“Creative” Neural Networks

The Hitchhiker’s Guide to ~~Deepfakes~~ Generative Modelling

Artem Maevskiy

Research Fellow – Institute for Functional Intelligent Materials @ National University of Singapore

MLHEP-2023

Erice, Apr 11 – 18, 2023

NUS National University of Singapore | Institute for Functional Intelligent Materials

INFN National Institute for Nuclear Physics

Practicum

EMFCSC — Ettore Majorana Foundation and Centre for Scientific Culture

Outline

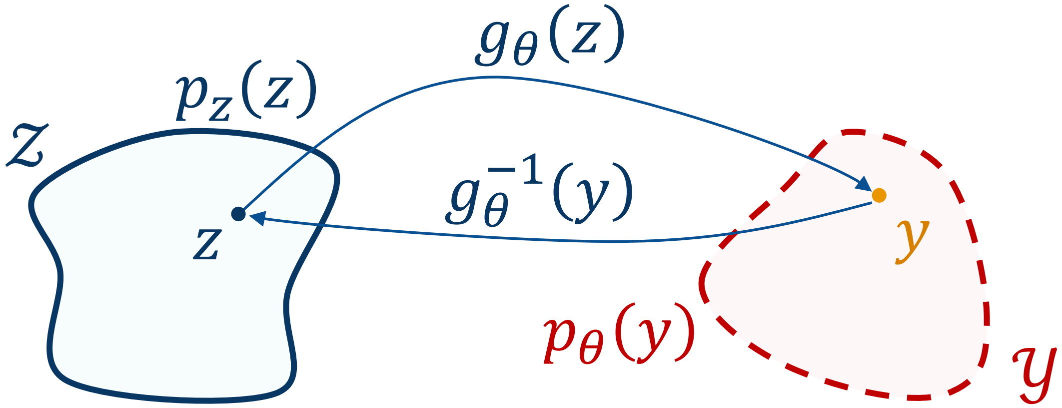
- Intro:
  - Generative modelling: what it is and how it’s useful
  - Generative modelling: how it’s done
- Generative Adversarial Networks (GAN)
- Variational Autoencoders (VAE)
- Normalizing Flows (NF)
- Evaluating Generative Models

Normalizing Flows

Basic idea

- Similarly to GAN and VAE, start with simple distribution for the latent codes $z \sim p_z$
- Similarly to GAN, map $z \to y$ deterministically
- Distinguishing feature: choose mapping $g_\theta(z)$ to be invertible

$$z = g_\theta^{-1}(y)$$



Simple max likelihood training

- Define $f_\theta(y) \equiv g_\theta^{-1}(y)$

$$\log p_\theta(y) = \log\left[\left|\det\frac{\partial f_\theta}{\partial y}\right| \cdot p_z\big(f_\theta(y)\big)\right]$$
$$= \log\left|\det\frac{\partial f_\theta}{\partial y}\right| + \log p_z\big(f_\theta(y)\big) \to \max_\theta$$

- For a sequence of transformations $f_\theta(\mathrm{y}) = f_\theta^n\left(\dots f_\theta^2\left(f_\theta^1(y)\right)\dots\right)$:

$$\log p_\theta(y) = \sum_i \log\left|\det\frac{\partial f_\theta^i}{\partial f_\theta^{i-1}}\right| + \log p_z\big(f_\theta(y)\big) \to \max_\theta$$

Practical considerations

- Our mappings should:
  - Be invertible
    - This means the dimensionalities of z and y are same
  - Be sufficiently expressive
  - Be computationally efficient (both functions and Jacobian determinant)

Simplest case

- Element-wise bijective function

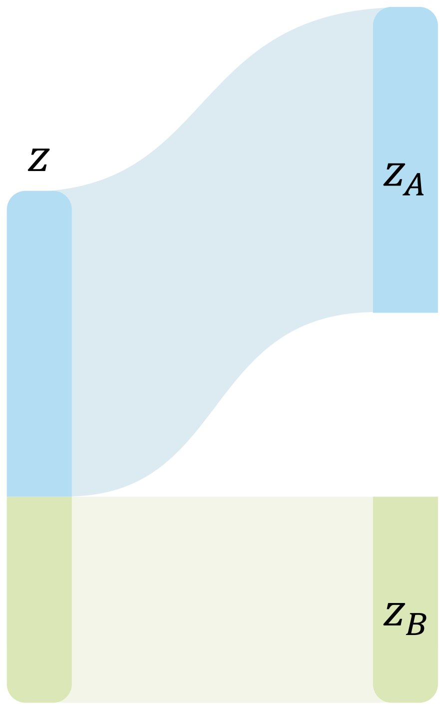
Linear flows

- For any invertible matrix $\mathbf{A}$, function $f(\mathbf{y}) = \mathbf{A}\mathbf{y} + \mathbf{b}$ is invertible too, e.g.:
  - Triangular
    - Possibly with permutations <= these can't be trained though
    - Inversion is $\sim \mathcal{O}(d^2)$
  - Orthogonal
    - Parameterized as product of reflections $\mathbf{H} = \mathbb{I} - \frac{2}{||\mathbf{v}||^2}\mathbf{v}\mathbf{v}^{\mathrm{T}}$

Coupling flows

$y = g(z), \quad g$:

$$z \to (z_A, z_B)$$

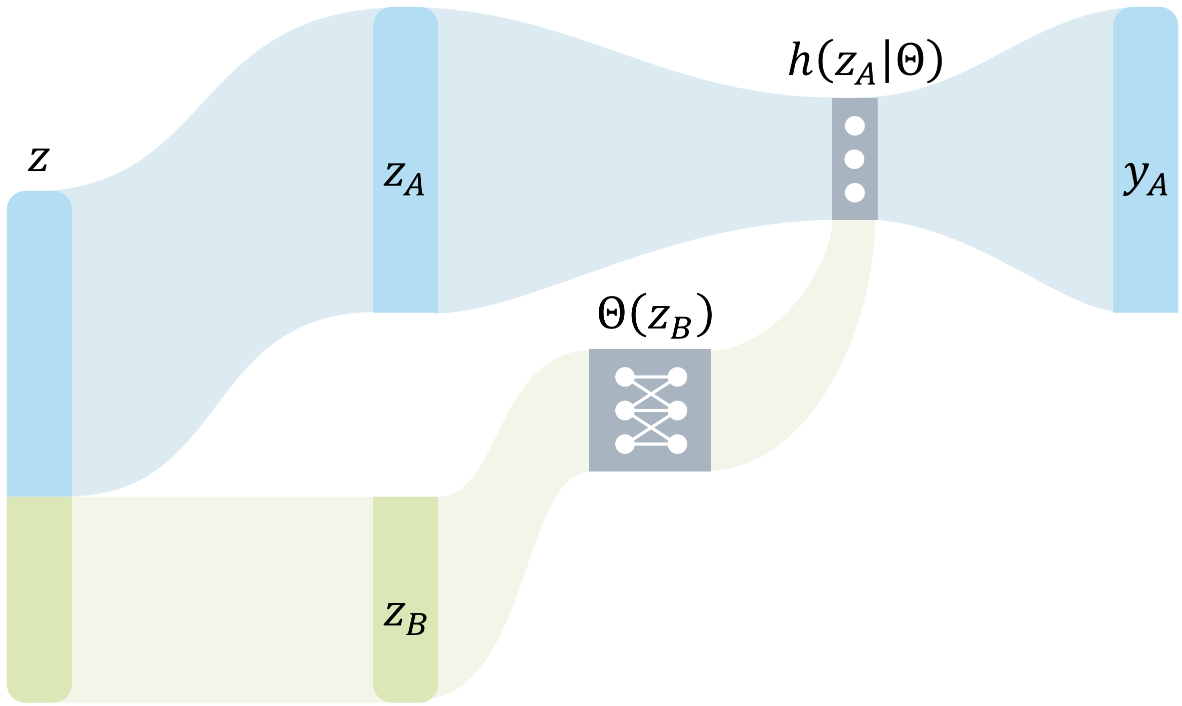


Coupling flows

$$y = g(z), \quad g\text{:}$$

$$z \rightarrow (z_A, z_B)$$

$$y_A = h(z_A | \Theta(z_B))$$

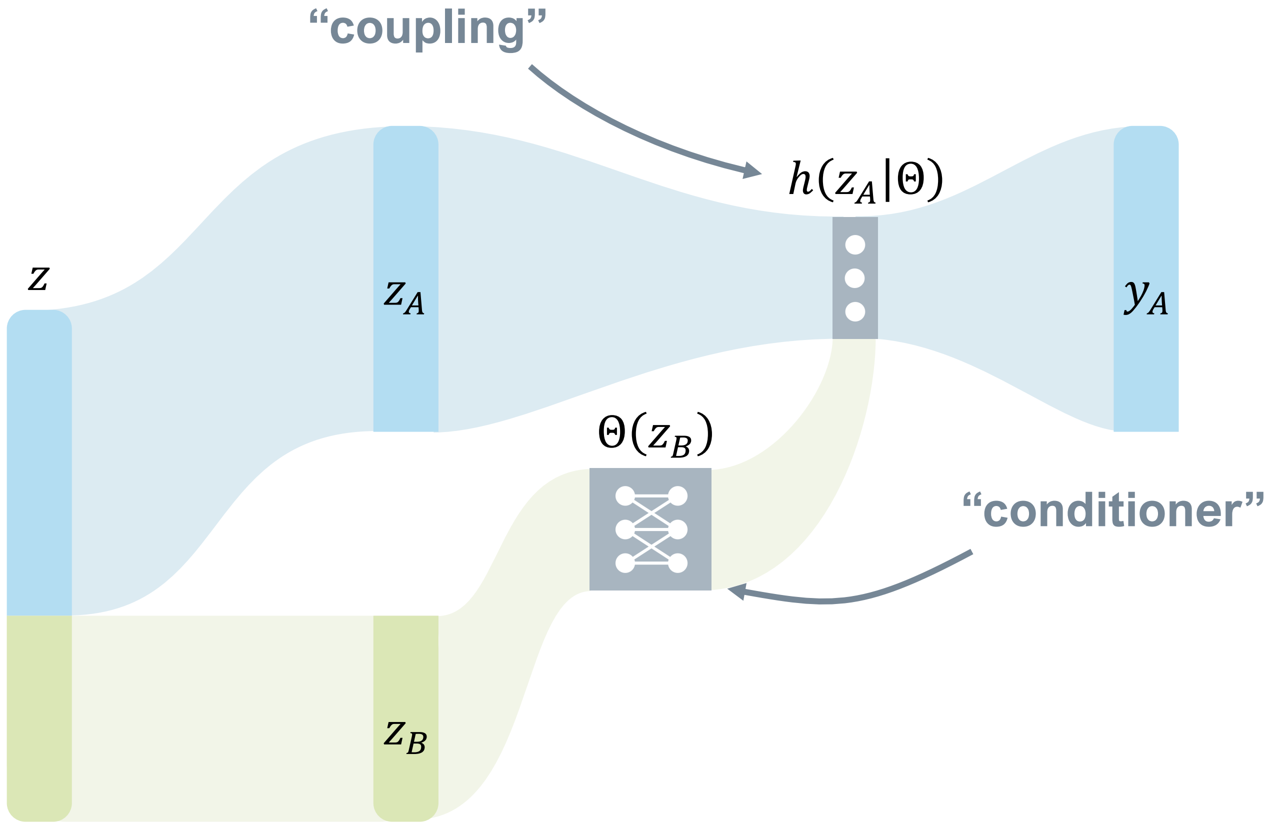


Coupling flows

$y = g(z), \quad g$:

$z \to (z_A, z_B)$

$y_A = h(z_A | \Theta(z_B))$



$h(z_A|\Theta)$ – some function, invertible wrt z_A

$\Theta(z_B)$ – any function (e.g., neural net)

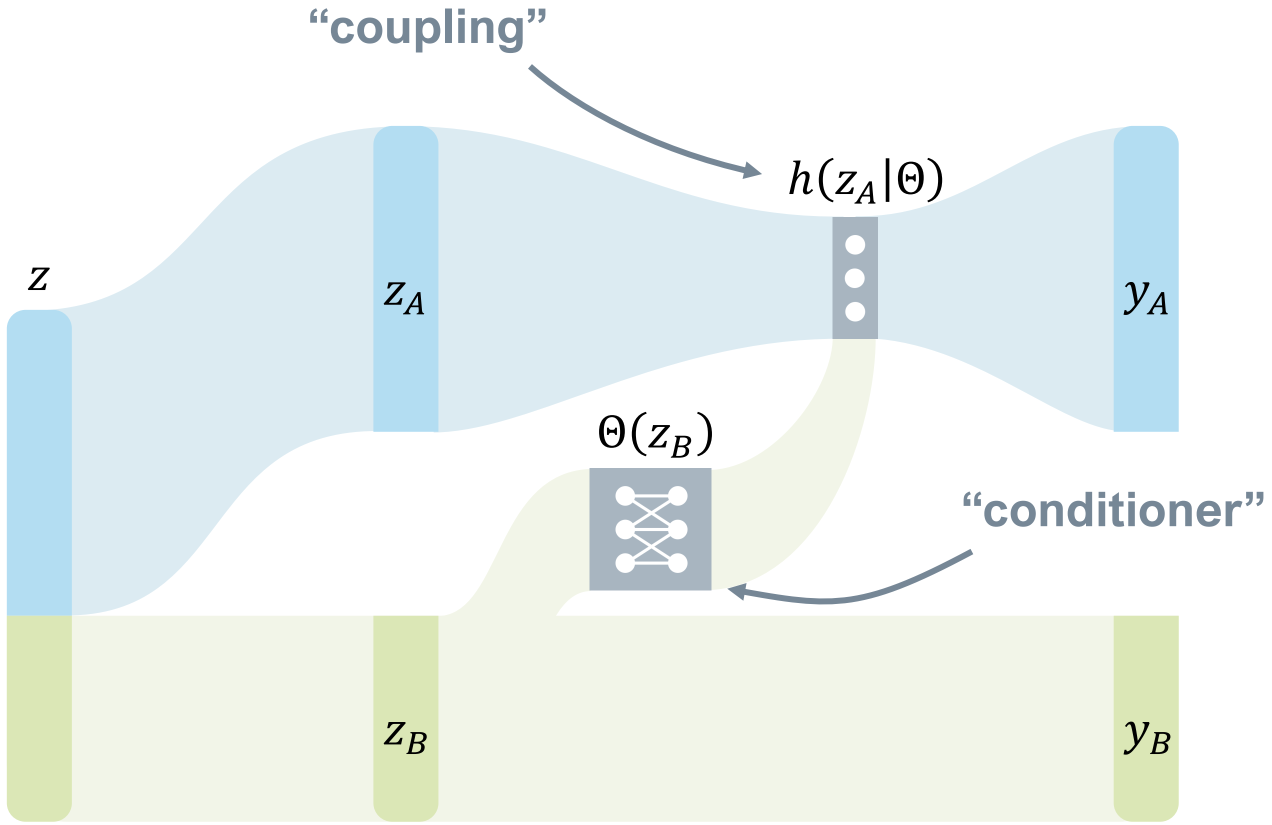
Coupling flows

$y = g(z), \quad g:$

$z \to (z_A, z_B)$

$y_A = h(z_A | \Theta(z_B))$

$y_B = z_B$



$h(z_A|\Theta)$ – some function, invertible wrt z_A

$\Theta(z_B)$ – any function (e.g., neural net)

Coupling flows

$y = g(z), \quad g$:

$z \to (z_A, z_B)$

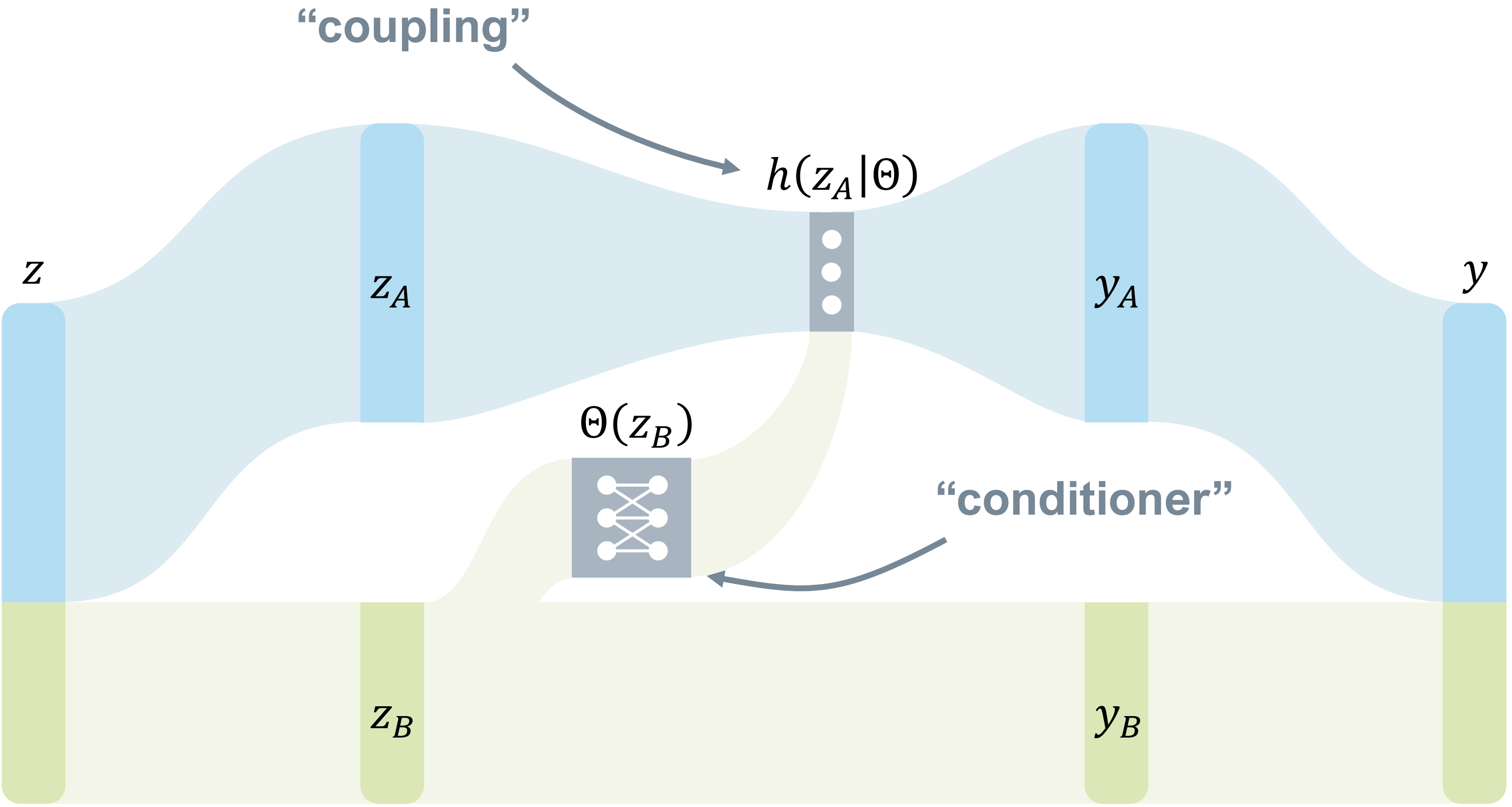
$y_A = h(z_A | \Theta(z_B))$

$y_B = z_B$

$(y_A, y_B) \to y$

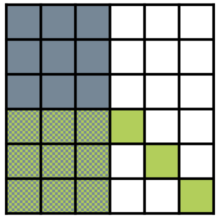
$h(z_A|\Theta)$ – some function, invertible wrt z_A

$\Theta(z_B)$ – any function (e.g., neural net)



Coupling flows

Jacobian structure:



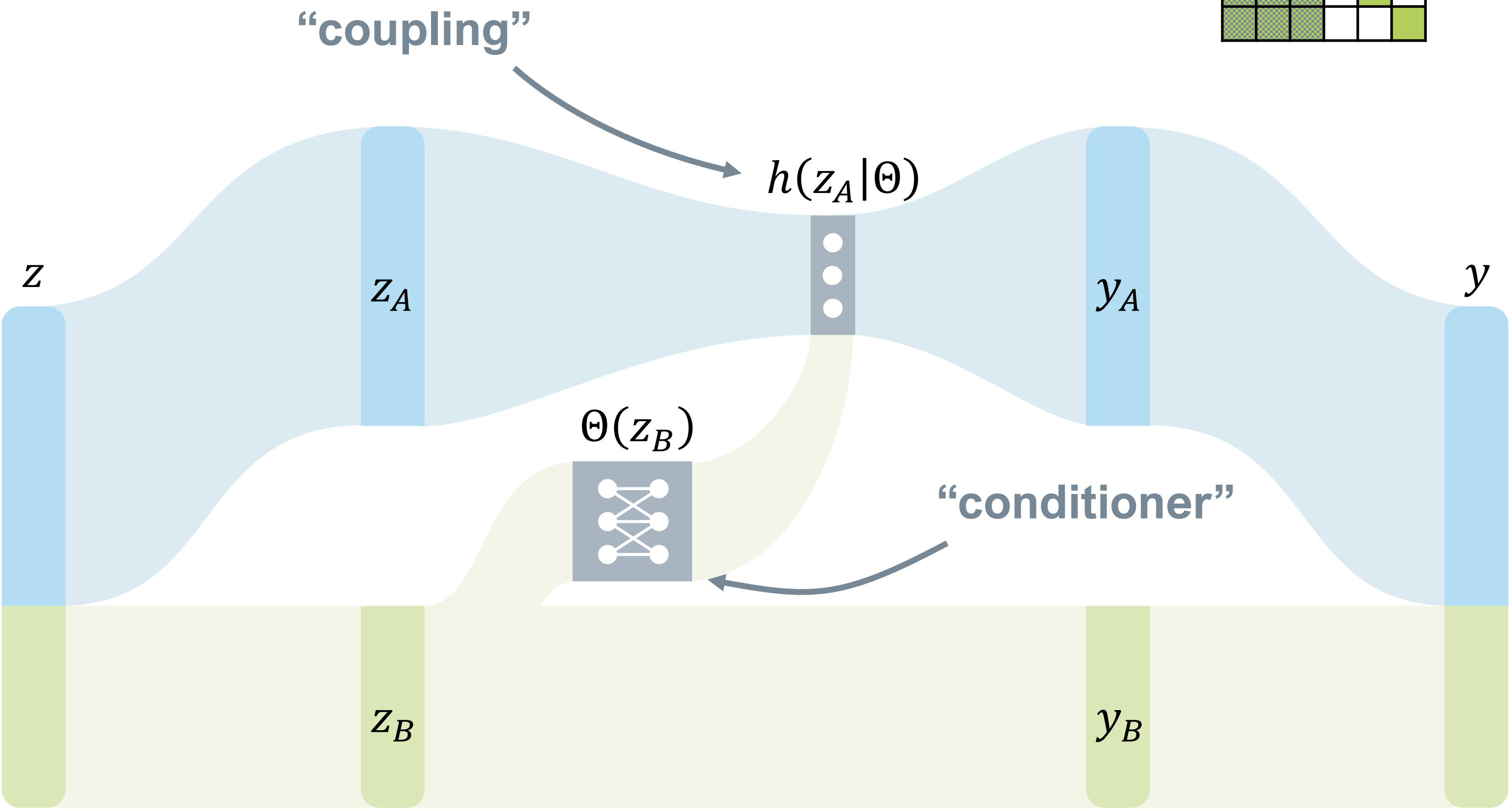
$y = g(z), \quad g$:

$$z \to (z_A, z_B)$$

$$y_A = h(z_A | \Theta(z_B))$$

$$y_B = z_B$$

$$(y_A, y_B) \to y$$



$h(z_A|\Theta)$ – some function, invertible wrt z_A

$\Theta(z_B)$ – any function (e.g., neural net)

Particular choices

- NICE (Non-linear Independent Components Estimation):

$$h(z_A|\Theta) = z_A + \Theta$$

- RealNVP (real-valued non-volume preserving transformations):

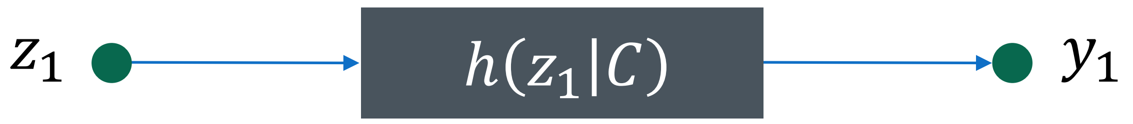
$$h(z_A|\Theta) = z_A \odot \exp \Theta_s + \Theta_{\mathrm{t}}$$

Autoregressive flows

- $z = (z_1, \dots, z_D)^T \to y = (y_1, \dots, y_D)^T$
- $h(\,\cdot\,|\theta)$: $\mathbb{R} \to \mathbb{R}$ – invertible function
- $y_i = h(z_i|\theta(z_{1:i-1}))$

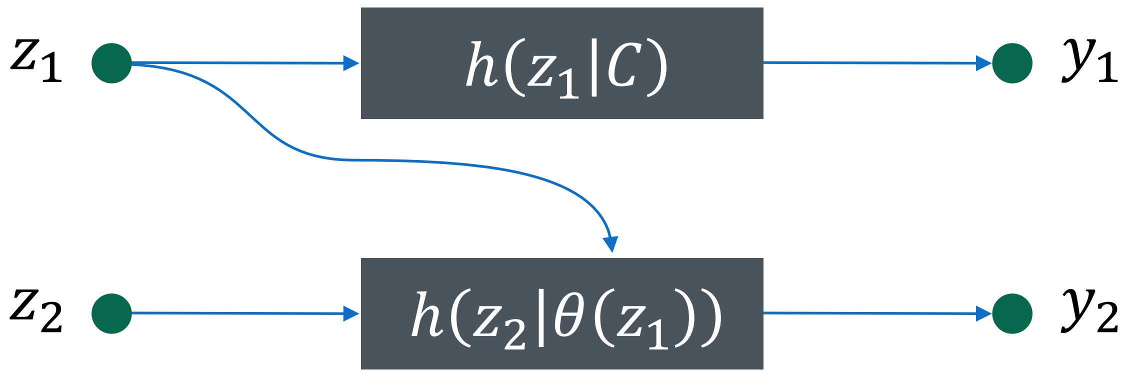
Autoregressive flows

- $z = (z_1, \dots, z_D)^T \to y = (y_1, \dots, y_D)^T$
- $h(\cdot|\theta)$: $\mathbb{R} \to \mathbb{R}$ – invertible function
- $y_i = h(z_i|\theta(z_{1:i-1}))$



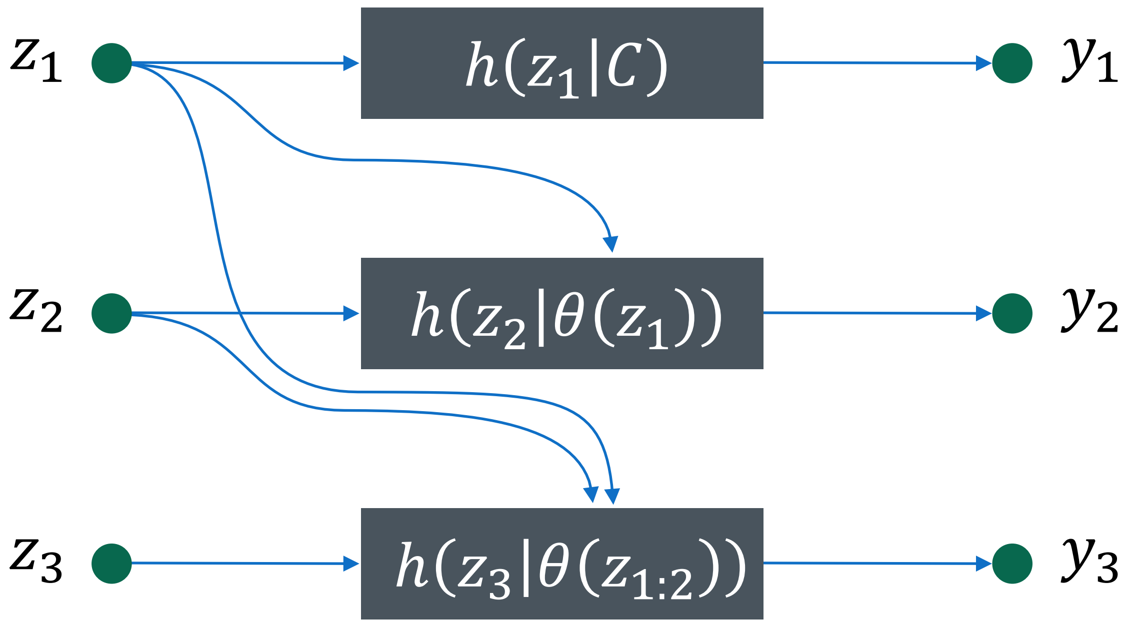
Autoregressive flows

- $z = (z_1, \dots, z_D)^T \rightarrow y = (y_1, \dots, y_D)^T$
- $h(\cdot\,|\theta)\colon \mathbb{R} \rightarrow \mathbb{R}$ – invertible function
- $y_i = h(z_i|\theta(z_{1:i-1}))$



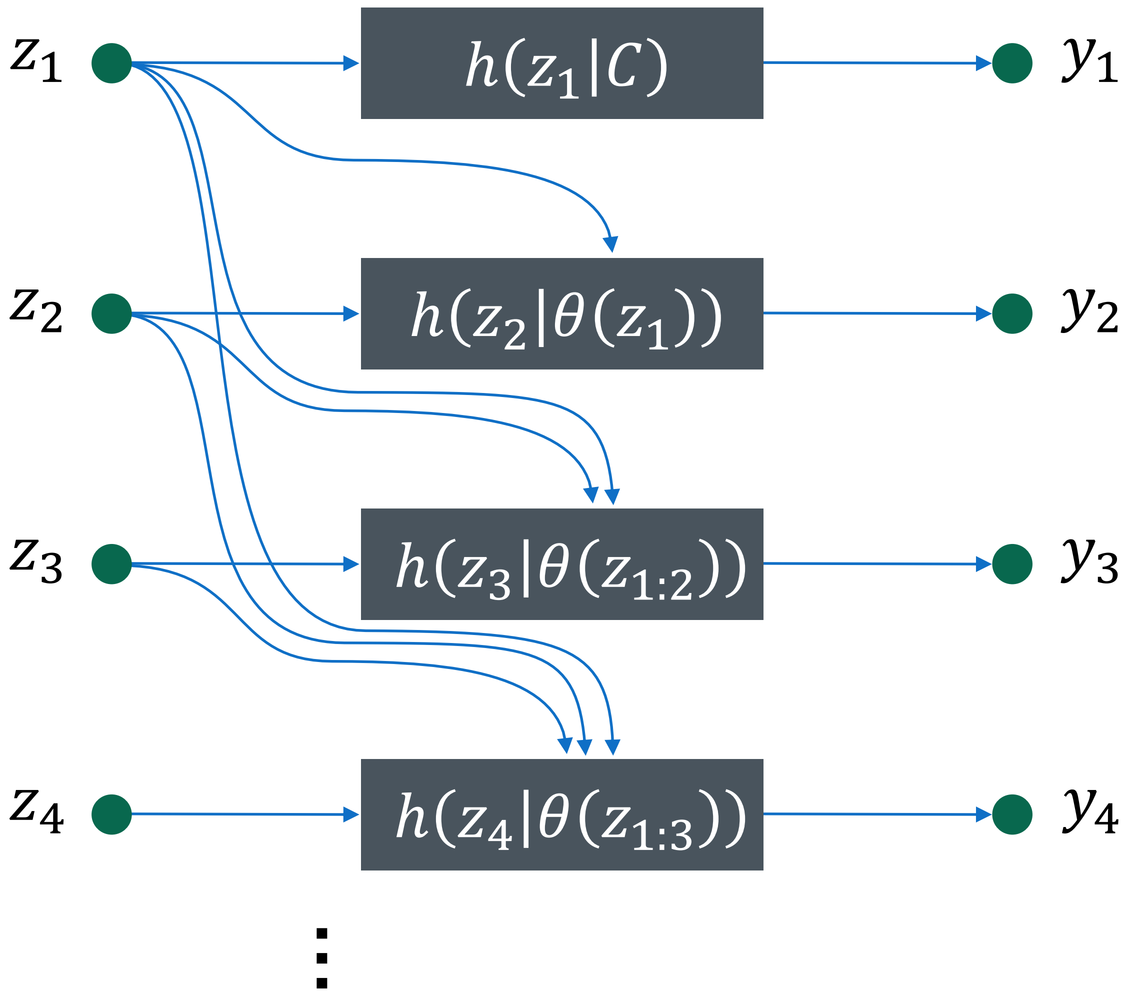
Autoregressive flows

- $z = (z_1, \dots, z_D)^T \to y = (y_1, \dots, y_D)^T$
- $h(\cdot|\theta)$: $\mathbb{R} \to \mathbb{R}$ – invertible function
- $y_i = h(z_i|\theta(z_{1:i-1}))$



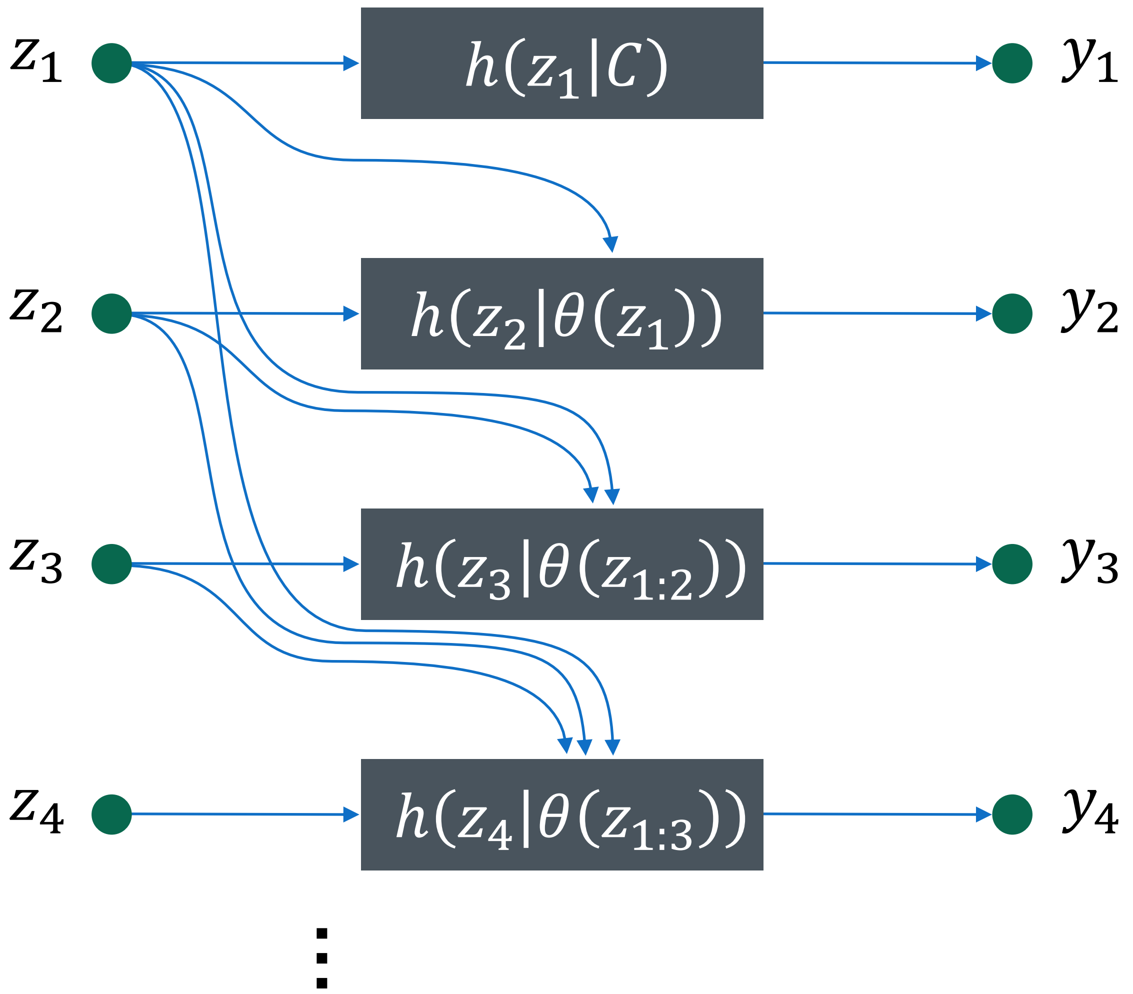
Autoregressive flows

- $z = (z_1, \dots, z_D)^T \to y = (y_1, \dots, y_D)^T$
- $h(\cdot|\theta)$: $\mathbb{R} \to \mathbb{R}$ – invertible function
- $y_i = h(z_i|\theta(z_{1:i-1}))$



Autoregressive flows

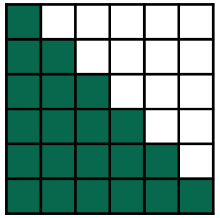
- $z = (z_1, \dots, z_D)^T \to y = (y_1, \dots, y_D)^T$
- $h(\,\cdot\,|\theta)\colon\ \mathbb{R} \to \mathbb{R}$ – invertible function
- $y_i = h(z_i|\theta(z_{1:i-1}))$
- Inverse involves recursion

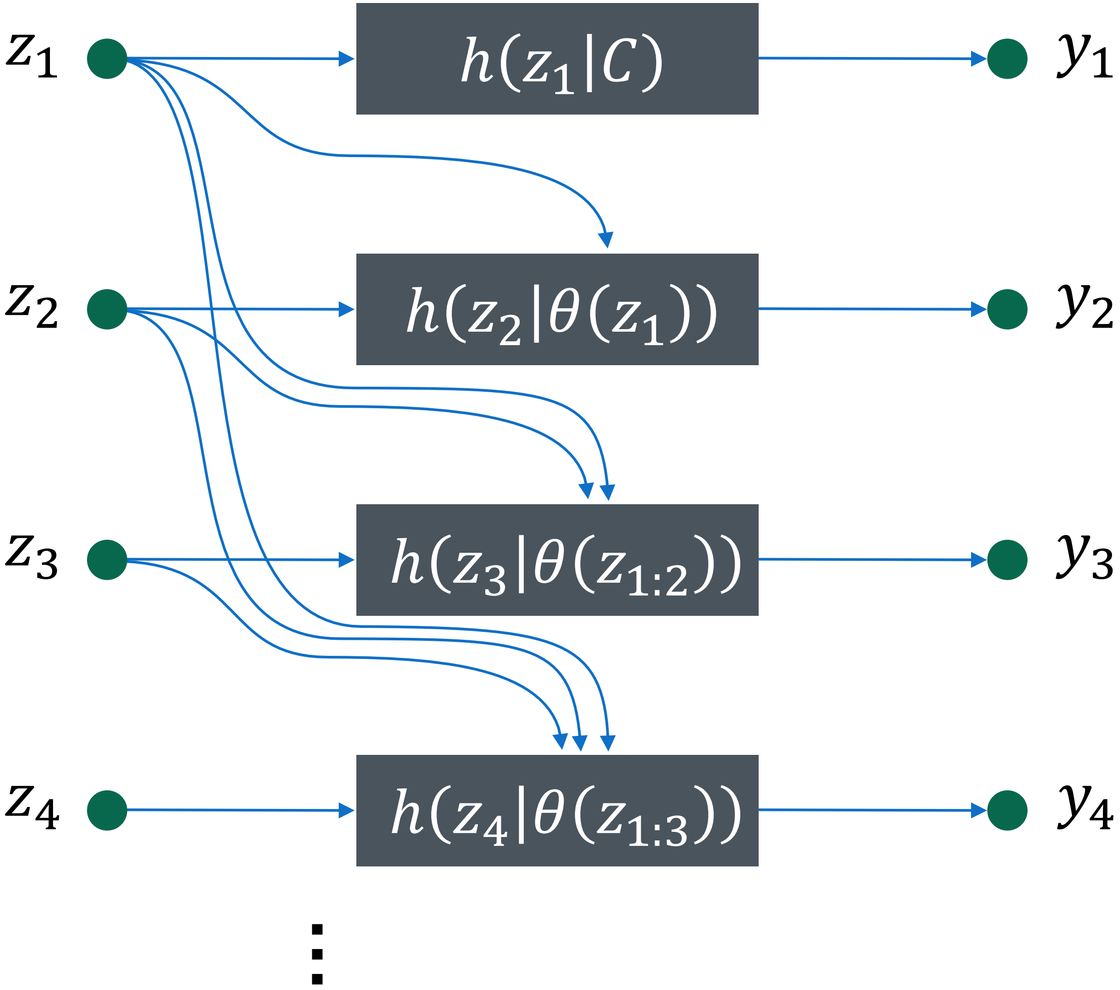


Autoregressive flows

- $z = (z_1, \ldots, z_D)^T \to y = (y_1, \ldots, y_D)^T$
- $h(\cdot\,|\theta)\colon\ \mathbb{R} \to \mathbb{R}$ – invertible function
- $y_i = h(z_i|\theta(z_{1:i-1}))$
- Inverse involves recursion

Jacobian structure:





Evaluating Generative Models

Evaluating generative models

- No single guide to follow
- Approaches are very problem-specific
  - E.g. perceived visual quality of generated images vs. quality of a generated dental crown model (https://arxiv.org/abs/1804.00064)
  - Most solutions are adapted to or invented for a given particular task
- We'll mention some approaches

Evaluating generative models

(most obvious thing to do)

- By-eye comparison (if your data allows that)
  - Compare individual objects or whole distributions (e.g. in projections)
  - There might be no need to do any complicated evaluation if the model results simply look bad

Evaluating generative models

(simple things to do)

- Compare meaningful physical characteristics (if applicable)
  - Means, medians, standard deviations, etc.
  - Correlations
- Statistical tests (χ^2, Kolmogorov-Smirnov, etc.)
  - between individual dimensions or projections

Additional classifier

- Train an independent model (e.g. xgboost) to distringuish real and fake samples
- Evaluate your GAN by checking the classifier's score (e.g. ROC AUC)
- Pros:
    - An objective quality measure
- Cons:
    - Resource consuming
    - Requires hyper-parameter tuning
    - May get picky to things that are not important

Inception score

- Introduced in https://arxiv.org/abs/1606.03498
- Apply the Inception model (pre-trained image classifier) to obtain the conditional label distribution $p(y|x)$ for each image x
  - this should be low-entropy (the classifier should be certain)
- Calculate marginal $p(y) = \int p(y|x = G(z))p(z)dz$
  - this should be high-entropy (diversity of samples)
- Combining these two requirements:

$$\mathrm{IS} = \exp\left[\mathbb{E}_x\left[\mathrm{KL}\left(p(y|x) \mid\mid p(y)\right)\right]\right]$$

Fréchet inception distance (FID score)

- Introduced in https://arxiv.org/abs/1706.08500
- One of the drawbacks of IS is that it doesn't care about the true distribution
- Instead one can compare distributions of activations at some Inception layer (originally – last pooling layer)
- The authors proposed calculating the Fréchet (aka Wasserstein-2) distance
- Distance between multivariate Gaussian approximations:

$$\mathrm{FID} = \left\|\mu_r - \mu_g\right\|^2 + \mathrm{Tr}\left[\Sigma_r + \Sigma_g - 2\left(\Sigma_r\Sigma_g\right)^{1/2}\right]$$

Precision and Recal distance (PRD)

Definition 1. *For $\alpha, \beta \in (0, 1]$, the probability distribution Q has precision α at recall β w.r.t. P if there exist distributions μ, ν_P and ν_Q such that*

Decomposition with a common part

$$P = \beta\mu + (1-\beta)\nu_P \quad \textit{and} \quad Q = \alpha\mu + (1-\alpha)\nu_Q. \tag{3}$$

Definition 2. *The set of attainable pairs of precision and recall of a distribution Q w.r.t. a distribution P is denoted by* $\mathrm{PRD}(Q, P)$ *and it consists of all* (α, β) *satisfying Definition 1 and the pair* $(0, 0)$.

- The authors provide an algorithm to calculate it for discrete distributions
- They convert Inception activations to discrete distribution using *k*-means clustering

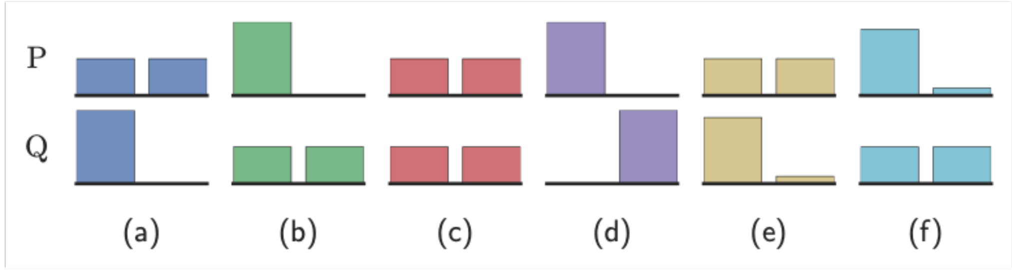


Figure 2: Intuitive examples of P and Q.

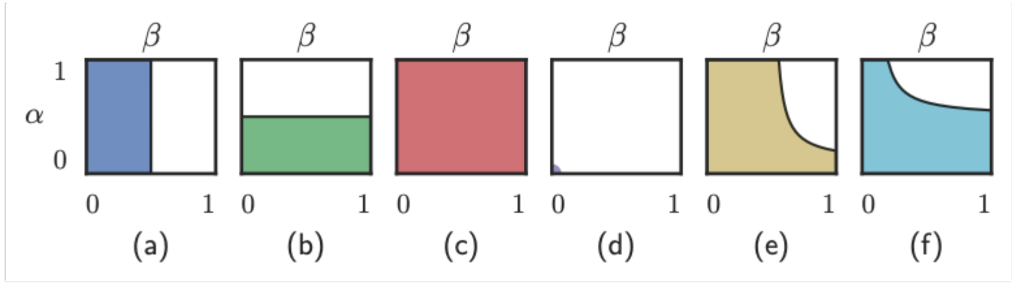


Figure 3: $\mathrm{PRD}(Q, P)$ for the examples above.

https://arxiv.org/abs/1806.00035

More metrics…

- An extensive comparison of a large variety of measures:
  - https://arxiv.org/abs/1802.03446

	Measure	Description
Quantitative	1. Average Log-likelihood [18, 22]	• Log likelihood of explaining realworld held out/test data using a density estimated from the generated data (*e.g.* using KDE or Parzen window estimation). $L = \frac{1}{N}\sum_i \log P_{model}(\mathbf{x}_i)$
	2. Coverage Metric [33]	• The probability mass of the true data "covered" by the model distribution $C := P_{data}(dP_{model} > t)$ with t such that $P_{model}(dP_{model} > t) = 0.95$
	3. Inception Score (IS) [3]	• KLD between conditional and marginal label distributions over generated data. $\exp\left(\mathbb{E}_{\mathbf{x}}\left[\mathbb{KL}\left(p\left(\mathrm{y} \mid \mathbf{x}\right) \Vert p\left(\mathrm{y}\right)\right)\right]\right)$
	4. Modified Inception Score (m-IS) [34]	• Encourages diversity within images sampled from a particular category. $\exp(\mathbb{E}_{\mathbf{x}_i}[\mathbb{E}_{\mathbf{x}_j}[(\mathbb{KL}(P(y\mid\mathbf{x}_i)\Vert P(y\mid\mathbf{x}_j))]])$
	5. Mode Score (MS) [35]	• Similar to IS but also takes into account the prior distribution of the labels over real data. $\exp\left(\mathbb{E}_{\mathbf{x}}\left[\mathbb{KL}\left(p\left(y \mid \mathbf{x}\right) \Vert p\left(y^{train}\right)\right)\right] - \mathbb{KL}\left(p\left(y\right) \Vert p\left(y^{train}\right)\right)\right)$
	6. AM Score [36]	• Takes into account the KLD between distributions of training labels vs. predicted labels, as well as the entropy of predictions. $\mathbb{KL}(p(y^{\mathrm{train}}) \Vert p(y)) + \mathbb{E}_{\mathbf{x}}\left[H(y\mid\mathbf{x})\right]$
	7. Fréchet Inception Distance (FID) [37]	• Wasserstein-2 distance between multi-variate Gaussians fitted to data embedded into a feature space $FID(r,g) = \Vert\mu_r - \mu_g\Vert_2^2 + Tr(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{\frac{1}{2}})$
	8. Maximum Mean Discrepancy (MMD) [38]	• Measures the dissimilarity between two probability distributions P_r and P_g using samples drawn independently from each distribution. $M_k(P_r, P_g) = \mathbb{E}_{\mathbf{x},\mathbf{x}' \sim P_r}[k(\mathbf{x},\mathbf{x}')] - 2\mathbb{E}_{\mathbf{x}\sim P_r, \boldsymbol{y}\sim P_g}[k(\mathbf{x},\boldsymbol{y})] + \mathbb{E}_{\boldsymbol{y},\boldsymbol{y}'\sim P_g}[k(\boldsymbol{y},\boldsymbol{y}')]$
	9. The Wasserstein Critic [39]	• The critic (*e.g.* an NN) is trained to produce high values at real samples and low values at generated samples $\hat{W}(\mathbf{x}_{test}, \mathbf{x}_g) = \frac{1}{N}\sum_{i=1}^{N}\hat{f}(\mathbf{x}_{test}[i]) - \frac{1}{N}\sum_{i=1}^{N}\hat{f}(\mathbf{x}_g[i])$
	10. Birthday Paradox Test [27]	• Measures the support size of a discrete (continuous) distribution by counting the duplicates (near duplicates)
	11. Classifier Two Sample Test (C2ST) [40]	• Answers whether two samples are drawn from the same distribution (*e.g.* by training a binary classifier)
	12. Classification Performance [1, 15]	• An indirect technique for evaluating the quality of unsupervised representations (*e.g.* feature extraction; FCN score). See also the GAN Quality Index (GQI) [41].
	13. Boundary Distortion [42]	• Measures diversity of generated samples and covariate shift using classification methods.
	14. Number of Statistically-Different Bins (NDB) [43]	• Given two sets of samples from the same distribution, the number of samples that fall into a given bin should be the same up to sampling noise
	15. Image Retrieval Performance [44]	• Measures the distributions of distances to the nearest neighbors of some query images (*i.e.* diversity)
	16. Generative Adversarial Metric (GAM) [31]	• Compares two GANs by having them engaged in a battle against each other by swapping discriminators or generators. $p(\mathbf{x}\mid y=1; M_1^{'})/p(\mathbf{x}\mid y=1; M_2^{'}) = \left(p(y=1\mid\mathbf{x}; D_1)p(\mathbf{x}; G_2)\right)/\left(p(y=1\mid\mathbf{x}; D_2)p(\mathbf{x}; G_1)\right)$
	17. Tournament Win Rate and Skill Rating [45]	• Implements a tournament in which a player is either a discriminator that attempts to distinguish between real and fake data or a generator that attempts to fool the discriminators into accepting fake data as real.
	18. Normalized Relative Discriminative Score (NRDS) [32]	• Compares n GANs based on the idea that if the generated samples are closer to real ones, more epochs would be needed to distinguish them from real samples.
	19. Adversarial Accuracy and Divergence [46]	• Adversarial Accuracy. Computes the classification accuracies achieved by the two classifiers, one trained on real data and another on generated data, on a labeled validation set to approximate $P_g(y\mid\mathbf{x})$ and $P_r(y\mid\mathbf{x})$. Adversarial Divergence: Computes $\mathbb{KL}(P_g(y\mid\mathbf{x}), P_r(y\mid\mathbf{x}))$
	20. Geometry Score [47]	• Compares geometrical properties of the underlying data manifold between real and generated data.
	21. Reconstruction Error [48]	• Measures the reconstruction error (*e.g.* L_2 norm) between a test image and its closest generated image by optimizing for z (*i.e.* $min_{\mathbf{z}}\Vert G(\mathbf{z}) - \mathbf{x}^{(test)}\Vert^2$)
	22. Image Quality Measures [49, 50, 51]	• Evaluates the quality of generated images using measures such as SSIM, PSNR, and sharpness difference
	23. Low-level Image Statistics [52, 53]	• Evaluates how similar low-level statistics of generated images are to those of natural scenes in terms of mean power spectrum, distribution of random filter responses, contrast distribution, etc.
	24. Precision, Recall and F_1 score [23]	• These measures are used to quantify the degree of overfitting in GANs, often over toy datasets.
Qualitative	1. Nearest Neighbors	• To detect overfitting, generated samples are shown next to their nearest neighbors in the training set
	2. Rapid Scene Categorization [18]	• In these experiments, participants are asked to distinguish generated samples from real images in a short presentation time (*e.g.* 100 ms); *i.e.* real v.s fake
	3. Preference Judgment [54, 55, 56, 57]	• Participants are asked to rank models in terms of the fidelity of their generated images (*e.g.* pairs, triples)
	4. Mode Drop and Collapse [58, 59]	• Over datasets with known modes (*e.g.* a GMM or a labeled dataset), modes are computed as by measuring the distances of generated data to mode centers
	5. Network Internals [1, 60, 61, 62, 63, 64]	• Regards exploring and illustrating the internal representation and dynamics of models (*e.g.* space continuity) as well as visualizing learned features