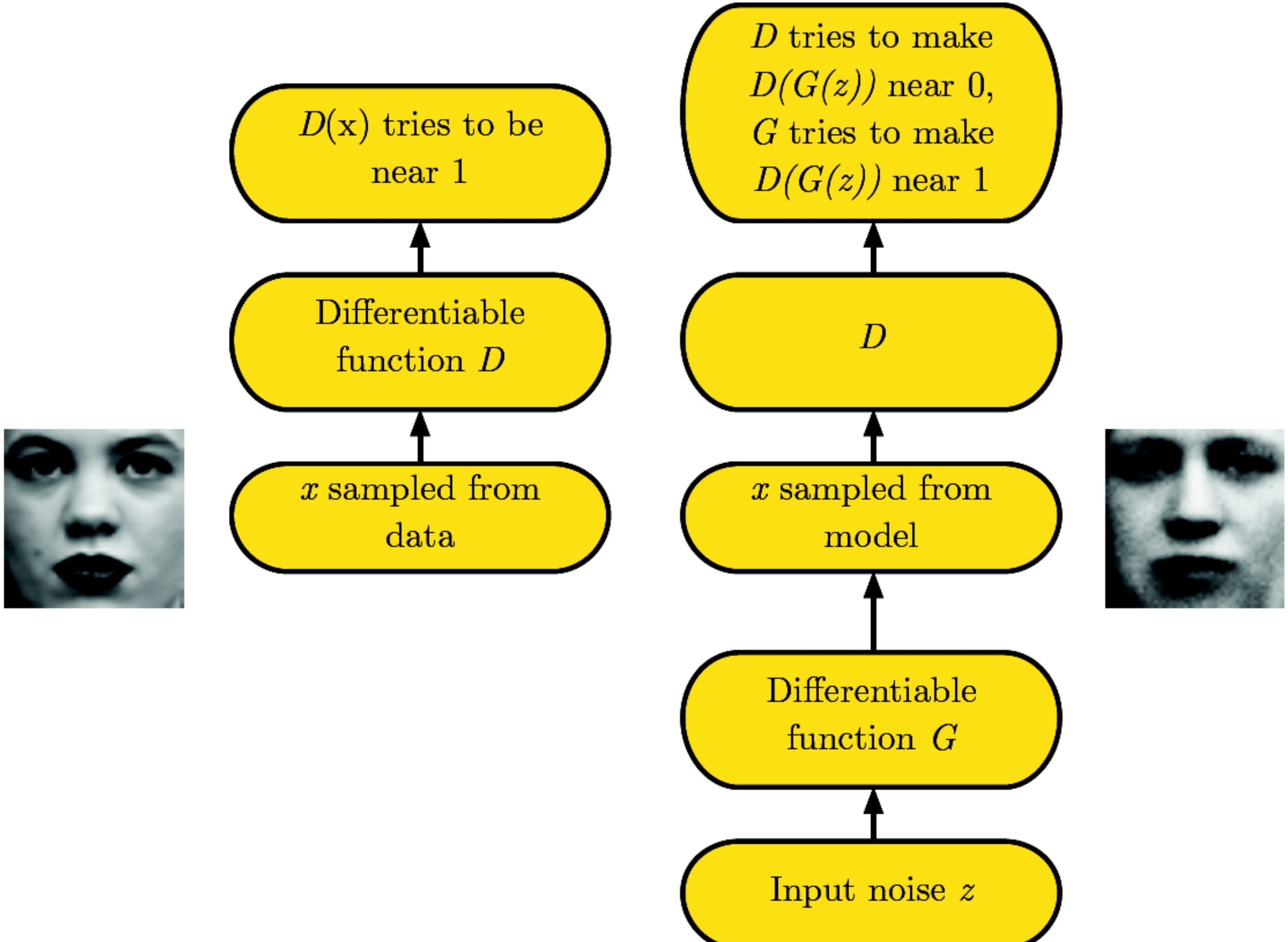
MANTISSA, Berkeley Lab.
01/17/2017

Generative Adversarial Networks



Kevin McGuinness

Generative Adversarial Networks



(Goodfellow 2016)

Generative Adversarial Networks – Loss Functions

Saturating game (Minimax):

$$J^{(D)} = -\frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}} \log D(\boldsymbol{x}) - \frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log\left(1 - D\left(G(\boldsymbol{z})\right)\right)$$

$$J^{(G)} = -J^{(D)}$$

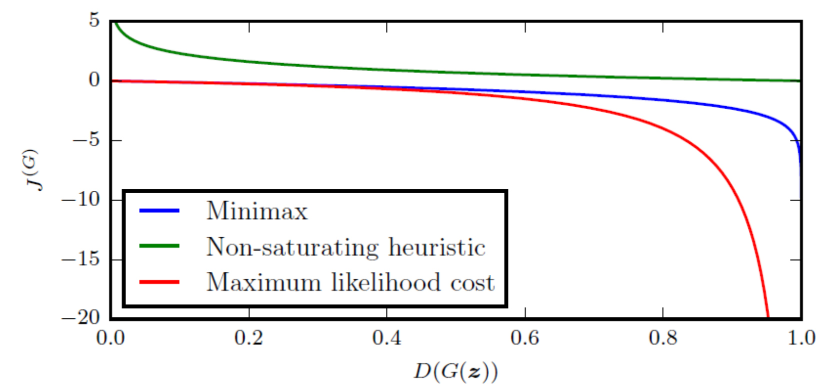
Ian Goodfellow arXiv:1701.00160

Generative Adversarial Networks – Loss Functions

Saturating game (Minimax):

$$J^{(D)} = -\frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}} \log D(\boldsymbol{x}) - \frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log\left(1 - D\left(G(\boldsymbol{z})\right)\right)$$

$$J^{(G)} = -J^{(D)}$$



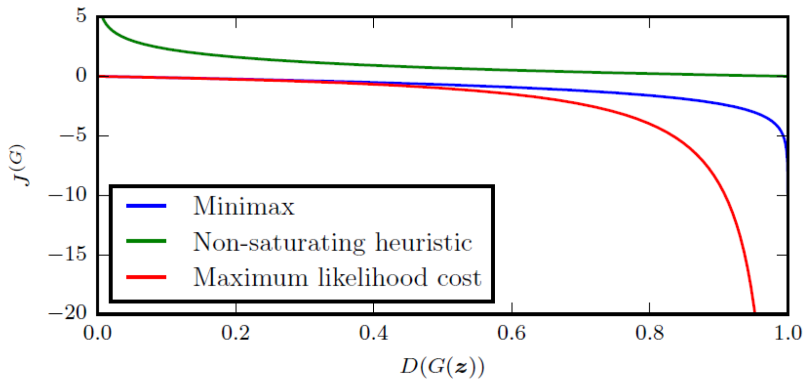
Ian Goodfellow arXiv:1701.00160

Generative Adversarial Networks – Loss Functions

Saturating game (Minimax):

$$J^{(D)} = -\frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}} \log D(\boldsymbol{x}) - \frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log\left(1 - D\left(G(\boldsymbol{z})\right)\right)$$

$$J^{(G)} = -J^{(D)}$$

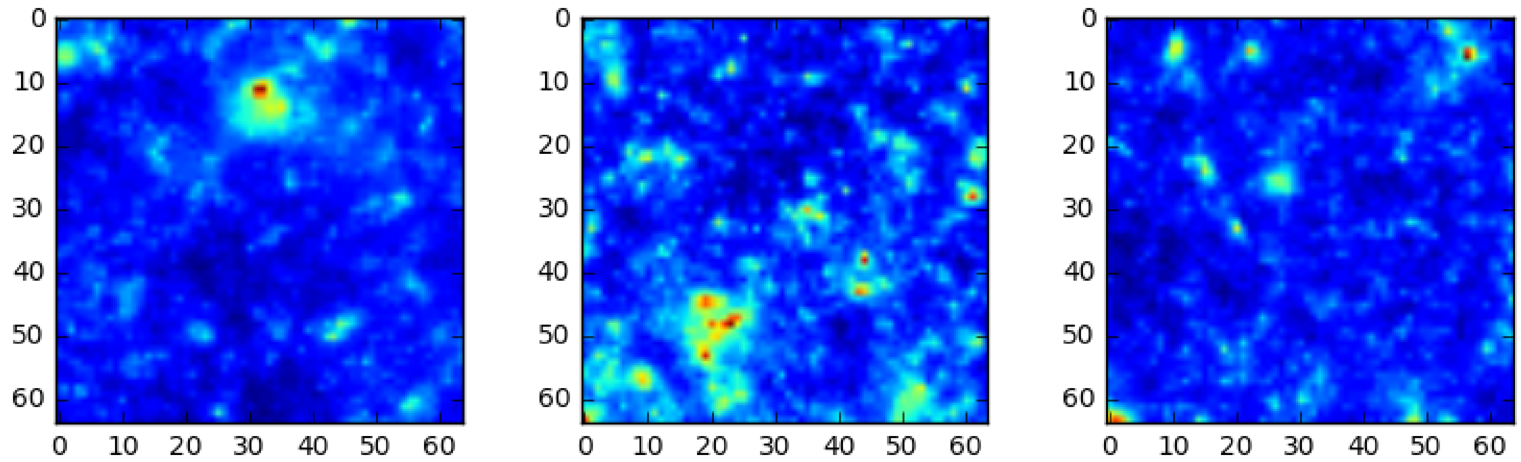


Non-saturating game (heuristic):

$$J^{(G)} = -\frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log D\left(G(\boldsymbol{z})\right)$$

Ian Goodfellow arXiv:1701.00160

Cosmology Mass Maps Simulator Emulator



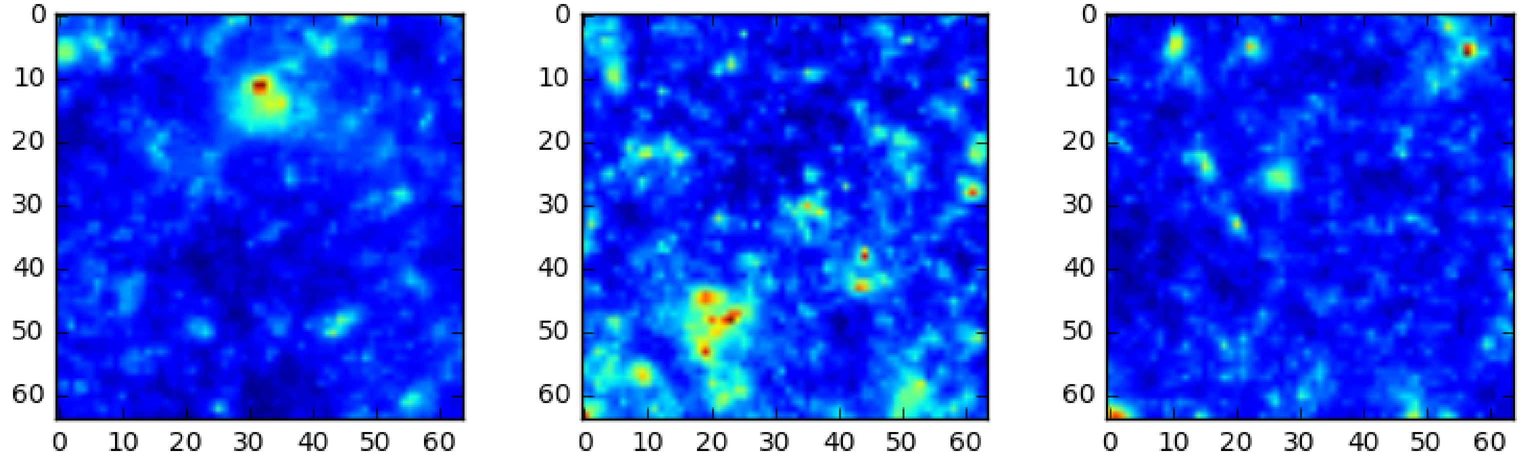
Basic idea:

Cosmologist need to run computationally expensive simulations of the mass density maps of the universe with different parameters σ = (σ1, σ2, …). The evolution of the universe is not deterministic, i.e. you can get “different” mass maps for the same set of parameters σ*.

We want to explore if we can use GANs to help in reducing the computational time. A reliable GAN duet might also be used to extract features or summary statistics.

<u>The fidelity of the generated images can be checked using a cosmologist metric (“summary statistics”).</u>

Cosmology Mass Maps Simulator Emulator



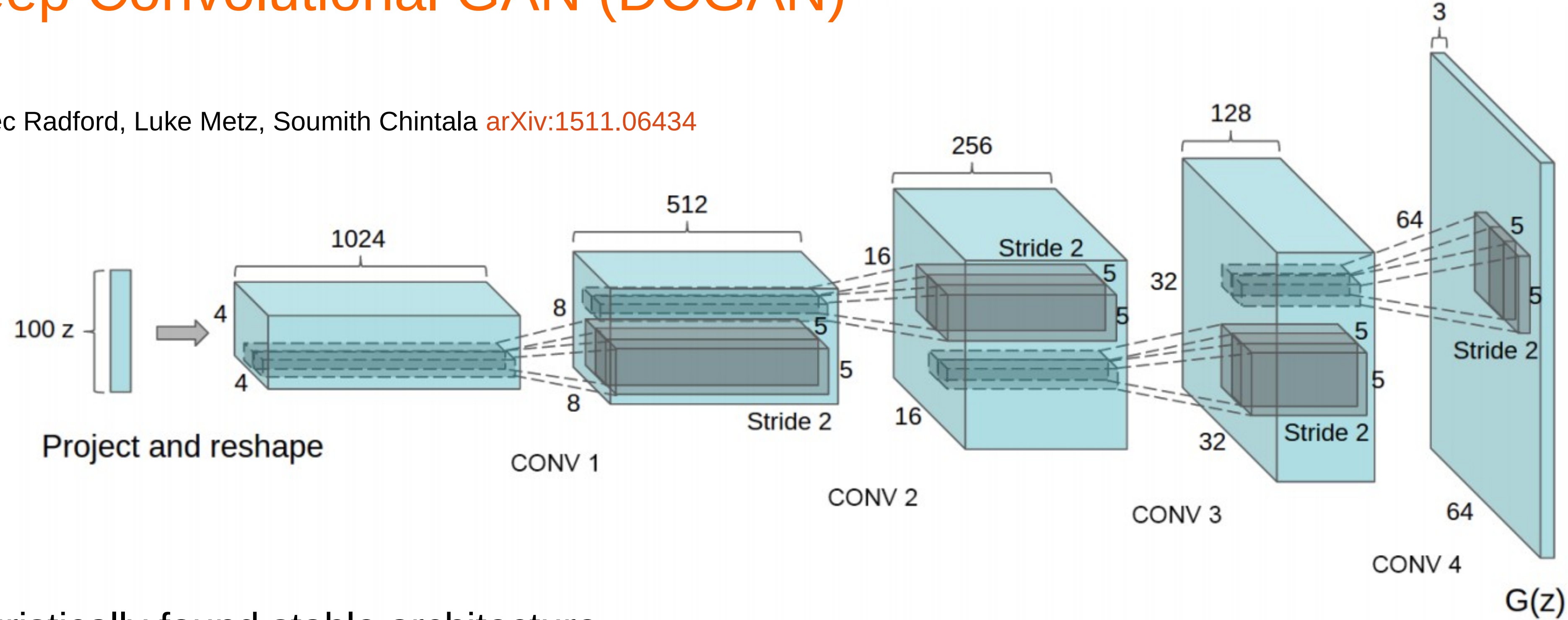
Dataset:

1000 1024x1024 mass maps generated at one σ* point. It is possible to generate more if needed.

- **Ultimate goal:** a conditional/parametric generator G(σ, z), where σ is the cosmologists vector of parameters and z is a vector of random noise
- **Current goal:** G(z) which will generate images at the fixed point σ* of our test sample

Deep Convolutional GAN (DCGAN)

Alec Radford, Luke Metz, Soumith Chintala arXiv:1511.06434

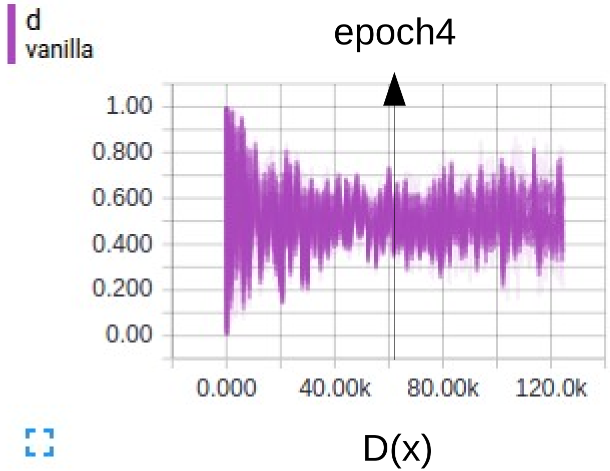


Heuristically found stable architecture

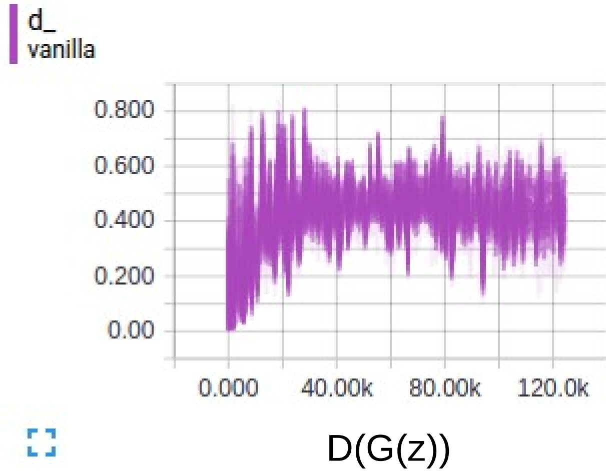
Architecture guidelines for stable Deep Convolutional GANs

- Replace any pooling layers with strided convolutions (discriminator) and fractional-strided convolutions (generator).
- Use batchnorm in both the generator and the discriminator.
- Remove fully connected hidden layers for deeper architectures.
- Use ReLU activation in generator for all layers except for the output, which uses Tanh.
- Use LeakyReLU activation in the discriminator for all layers.

Cosmo DCGAN 64x64



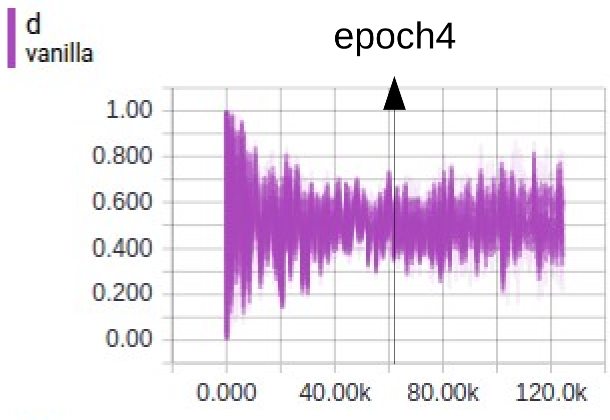
D(x)



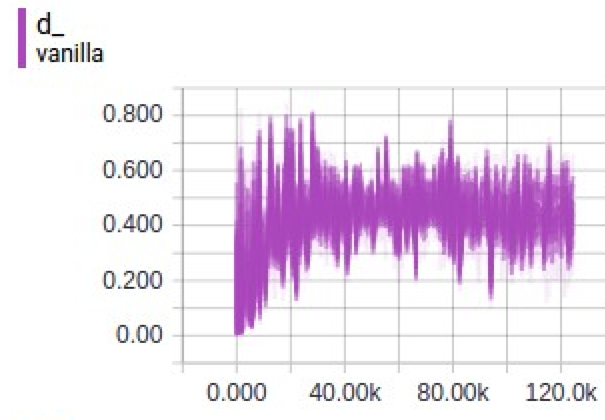
D(G(z))

- Trained on 1M 64x64 images (float32)
- Total of 8 epochs with evaluations at 4 and 8

Cosmo DCGAN 64x64

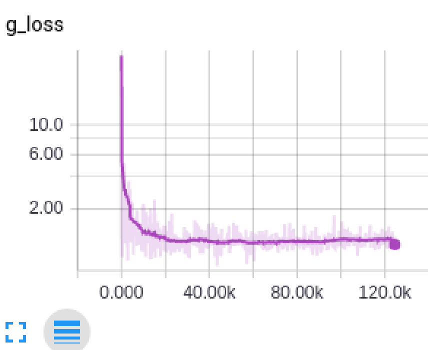


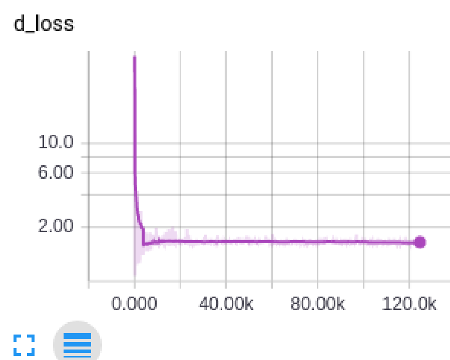
D(x)

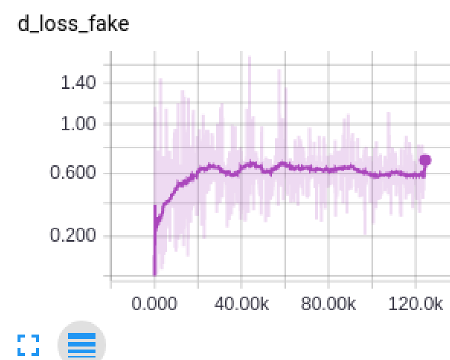


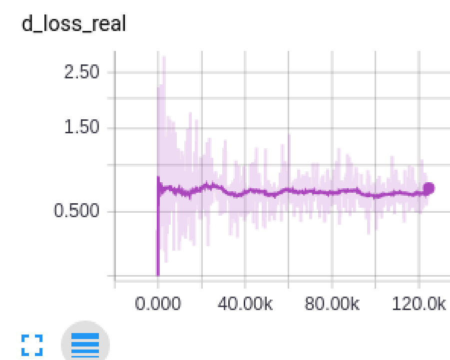
D(G(z))

- Trained on 1M 64x64 images (float32)
- Total of 8 epochs with evaluations at 4 and 8

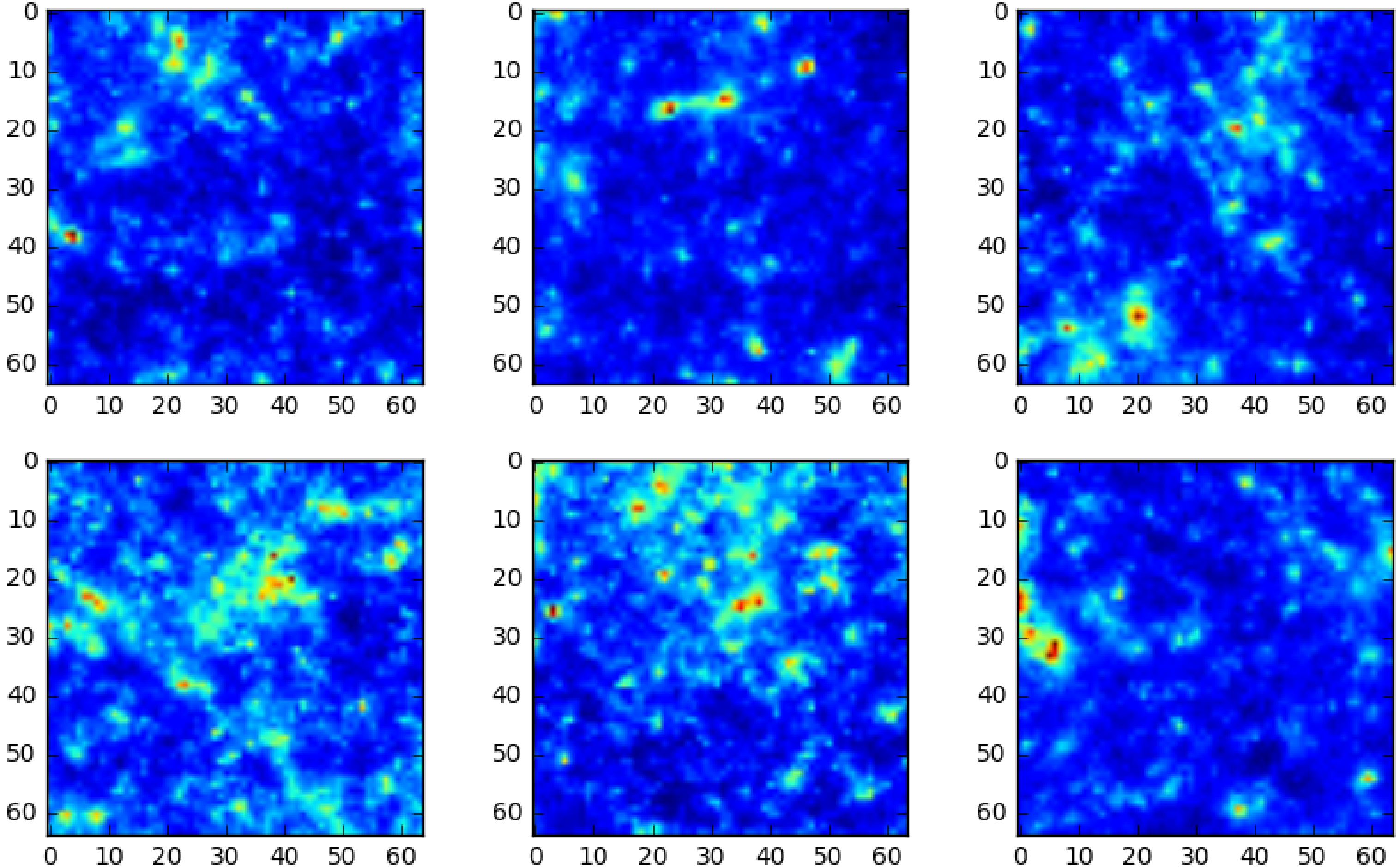






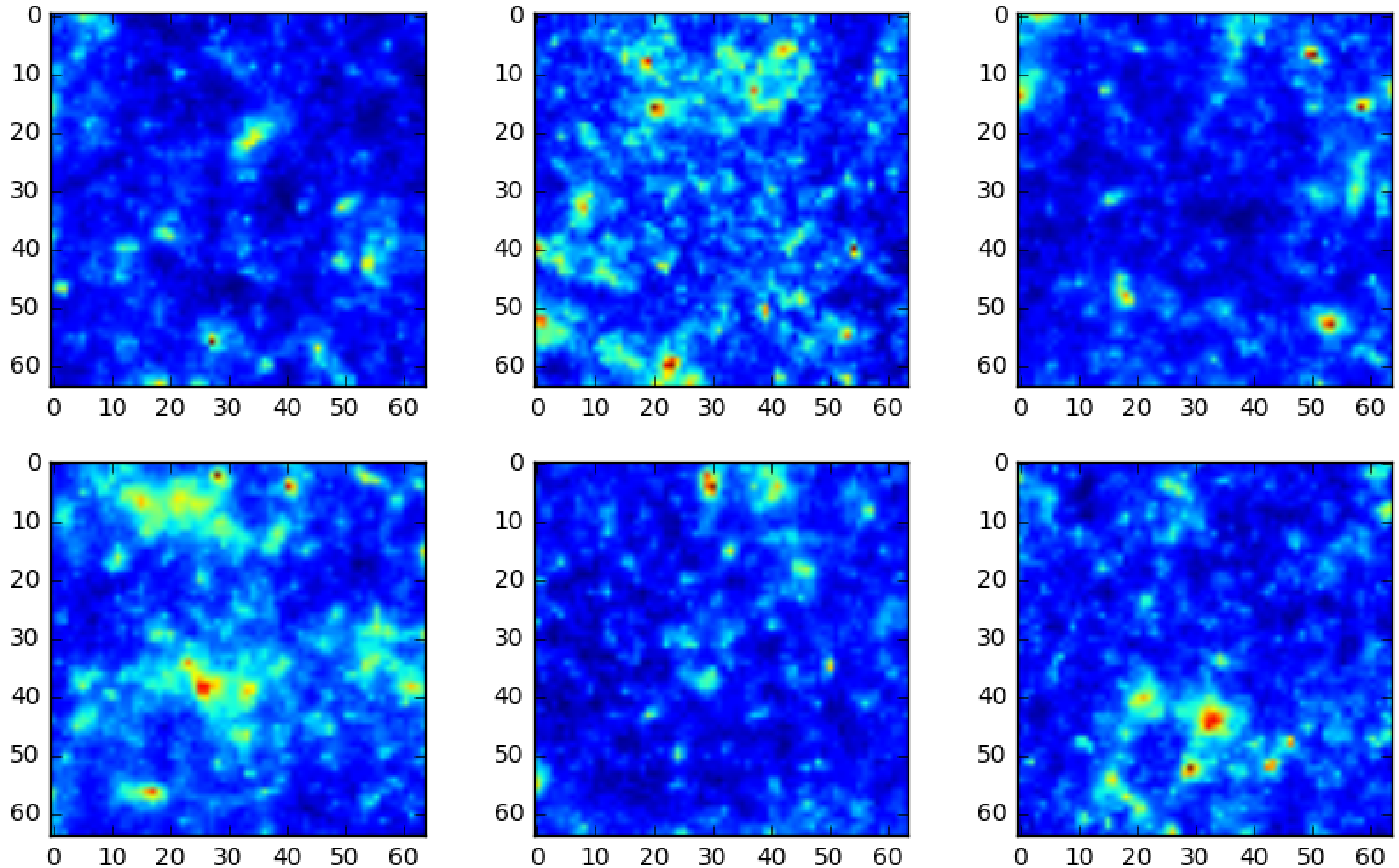


Cosmo DCGAN 64x64



Epoch4: Generated Images

Cosmo DCGAN 64x64

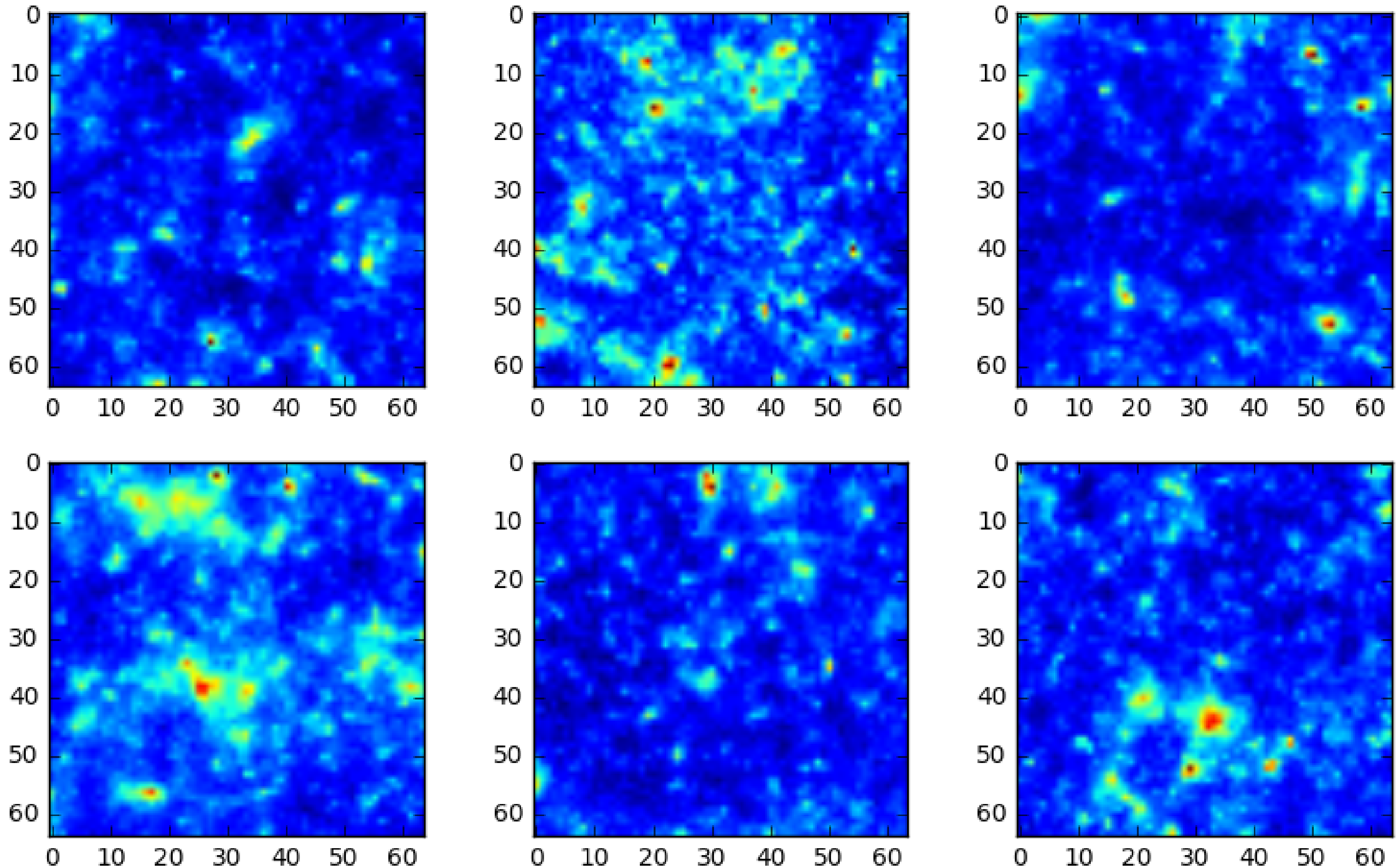


Epoch4

Cosmo DCGAN 64x64

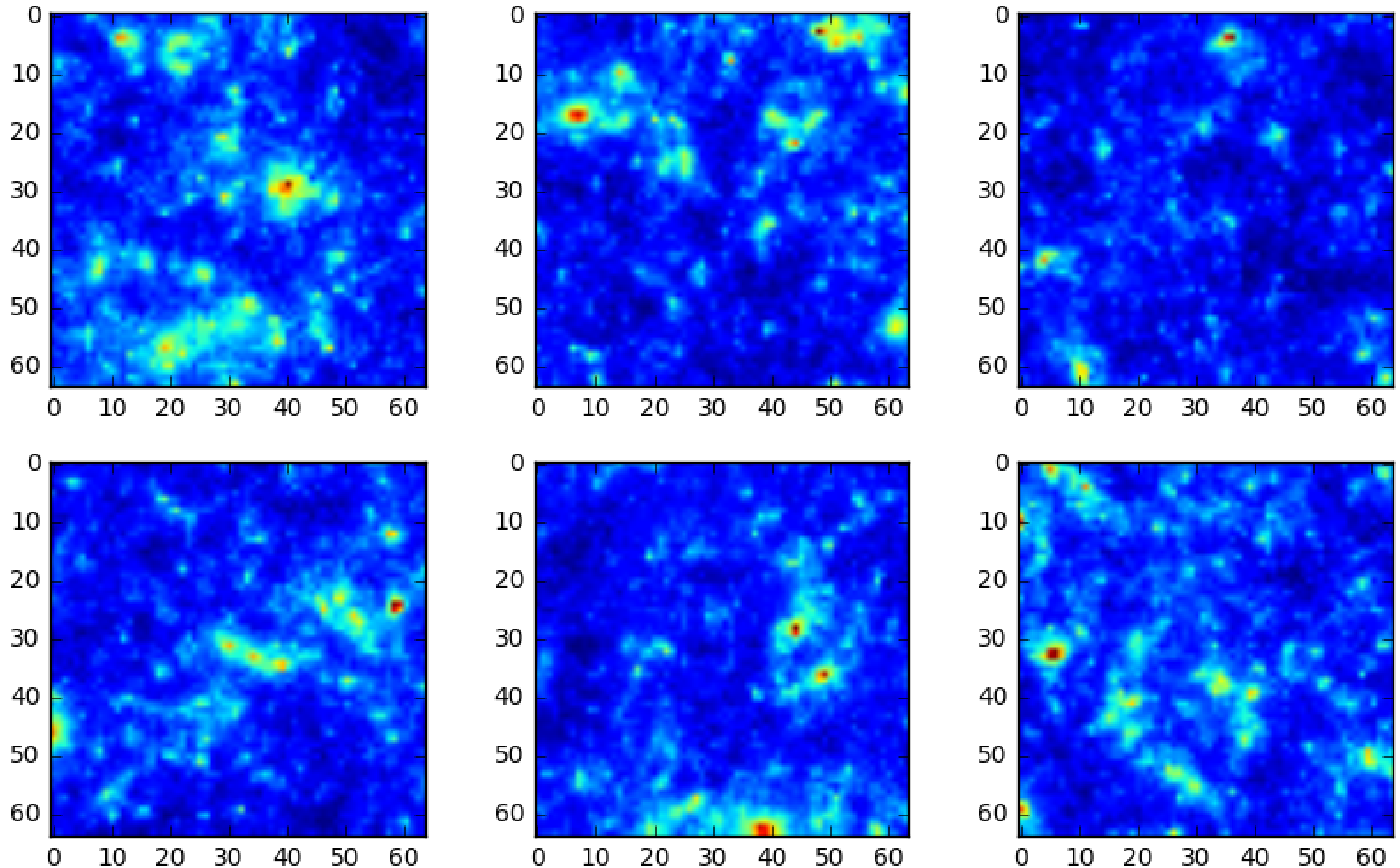
Generated

Real



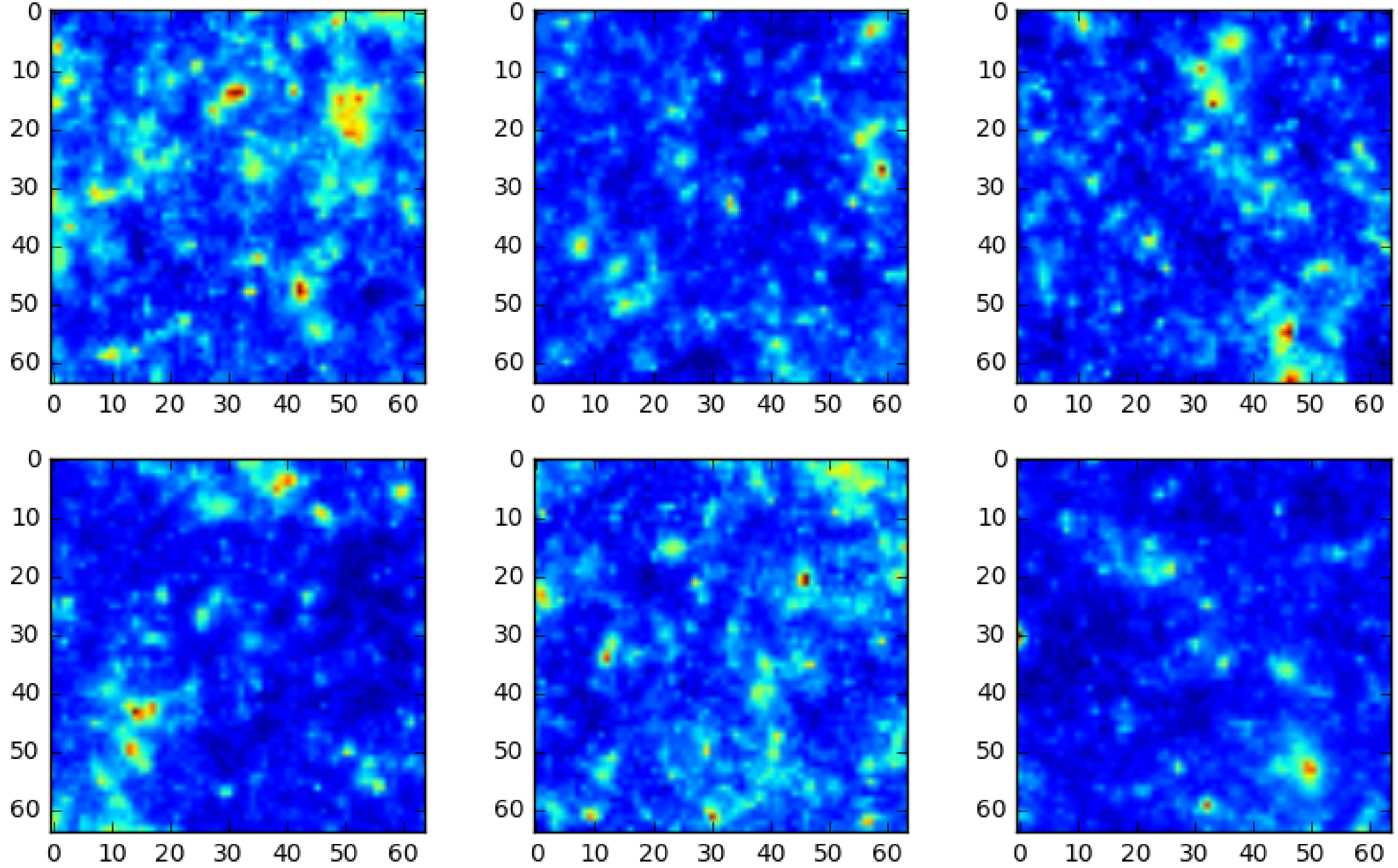
Epoch4

Cosmo DCGAN 64x64



Epoch8: Generated Images

Cosmo DCGAN 64x64

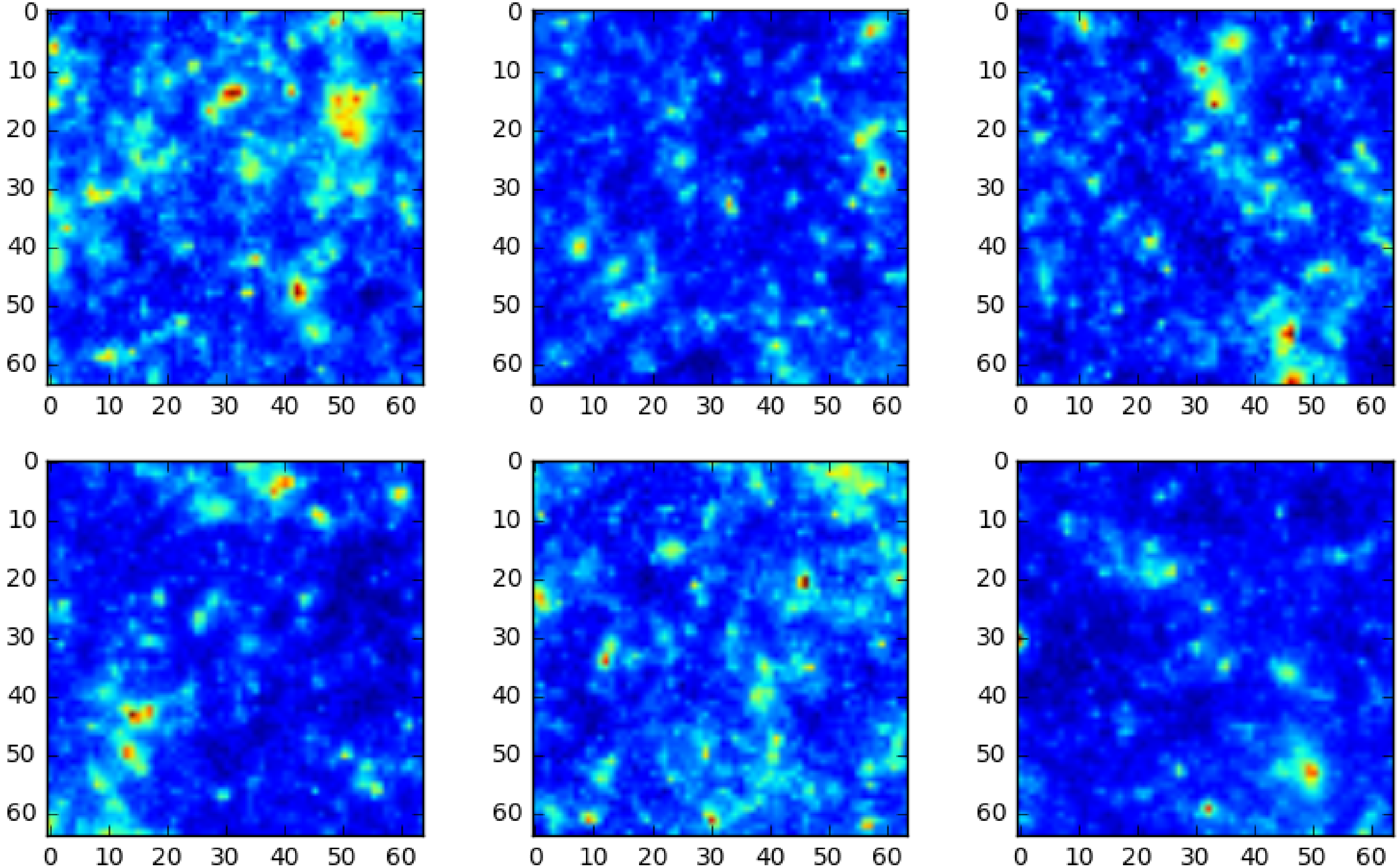


Epoch8

Cosmo DCGAN 64x64

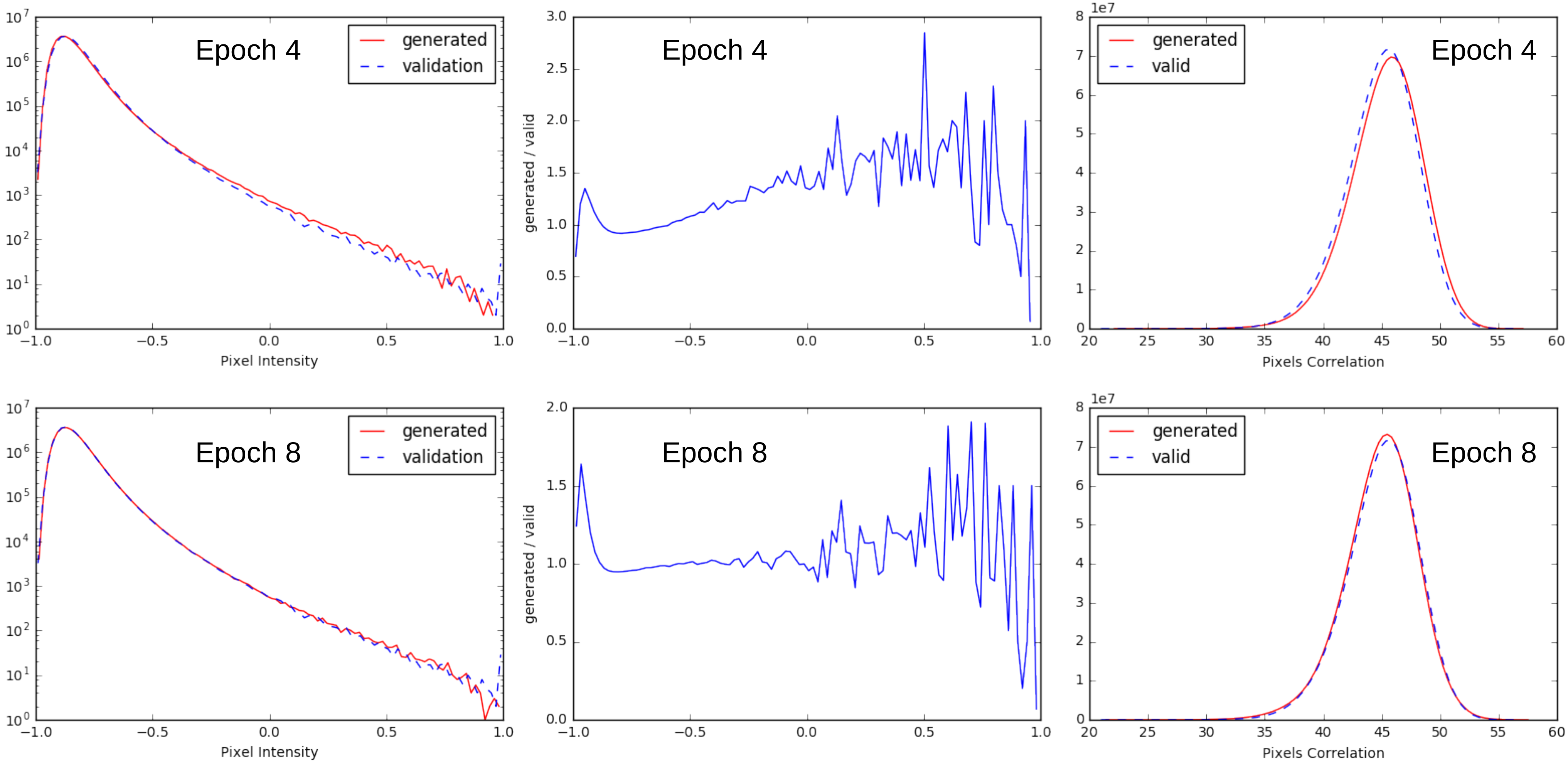
Generated

Real



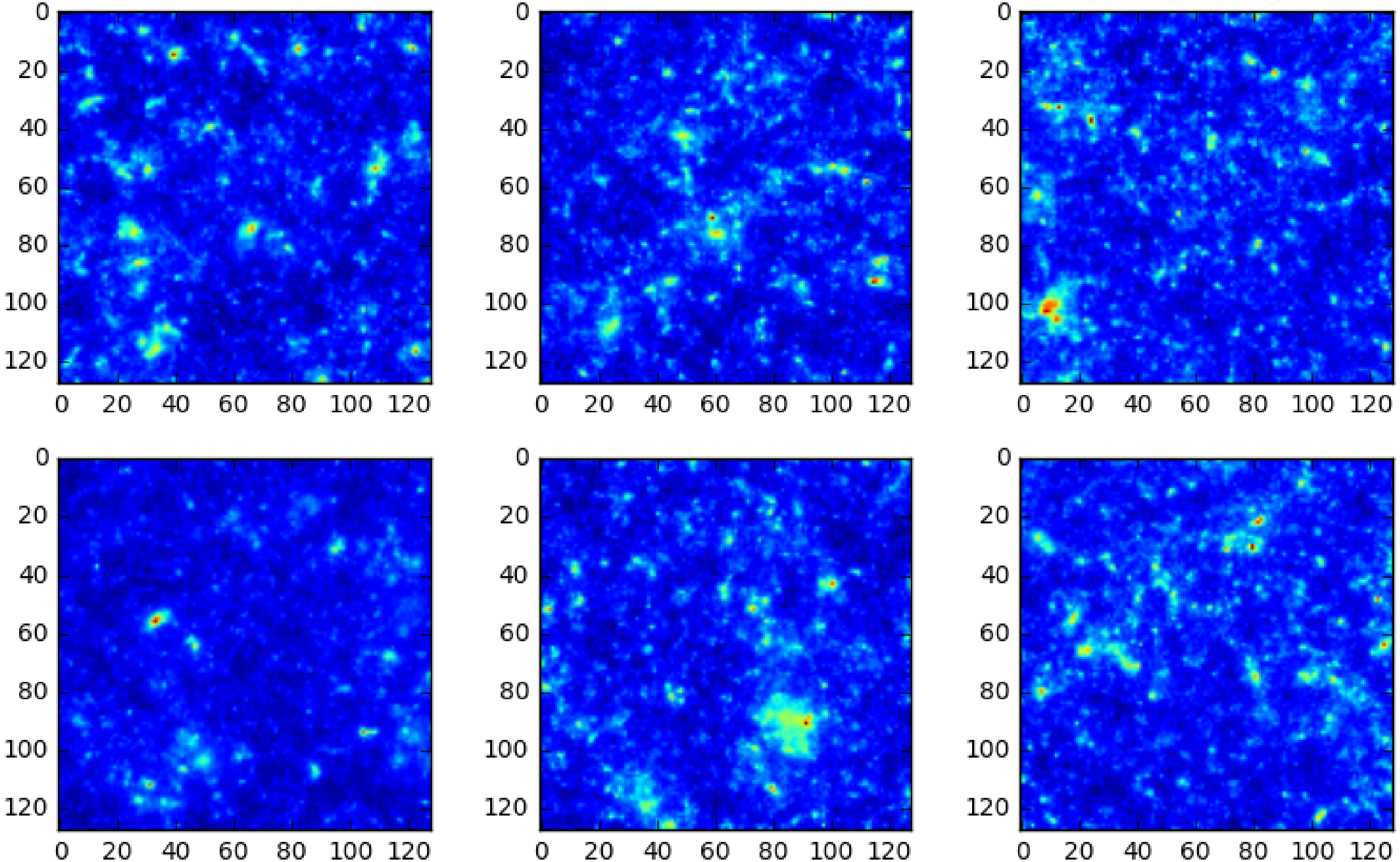
Epoch8

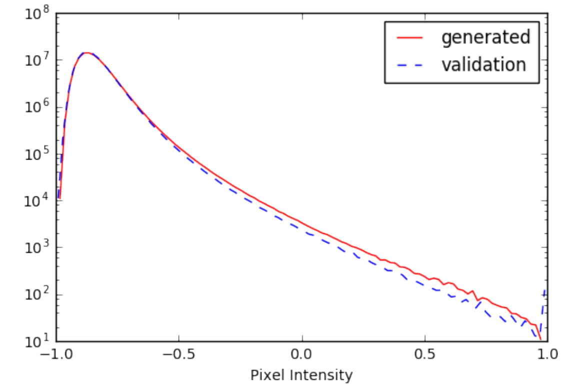
Cosmo DCGAN 64x64

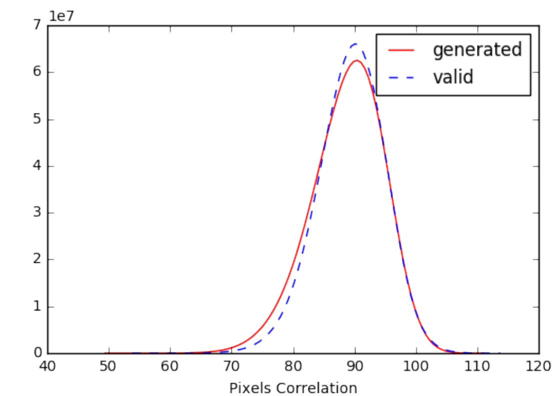


First-order (MickyMouse) tests for the reproducibility of correlations in real mass maps

Cosmo DCGAN 128x128

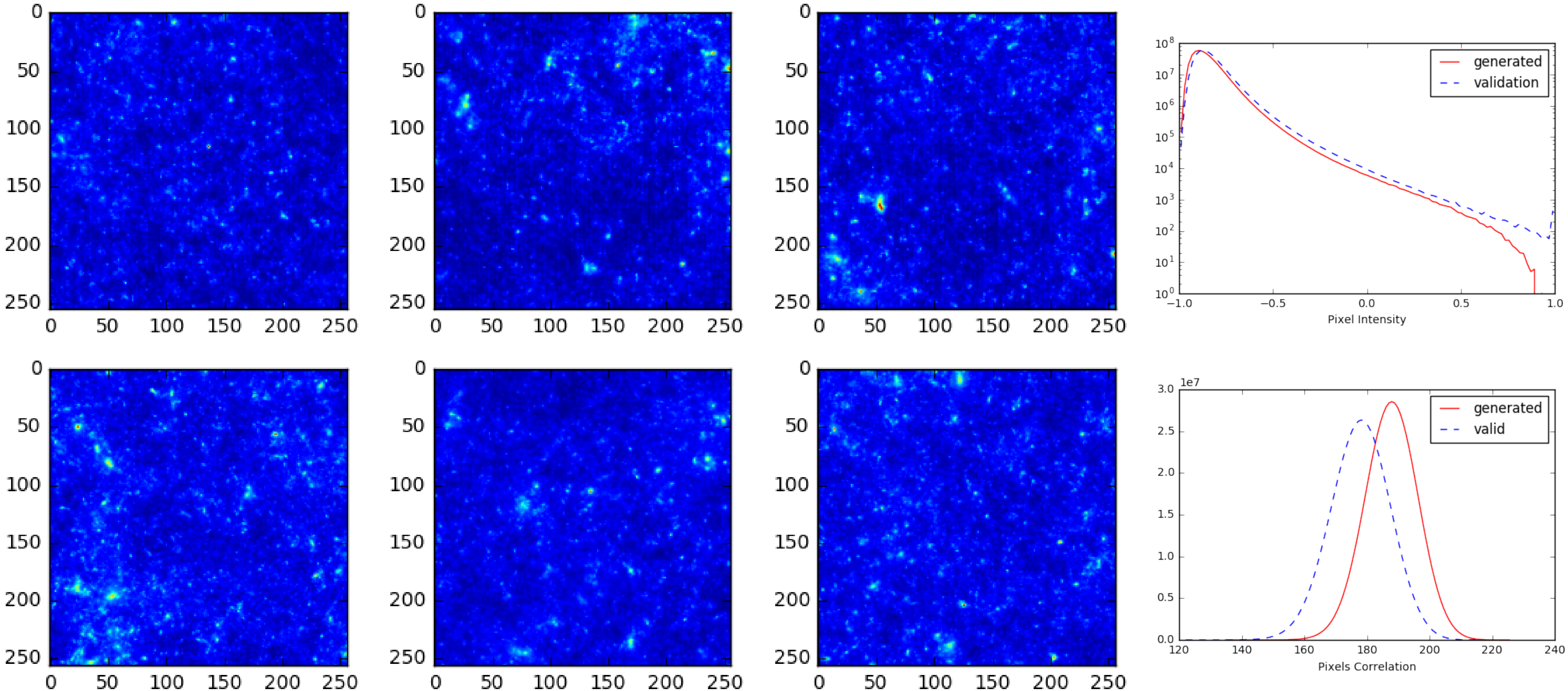






10 epochs on 200k images

Cosmo DCGAN 256x256



3 epochs on 200k images