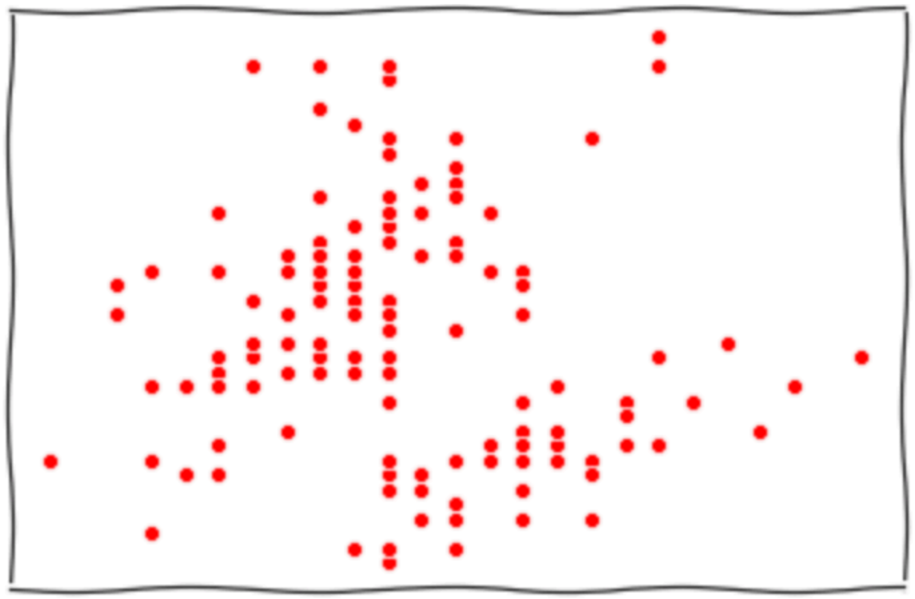
Lec 20: Triangular Flows

Yaoliang Yu

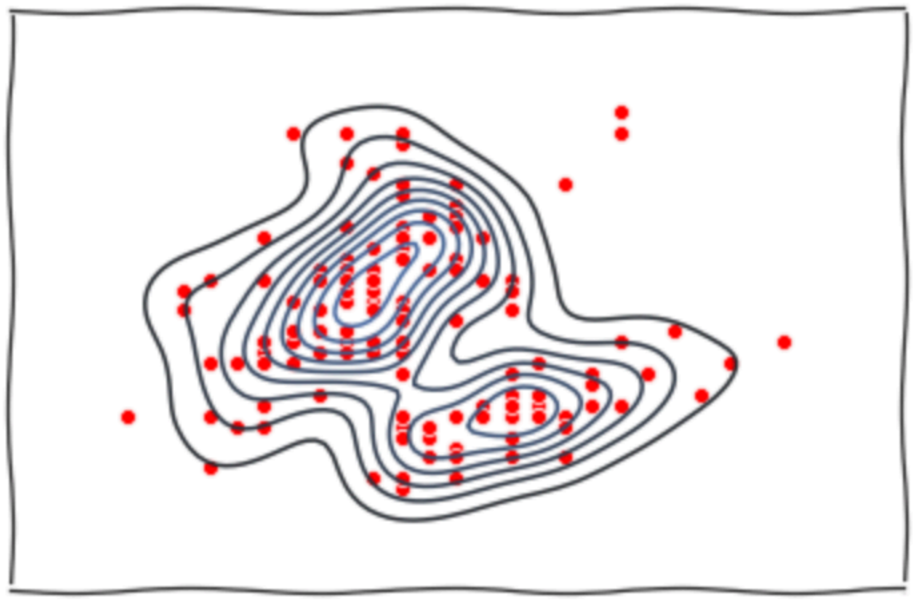
July 16, 2020

UNIVERSITY OF WATERLOO

density estimation



data $= \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$



estimate $p(\mathbf{x})$

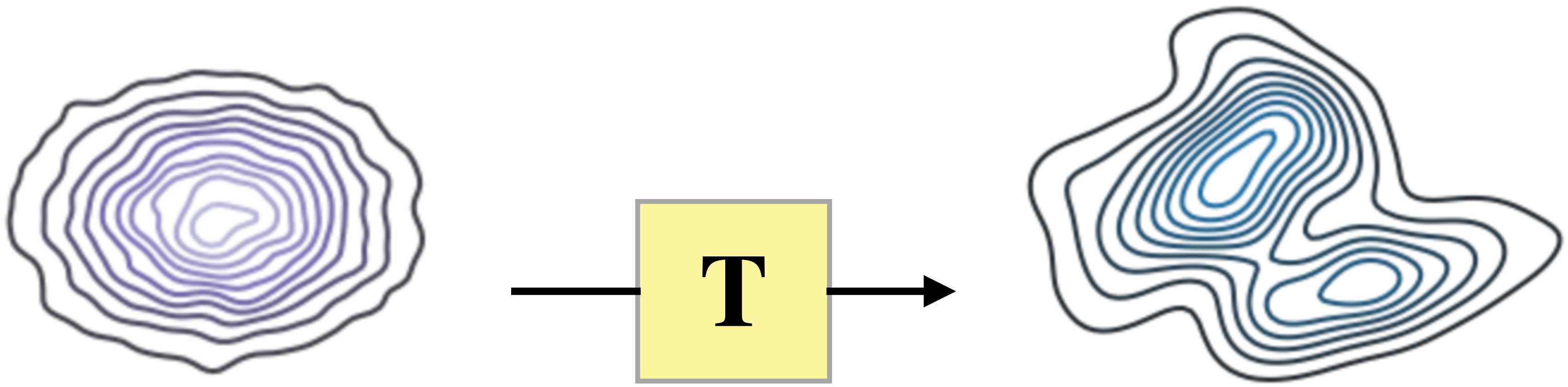
If we can generate, then we can classify

Realistic

Kingma et. al. Glow:Generative Flow with Invertible 1x1 Convolutions, NeurIPS 2018

Overview

find *deterministic* maps from source to target density



source $q(z)$

target $p(x)$

$$x_i = \mathbf{T}(z_i)$$

learn *bijective* & *differentiable* transformations

change of variables gives target density

$$(T_{\#}q)(x_i) = q(\mathbf{T}^{-1}(x_i)) \cdot \left| \mathbf{T}'\left(\mathbf{T}^{-1}(x_i)\right) \right|$$

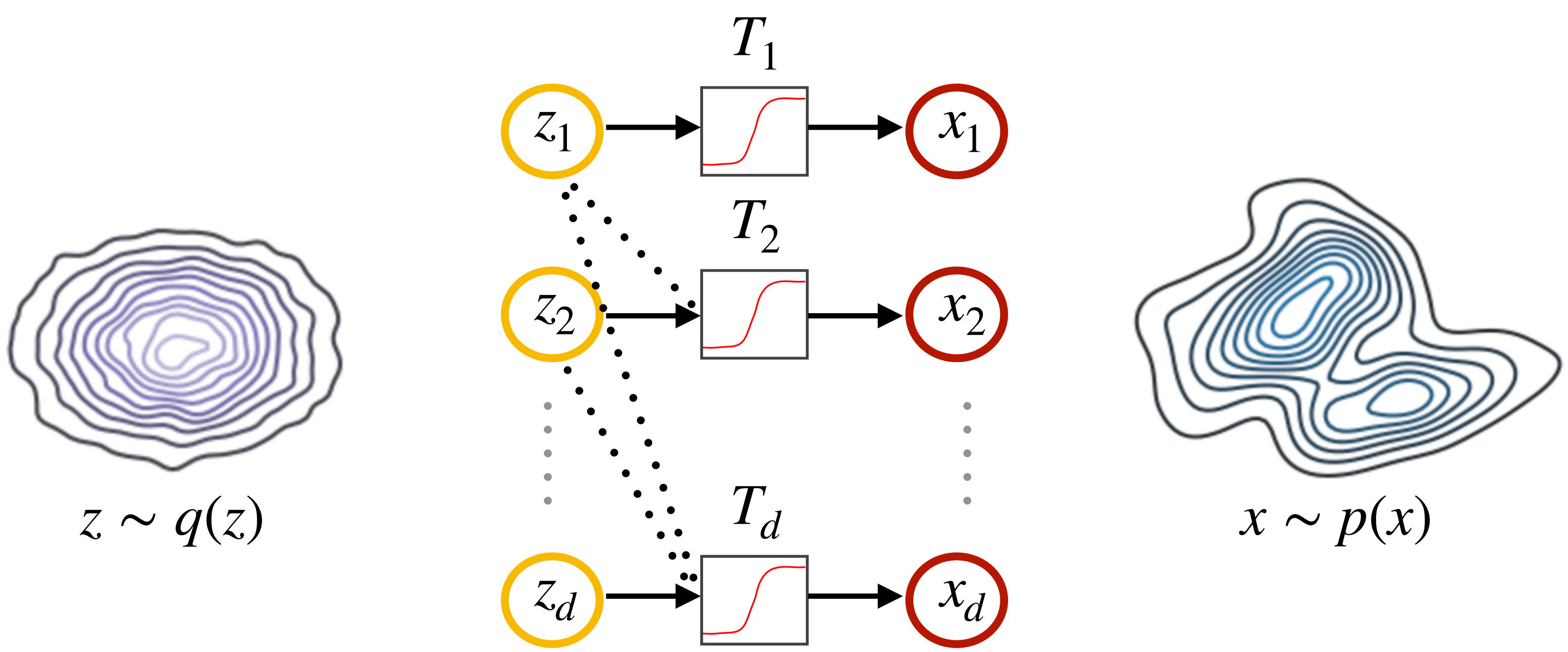
computation of inverse and Jacobian must be *cheap*

always possible via triangular maps

$$\max_{\mathbf{T}} \mathscr{L}(\mathbf{T}) := \max_{\mathbf{T}} \prod_{i=1}^{n} (T_{\#}q)(x_i)$$

unifying framework

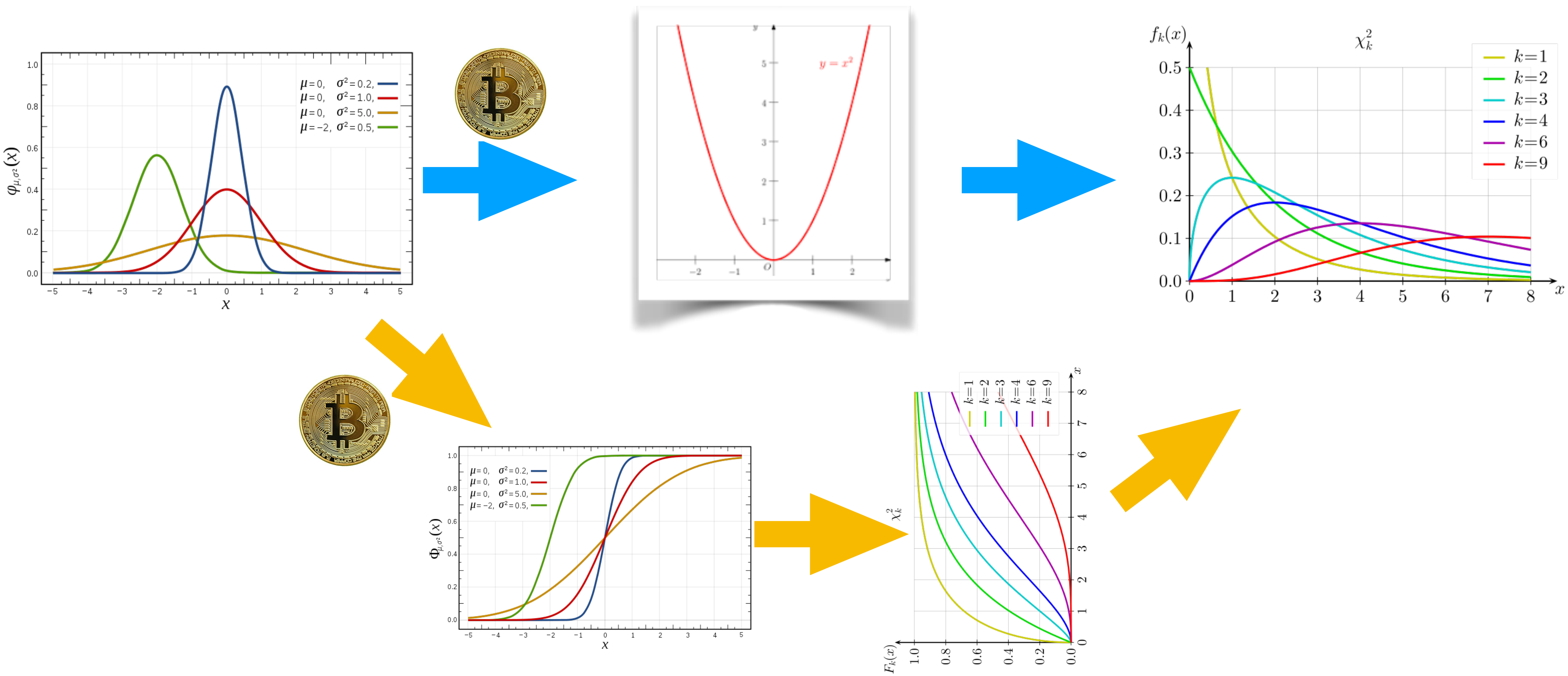
density estimation via increasing triangular maps



In a nutshell

Given simulator for sampling from a normal distribution

How to simulate samples from a chi^2 distribution?



increasing triangular maps

$$\mathbf{T} : \mathbb{R}^d \rightarrow \mathbb{R}^d$$

$$\begin{aligned} x_1 &= T_1(z_1) \\ x_2 &= T_2(z_1, z_2) \\ x_3 &= T_3(z_1, z_2, z_3) \\ &\vdots \\ x_d &= T_d(z_1, z_2, z_3, \ \ldots, z_d) \end{aligned}$$

$$\nabla_{\mathbf{z}} \mathbf{T} = \begin{bmatrix} \frac{\partial T_1}{\partial z_1} & 0 & \ldots & 0 \\ \frac{\partial T_2}{\partial z_1} & \frac{\partial T_2}{\partial z_2} & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial T_d}{\partial z_1} & \frac{\partial T_d}{\partial z_2} & \ldots & \frac{\partial T_d}{\partial z_d} \end{bmatrix}$$

triangular : T_j **is a function of** $z_1, z_2, \ldots, z_j$

increasing : T_j **is increasing w.r.t** z_j

$$\frac{\partial T_j}{\partial z_j} > 0$$

triangular maps

inverse and Jacobian are *easy* to compute

Theorem (paraphrase) : there always exists a unique* increasing triangular map that transforms a source density to a target density

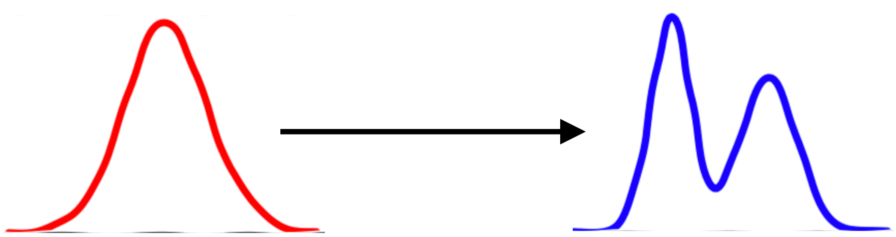
Bogachev, V. et. al. Triangular Transformation of Measures, Sbornik: Mathematics, 2005

***** *for a fixed ordering***

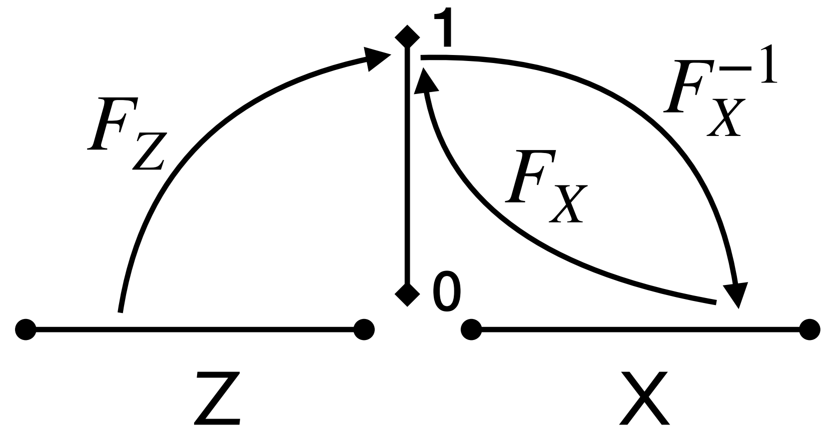
examples

increasing rearrangement

$$T := F_X^{-1} \circ F_Z$$

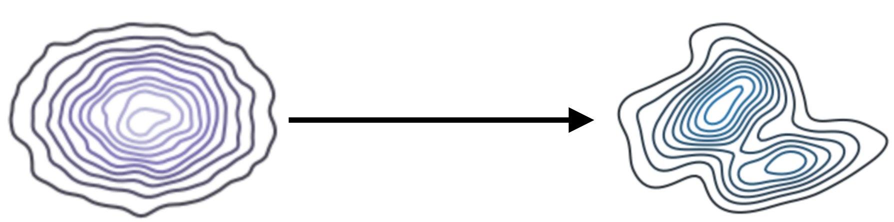
$T:$ 

$Z \sim (q, F_Z)$ $X \sim (p, F_X)$



Knothe-Rosenblatt transformation

iterative application of increasing rearrangement

$T:$ 

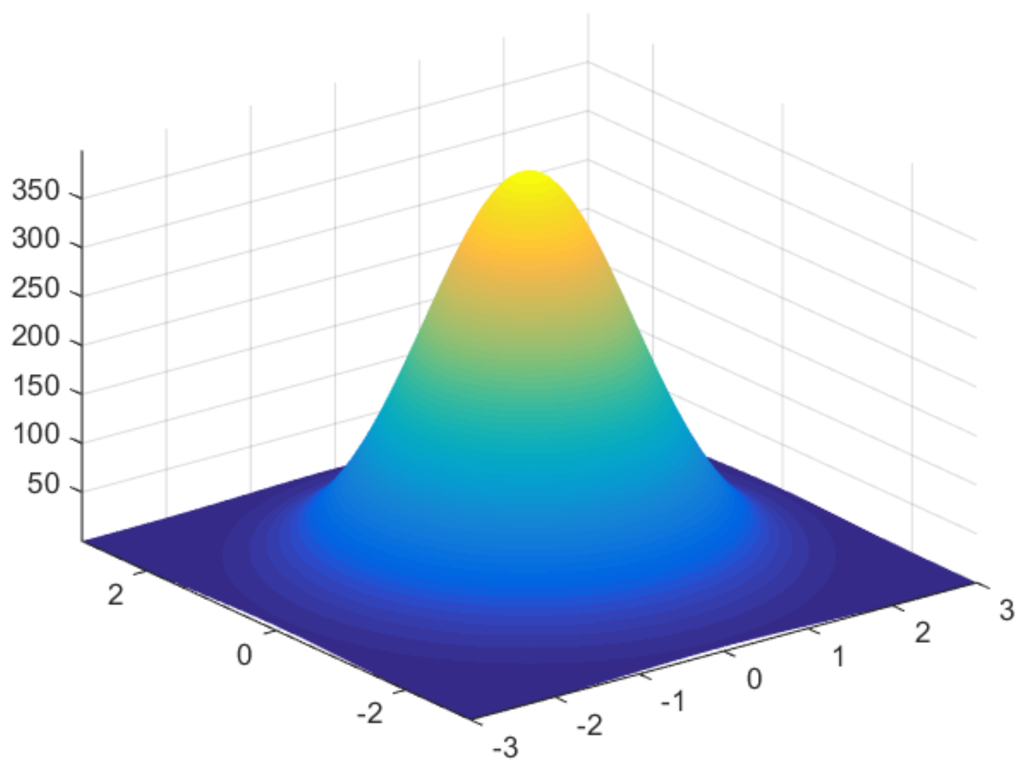
$Z \sim (q, F)$ $X \sim (p, G)$

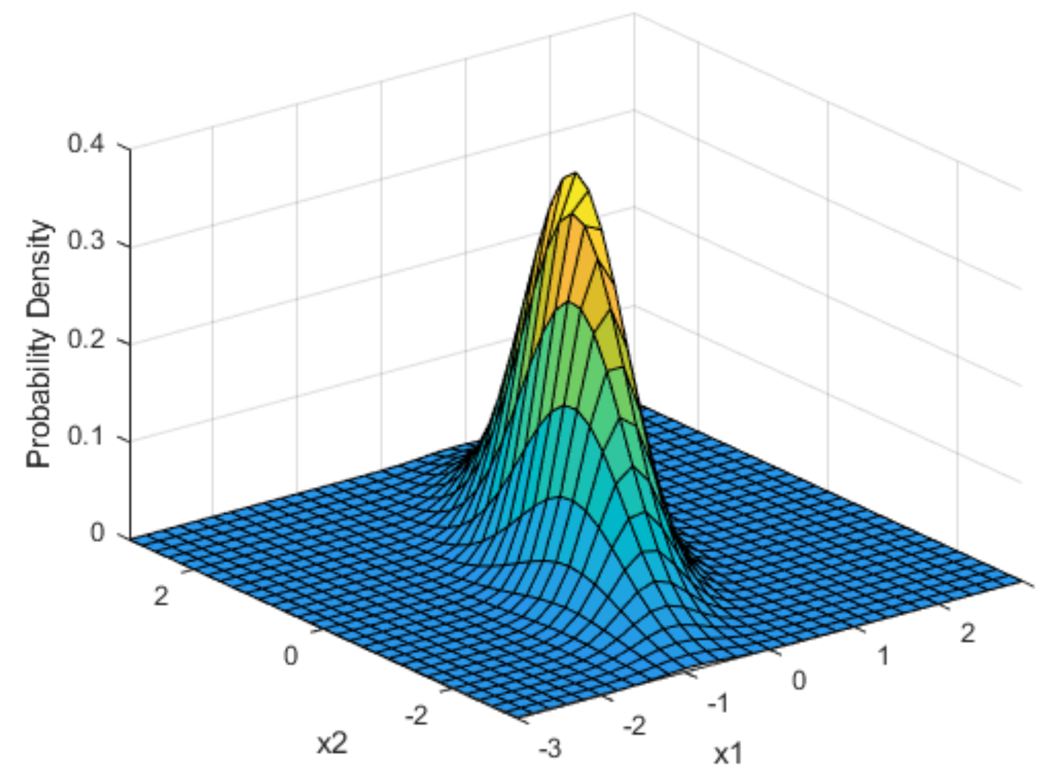
$$T_1(z_1) := G_1^{-1} \circ F_1(z_1)$$

$$T_2(z_2, z_1) := G_{2|1}^{-1} \circ F_{2|1}(z_2)$$

Villani, C. Optimal Transport: Old and New, vol 338, Springer, 2008

More Examples



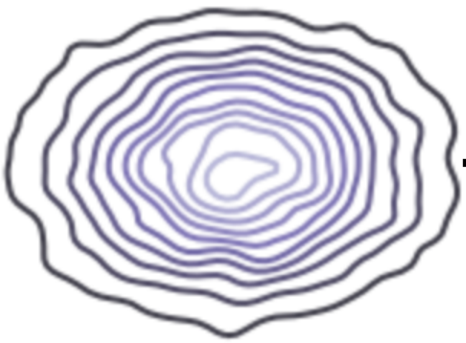


$$TT^\top = \Sigma$$

$$Z \sim \mathcal{N}(0, I) \longrightarrow TZ =: X \sim \mathcal{N}(0, \Sigma)$$

Unique increasing triangular T = chol(Σ)

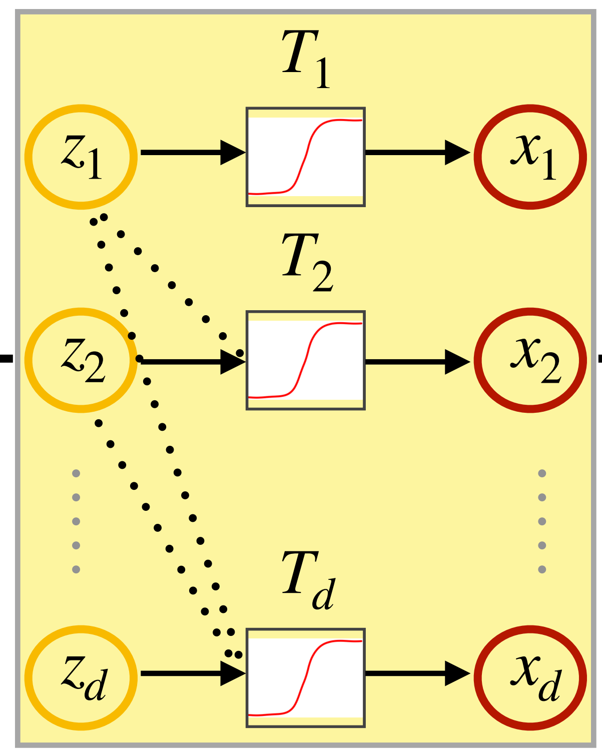
Maximum Likelihood revisited



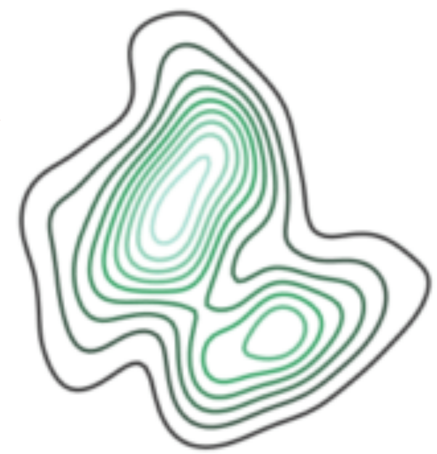
source

$q(\mathbf{z})$

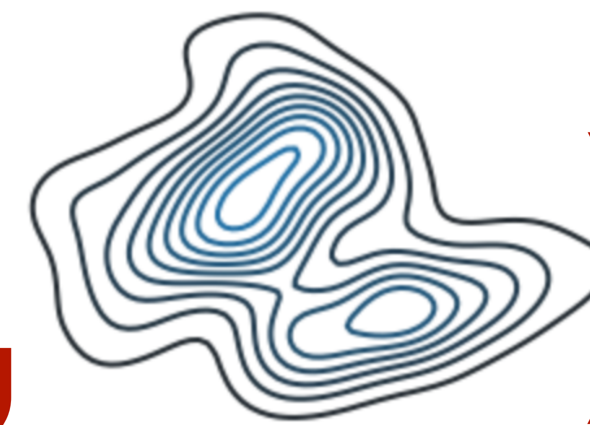
$\mathbf{T}$



D(



,



)

Push-forward

$(\mathbf{T}_\# q)(\mathbf{x})$

=

$$q(\mathbf{T}^{-1}(\mathbf{x})) \cdot \left| \mathbf{T}'\left(\mathbf{T}^{-1}(\mathbf{x}) \right) \right|^{-1}$$

target

$p(\mathbf{x})$

KL

learn T by maximizing likelihood

$$\min_{\mathbf{T}} \sum_{i=1}^{n} \left[-\log q\left(\mathbf{T}^{-1}(\mathbf{x}_i) \right) + \sum_j \log \partial_j T_j\left(\mathbf{T}^{-1}(\mathbf{x}_i) \right) \right]$$

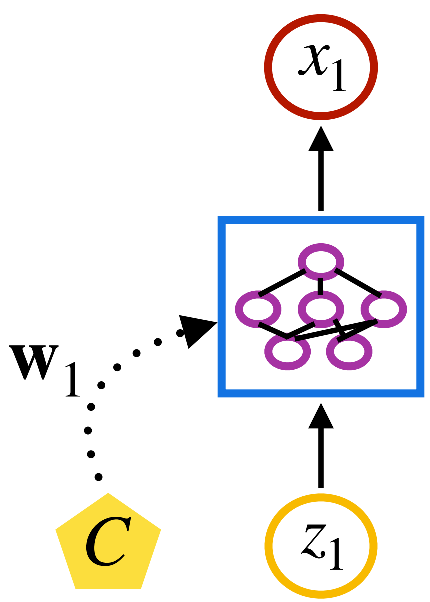
explicitly evaluating $q_\theta(\mathbf{x})$

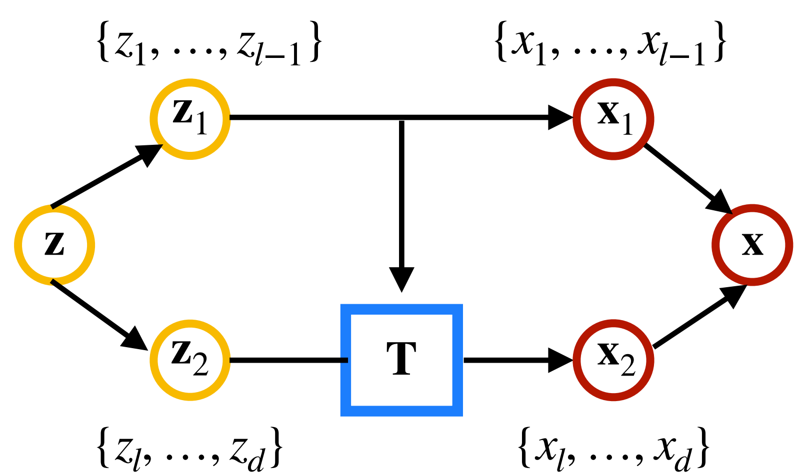
Marzouk, Y. et.al. Sampling via Measure Transport: An Introduction, Springer, 2016

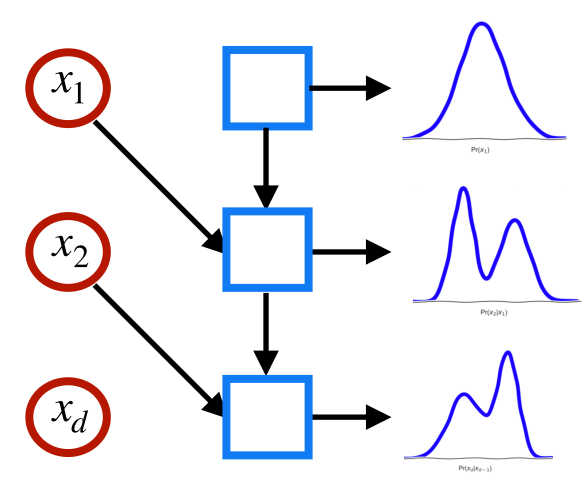
[JSY]. Sum-of-squares Polynomial Flow. ICML, 2019

flow models as triangular maps

study commonalities & differences of flow based methods

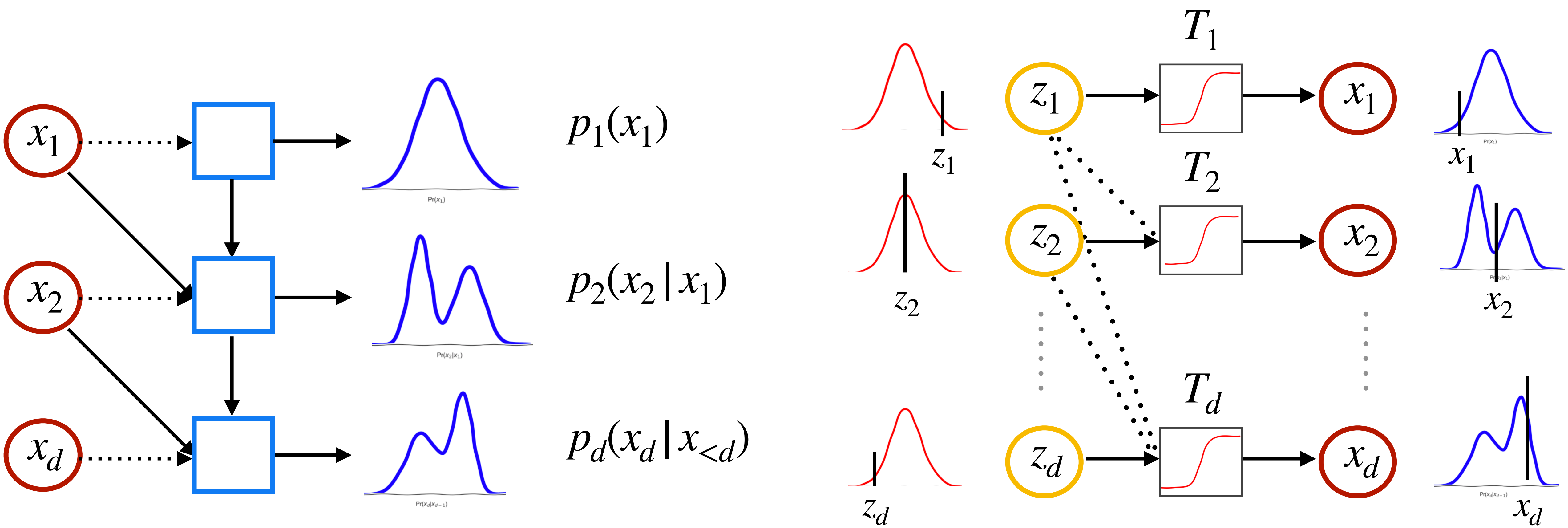






autoregressive models

$$p(x) = p_1(x_1) \cdot p_2(x_2 \mid x_1) \cdot \ldots \cdot p_d(x_d \mid x_{<d})$$



choosing a conditional implicitly fixes a family of triangular maps

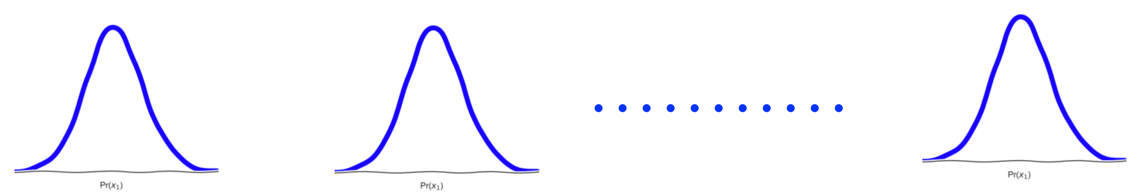
$$x_j = T_j\left(z_j;\ \theta_j(z_{<j})\right)$$

Larochelle, H and Murray, I. The Neural Autoregressive Distribution Estimator, AISTATS, 2011

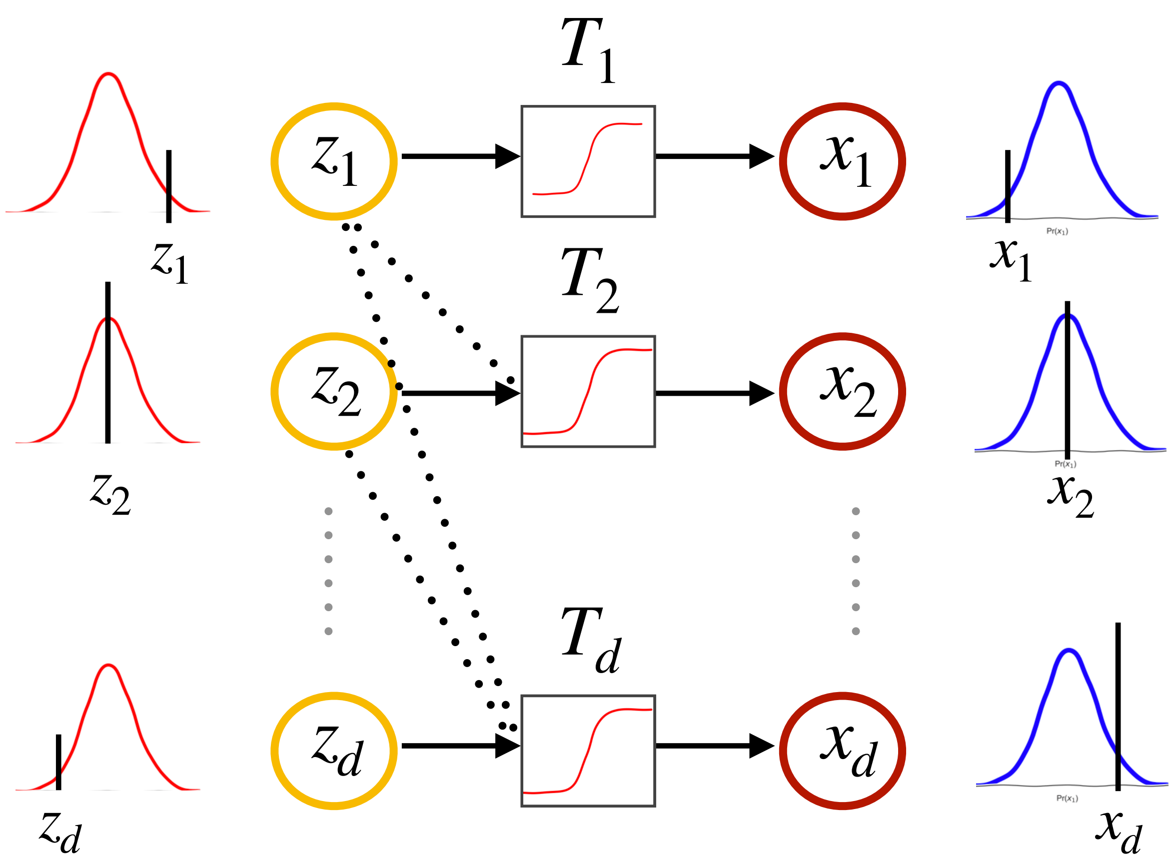
Uria, et.al. Neural Autoregressive Distribution Estimation, JMLR, 2016

AR with Gaussian conditionals

$$p(x) = p_1(x_1) \cdot p_2(x_2 \,|\, x_1) \cdot \ldots \cdot p_d(x_d \,|\, x_{<d})$$



$\mathcal{N}(\mu_1, \sigma_1^2)$ $\mathcal{N}(\mu_2, \sigma_2^2)$ $\mathcal{N}(\mu_d, \sigma_d^2)$



$$x_1 = \sigma_1 \cdot z_1 + \mu_1 =: T_1(z_1)$$

$$x_2 = \sigma_2(z_1) \cdot z_2 + \mu_2(z_1) := T_2(z_2\,; z_1)$$

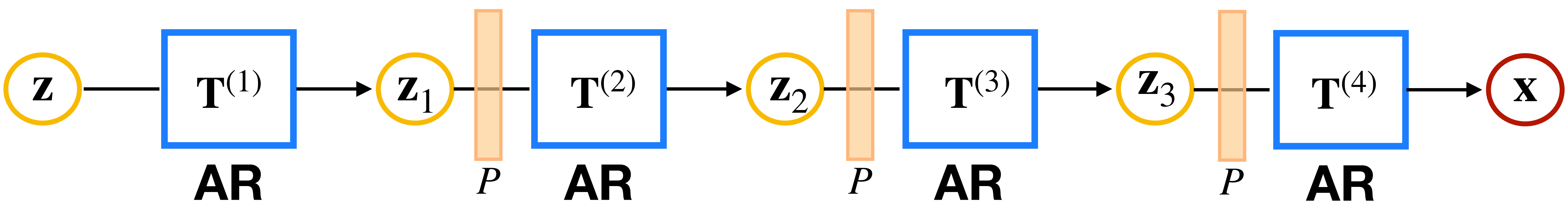
$$\vdots$$

$$x_d = \sigma_d(z_{<d}) \cdot z_d + \mu_d(z_{<d}) := T_d(z_d\,; z_{<d})$$

Kingma, et.al. Improved Variational Inference with Inverse Autoregressive Flow, NeurIPS, 2016

masked autoregressive flows (MAFs)

deep autoregressive flows with Gaussian conditionals*



$$(T_\# q)(x) = q(z) \cdot \left| \nabla \mathbf{T}^{(1)} \right|^{-1} \cdot \left| \nabla \mathbf{T}^{(2)} \right|^{-1} \cdot \left| \nabla \mathbf{T}^{(3)} \right|^{-1} \cdot \left| \nabla \mathbf{T}^{(4)} \right|^{-1}$$

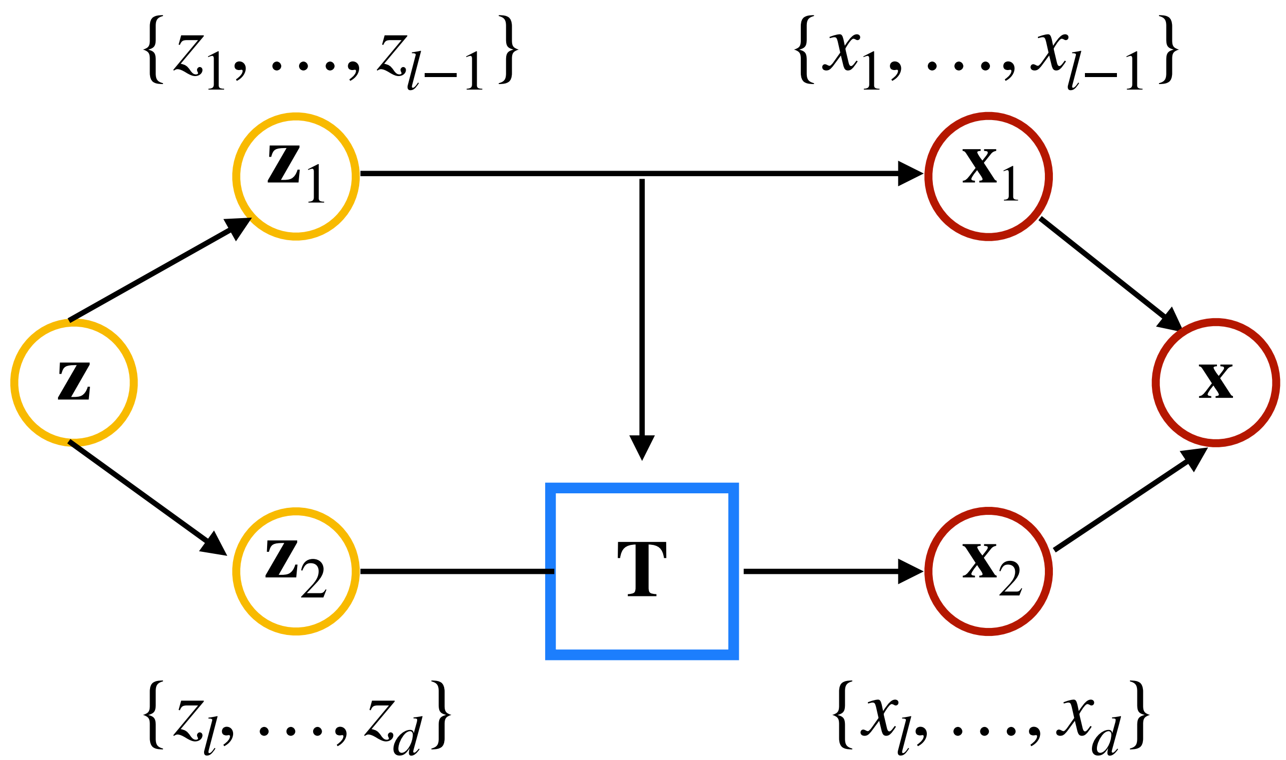
$$x_j = z_j \cdot \exp\big(\alpha_j(z_{<j})\big) + \mu_j(z_{<j}) =: T_j(z_j\ ; z_{<j})$$

triangular maps are fundamental blocks for complex models

Papamakarios, et.al. Masked Autoregressive Flow for Density Estimation, NeurIPS, 2017

* can consider mixture of gaussians

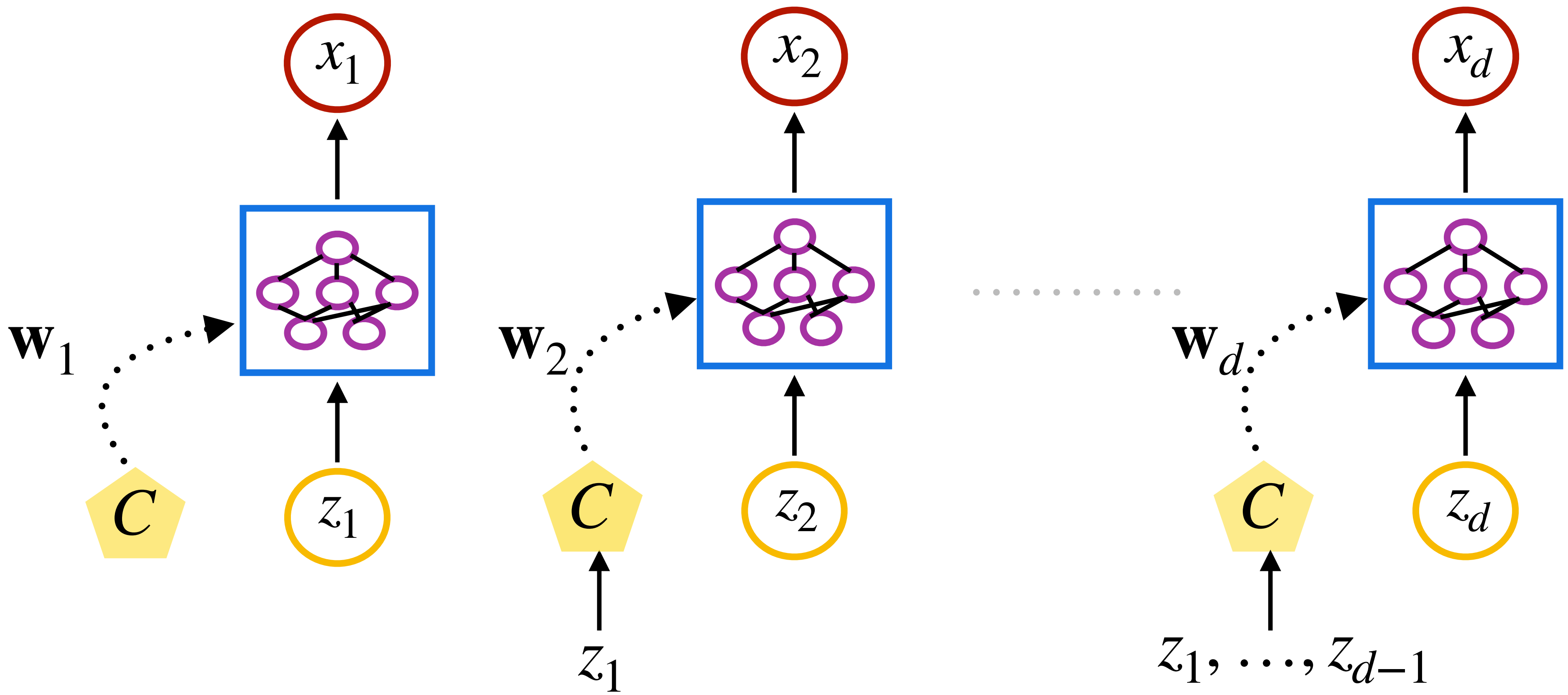
real-NVP



$$T_j(z_j\ ;\ z_{<l}) = \exp\Big(\alpha_j(z_{<l}) \cdot \mathbf{1}_{j \notin [l-1]}\Big) \cdot z_j + \mu_j(z_{<l}) \cdot \mathbf{1}_{j \notin [l-1]}$$

Dinh, et.al. Density estimation using Real NVP, ICLR, 2017

neural autoregressive flows (NAFs)



$$x_j = \textbf{DNN}\left(z_j\ ;\ \mathbf{w}_j(z_{<j})\right) =: T_j(z_j\ ; z_{<j})$$

Strictly positive weights & strictly monotonic activation function ensure that the map is increasing

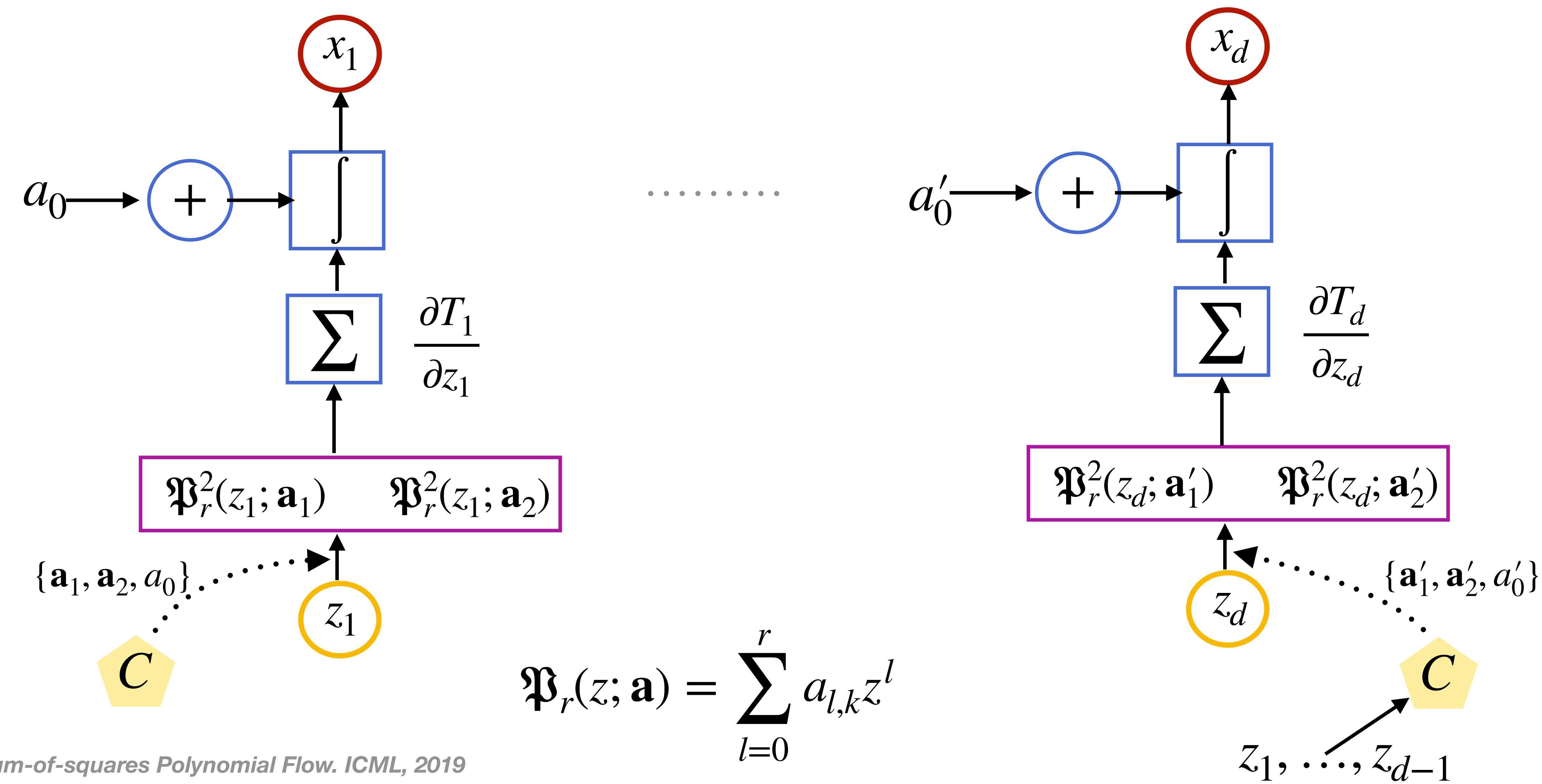
Universal

Huang, et.al. Neural Autoregressive Flows, ICML, 2018

sum-of-squares polynomial flows

goal : learn a universal increasing triangular function

increasing functions $\approx$ primitive of non-negative polynomials $=$ primitive of sum-of-squares of polynomials



$$\mathfrak{P}_r(z; \mathbf{a}) = \sum_{l=0}^{r} a_{l,k} z^l$$

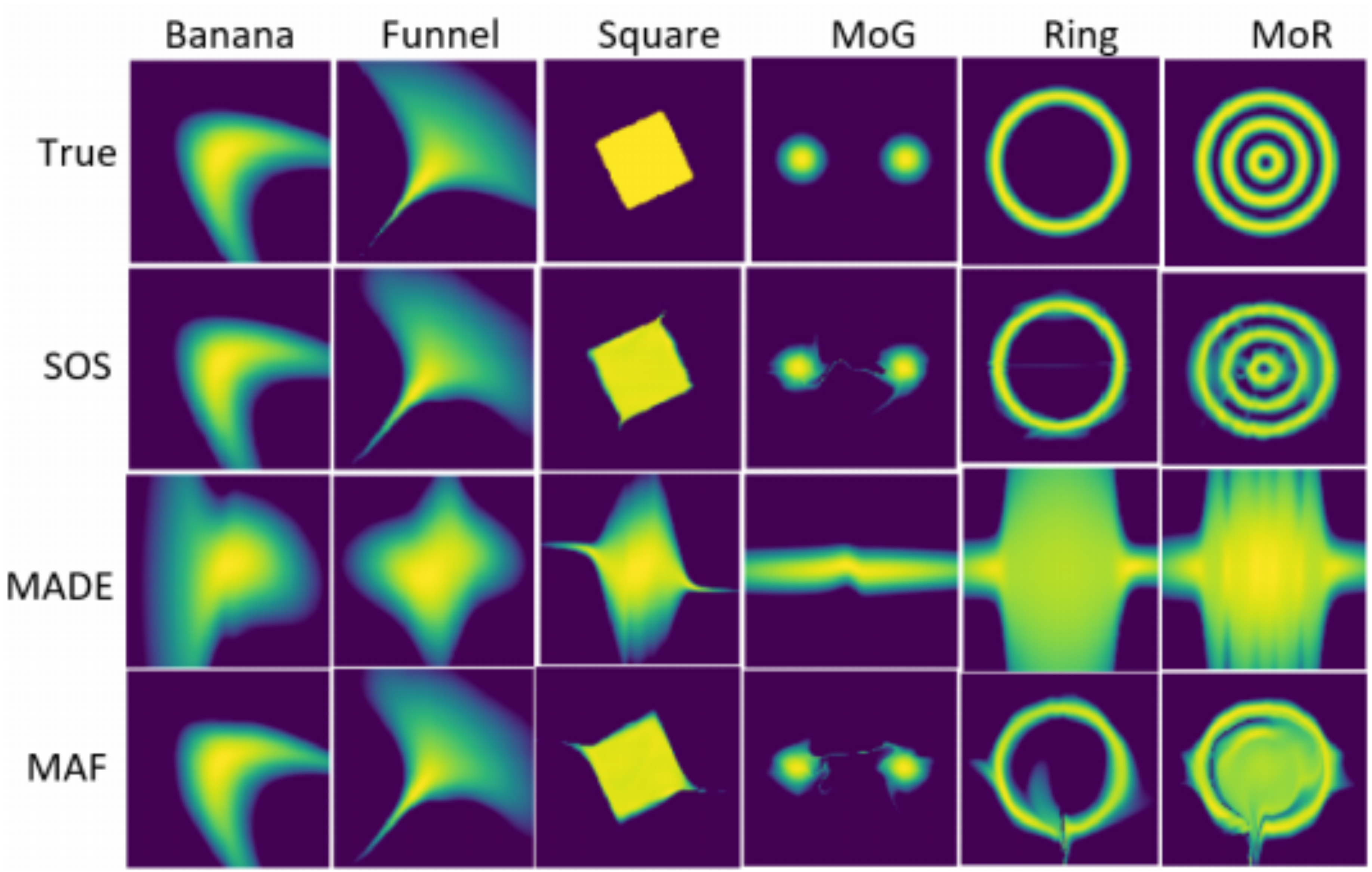
[JSY]. Sum-of-squares Polynomial Flow. ICML, 2019

Summarize

Model	conditioner C_j output	$T_j(z_j\,; C_j(z_1, \ldots, z_{j-1}))$				Δ
Mixture (e.g. McLachlan & Peel, 2004)	$\boldsymbol{\theta}_j$	$S_j(z_j; \boldsymbol{\theta}_j)$	✗	✗	✓	I
(Bengio & Bengio, 1999)	$\boldsymbol{\theta}_j(z_{<j})$	$S_j(z_j; \boldsymbol{\theta}_j)$	✗	✗	?	I
MADE (Germain et al., 2015)	$\boldsymbol{\theta}_j(z_{<j})$	$S_j(z_j; \boldsymbol{\theta}_j)$	✓	✓	?	I
NICE (Dinh et al., 2015)	$\mu_j(z_{<l})$	$z_j + \mu_j \cdot \mathbf{1}_{j \notin [l]}$	✗	✗	?	E
NADE (Uria et al., 2016)	$\boldsymbol{\theta}_j(z_{<j})$	$S_j(z_j; \boldsymbol{\theta}_j)$	✓	✗	?	I
IAF (Kingma et al., 2016)	$\sigma_j(z_{<j}),\ \mu_j(z_{<j})$	$\sigma_j z_j + (1 - \sigma_j)\mu_j$	✓	✓	?	E
MAF (Papamakarios et al., 2017)	$\alpha_j(z_{<j}),\ \mu_j(z_{<j})$	$z_j \exp(\alpha_j) + \mu_j$	✓	✓	?	E
Real-NVP (Dinh et al., 2017)	$\alpha_j(z_{<l}), \mu_j(z_{<l})$	$\exp(\alpha_j \cdot \mathbf{1}_{j \notin [l]}) \cdot z_j + \mu_j \cdot \mathbf{1}_{j \notin [l]}$	✗	✗	?	E
NAF (Huang et al., 2018)	$\mathbf{w}_j(z_{<j})$	$\mathrm{DNN}(z_j\,; \mathbf{w}_j)$	✓	✓	✓	E
SOS	$\mathbf{a}_j(z_{<j})$	$\mathfrak{P}_{2r+1}(z_j; \mathbf{a}_j)$	✓	✓	✓	E

[JSY]. Sum-of-squares Polynomial Flow. ICML, 2019

Toy examples



Germain, et.al. MADE: Masked Autoencoder for Density Estimation, ICML, 2015
Papamakarios, et.al. Masked Autoregressive Flow for Density Estimation, NeurIPS, 2017
Oliva, et.al. Transformation Autoregressive Networks, ICML, 2018
Huang, et.al. Neural Autoregressive Flows, ICML, 2018

Effect of ordering

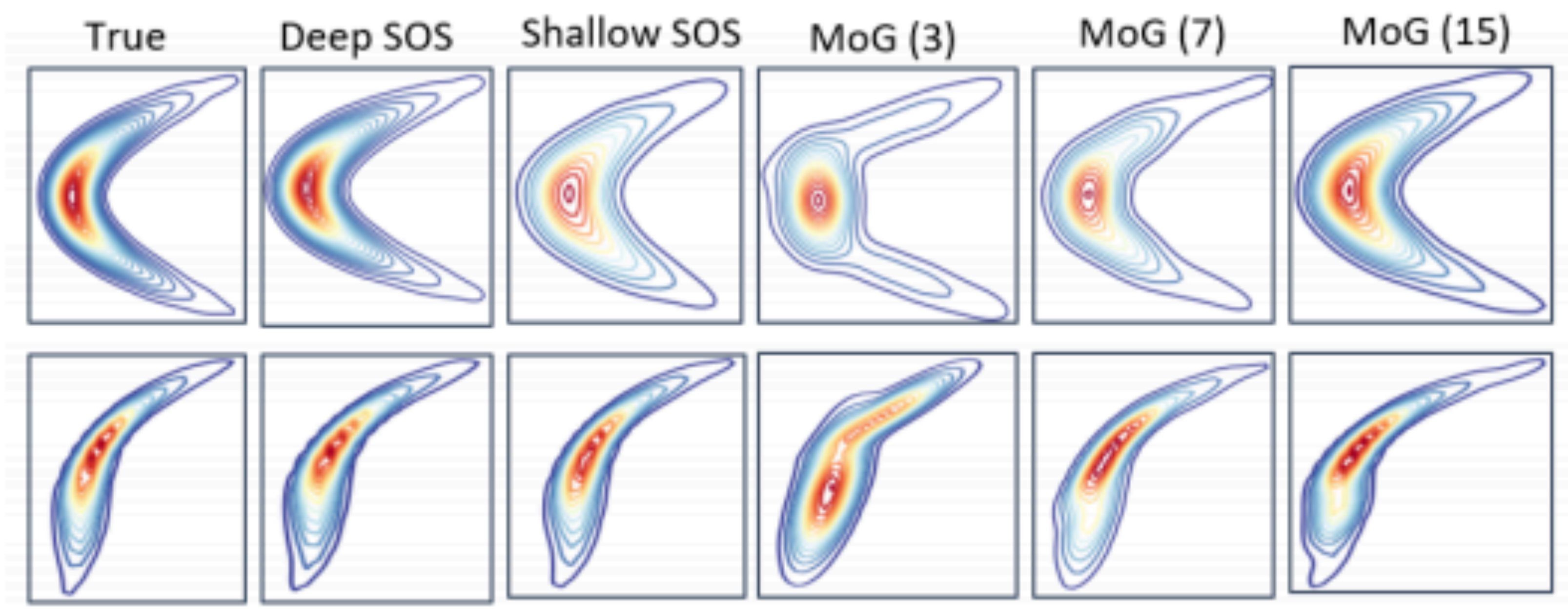
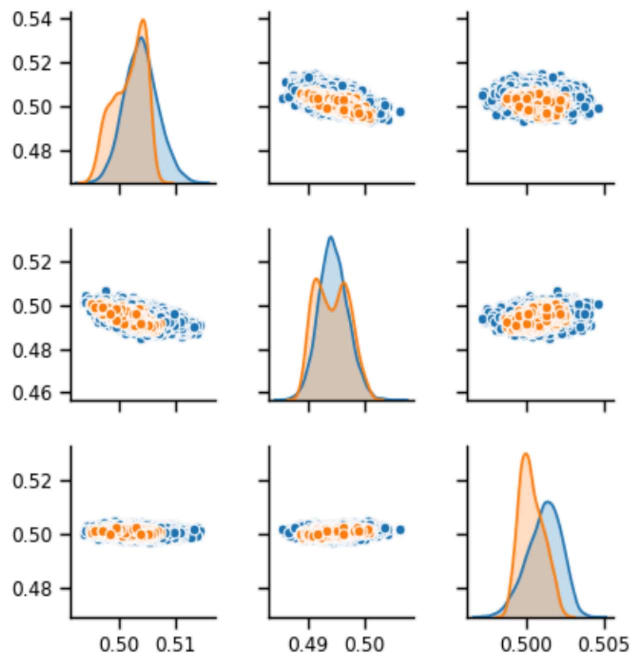
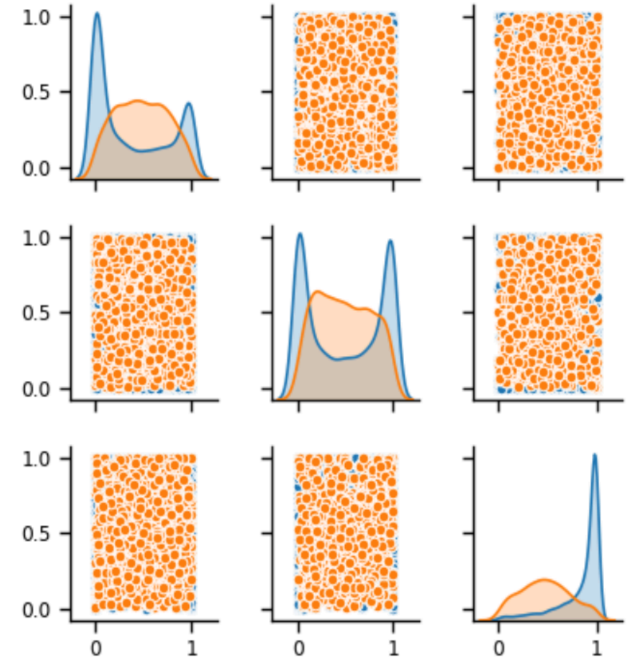


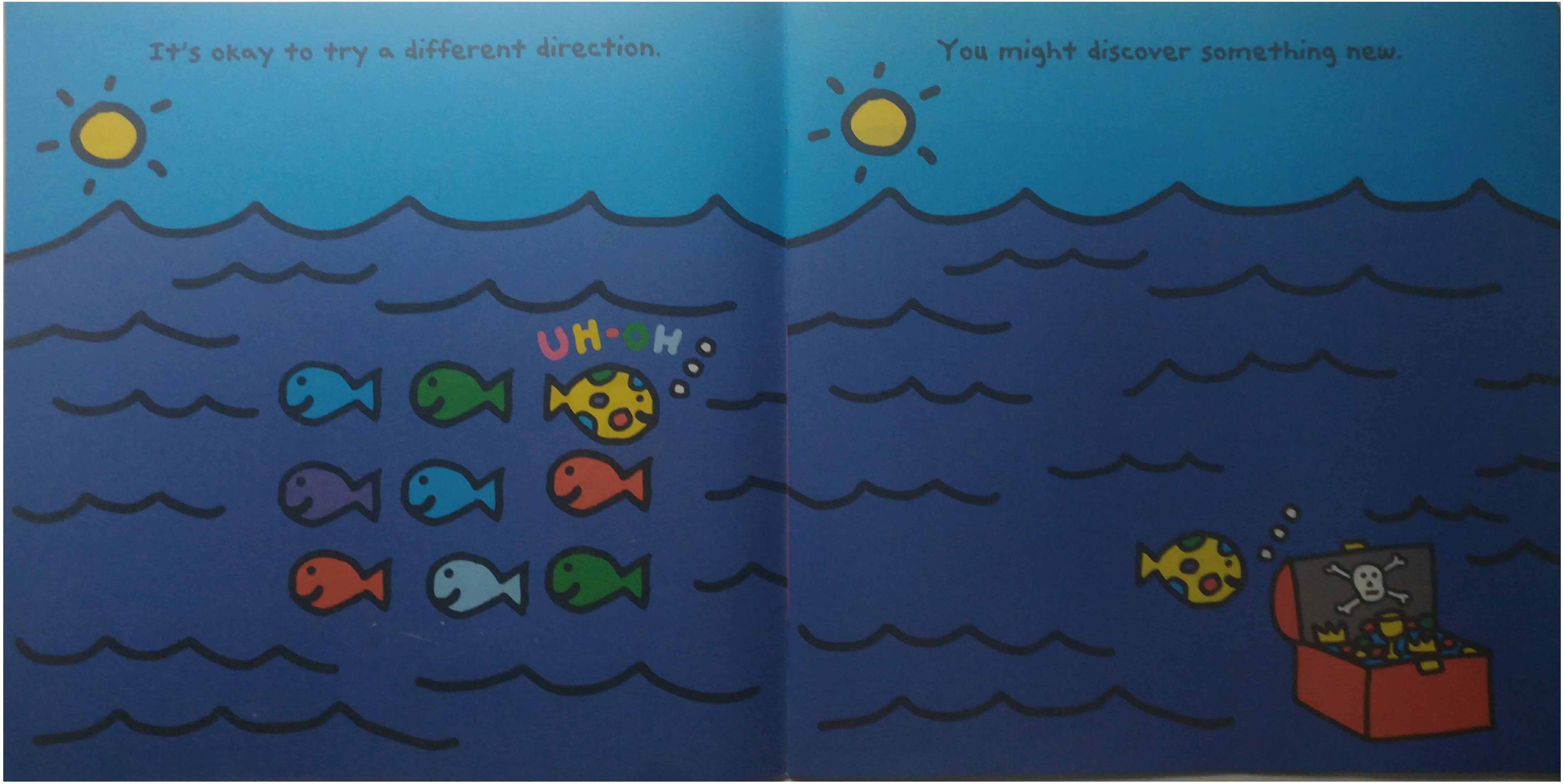
Figure 4. **Top:** Left plot shows the target density given by $p(x_1, x_2) = \mathcal{N}(x_2\,; 0, 4)\mathcal{N}(x_1\,; 0.25x_2^2, 1)$. The second plot shows the density learnt by SOS flows with 3 blocks and a sum of 2 polynomials with degree 3 with ordering (x_1, x_2). Third plot shows the density learnt by SOS flows with 1 block and a sum of 2 polynomials with degree 4 and ordering (x_1, x_2). The last three plots estimate this density using a Mixture of Gaussian conditionals with varying components given in parenthesis and ordering (x_1, x_2). **Bottom:** Same as Top but with target density given by $p(x_1, x_2) = \mathcal{N}(x_2\,; 2, 2)\mathcal{N}(x_1\,; 0.33x_1^3, 1.5)$.

Application to novelty detection

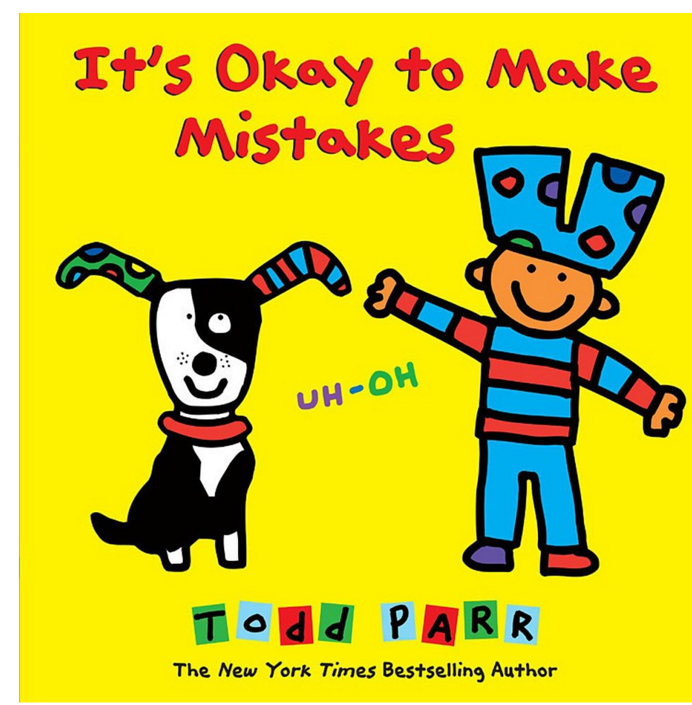
Multivariate triangular quantile maps





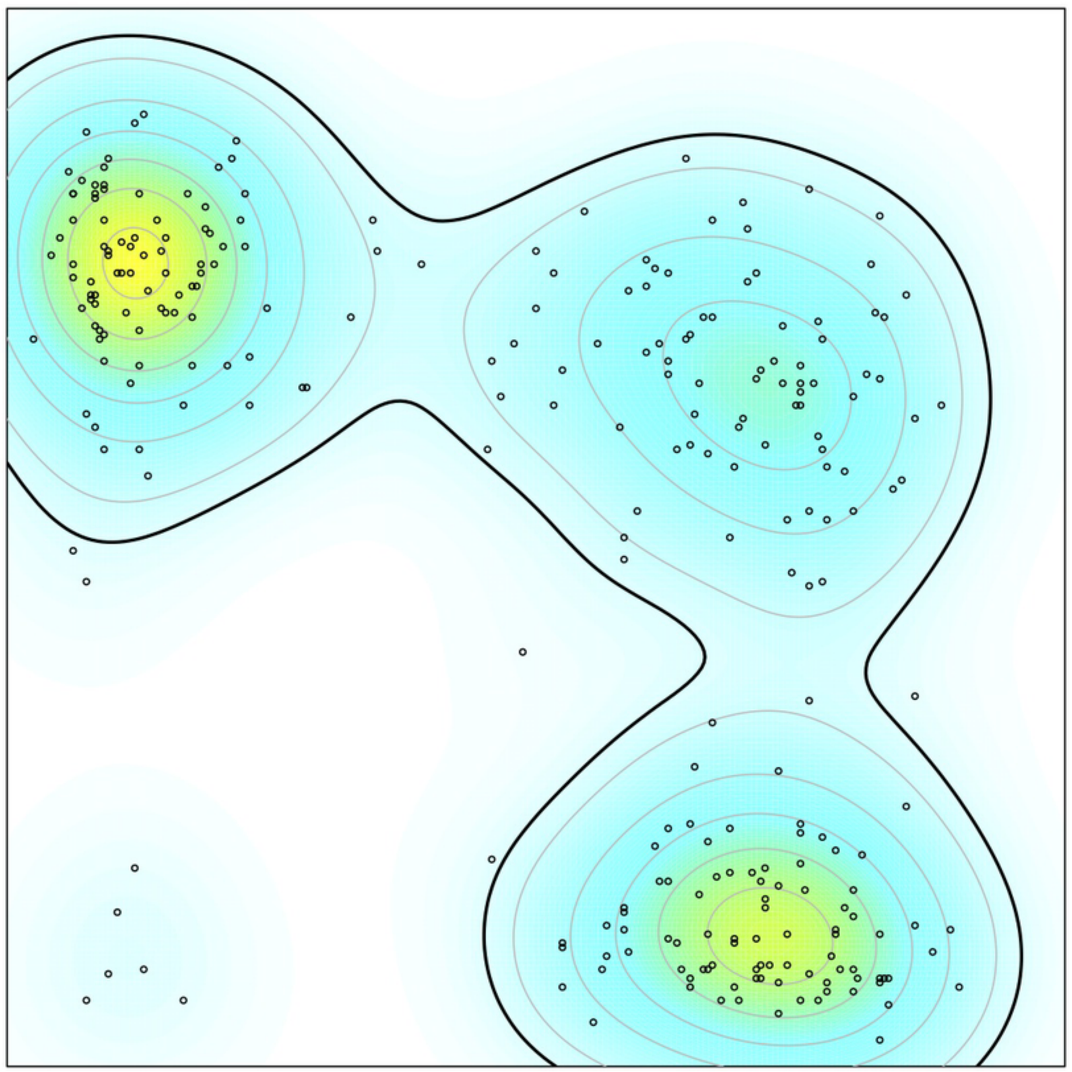


during training only nominal data is available.

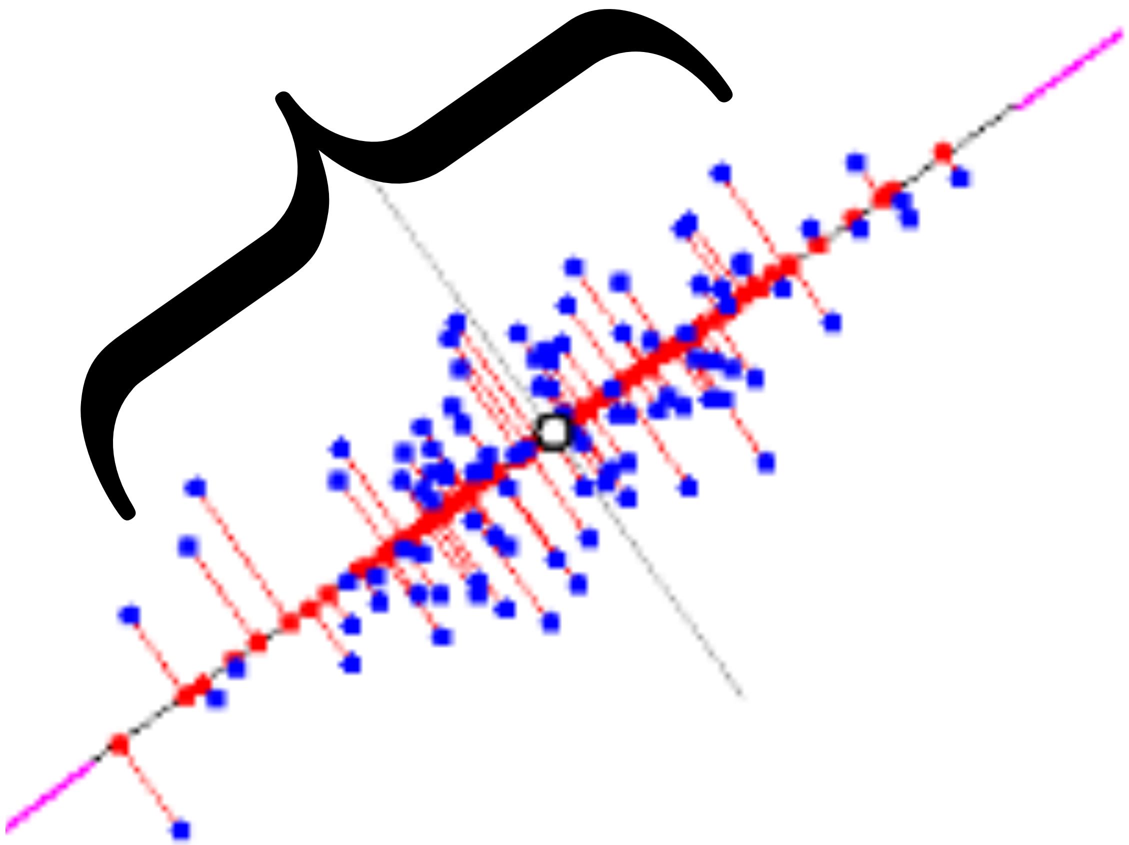


Two Approaches, One Idea

Novelty ≈ Low density region



Novelty = $[\![-\hat{p} > -\alpha]\!]$



Novelty = $[\![\, |\hat{Q}^{-1} - \frac{1}{2}| > \alpha]\!]$

Ben-David and Lindenbaum. Learning Distributions by Their Density Levels: A Paradigm for Learning without a Teacher. JCSS 1997.

Steinwart, Hush and Scovel. A classification framework for anomaly detection. JMLR, 2005.

Schölkopf, Platt, Shawe-Taylor, Smola and Williamson. Estimating the Support of a High-Dimensional Distribution. Neural Computation, 2001.

Takeda and Sugiyama. v-Support Vector Machine as Conditional Value-at-Risk Minimization. ICML, 2008.

Triangular Quantile Map

Let $\mathbf{U} \sim \mathrm{Uniform}[0,1]^d$ and $\mathbf{X} \in \mathbb{R}^d$ any random vector. We call the increasing triangular map $\mathbf{Q} = \mathbf{Q}_{\mathbf{X}} : [0,1]^d \to \mathbb{R}^d$ the triangular quantile map of $\mathbf{X}$ if $\mathbf{Q}(\mathbf{U}) \sim \mathbf{X}$.

Composable!

Let $\mathbf{Y} = \mathbf{T}(\mathbf{X})$ for some increasing triangular map $\mathbf{T}$. Then, $\mathbf{Q}_{\mathbf{Y}} = \mathbf{T} \circ \mathbf{Q}_{\mathbf{X}}$.

- d=1: usual definition of quantile (inverse of cdf), advocated in (Parzen 1979)
- Precursors in Rosenblatt, Knothe, Ruschendorf, Decurninge …
- Other multivariate quantiles exist (e.g. Chernozhukov et al)

[WSY]. Multivariate Triangular Quantiles for Novelty Detection. NeurIPS, 2019

One Stone, Two Birds

Novelty $\approx$ Low density region

Regularization

$$\min_{\mathbf{f},\mathbf{Q}} \quad \gamma \mathsf{KL}(\mathbf{f}_{\#}p \| \mathbf{Q}_{\#}q) + \lambda \ell(\mathbf{f}) + \zeta g(\mathbf{Q})$$

density/quantile

dim reduction

[WSY]. *Multivariate Triangular Quantiles for Novelty Detection. NeurIPS, 2019*

Implementation

$$\min_{\mathbf{f},\mathbf{Q}} \quad \gamma \mathrm{KL}(\mathbf{f}_{\#}p \| \mathbf{Q}_{\#}q) + \lambda \ell(\mathbf{f}) + \zeta g(\mathbf{Q})$$

Parameterize Q using SOS flow

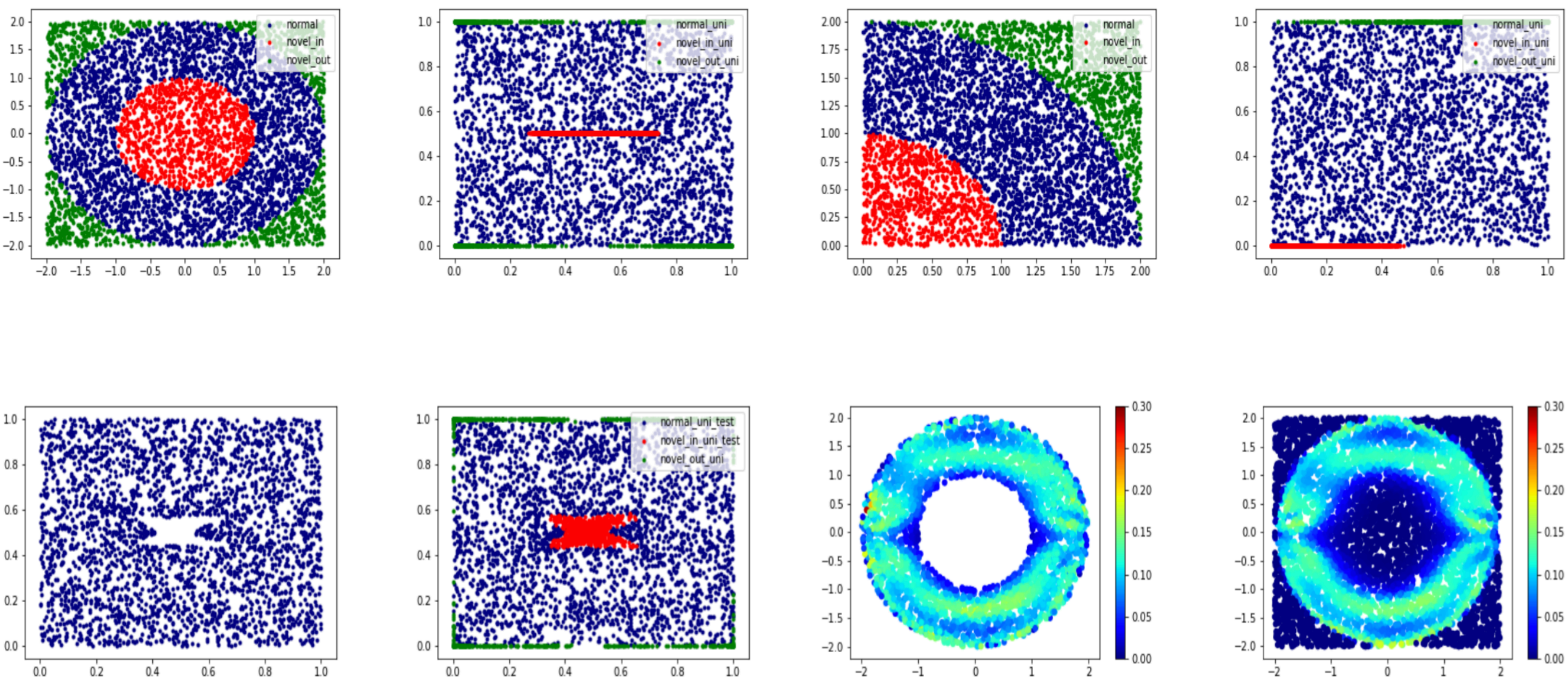
Solve by multiple gradient descent (no parameter tuning)

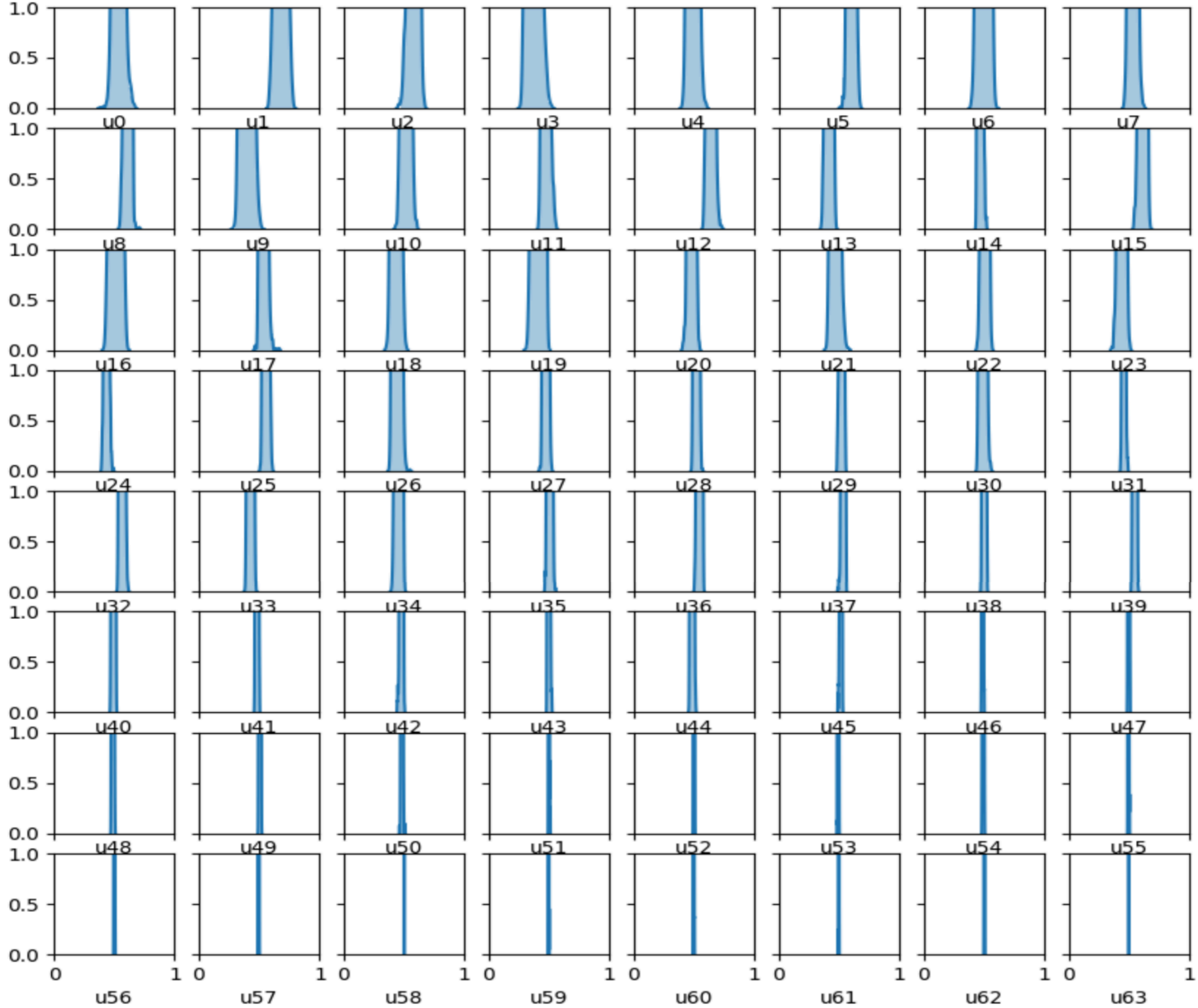
Dimensionality reduction $\mathbf{Z} = \mathbf{f}(\mathbf{X})$

- **Density:** $\log |\mathbf{Q}'(\mathbf{Q}^{-1}(\mathbf{Z}))| + \|\mathbf{Q}^{-1}(\mathbf{Z})\|^2/2 > \alpha$
- **Quantile:** $\|\mathbf{Q}^{-1}(\mathbf{Z})) - \frac{1}{2}\|_{\infty} > \alpha$

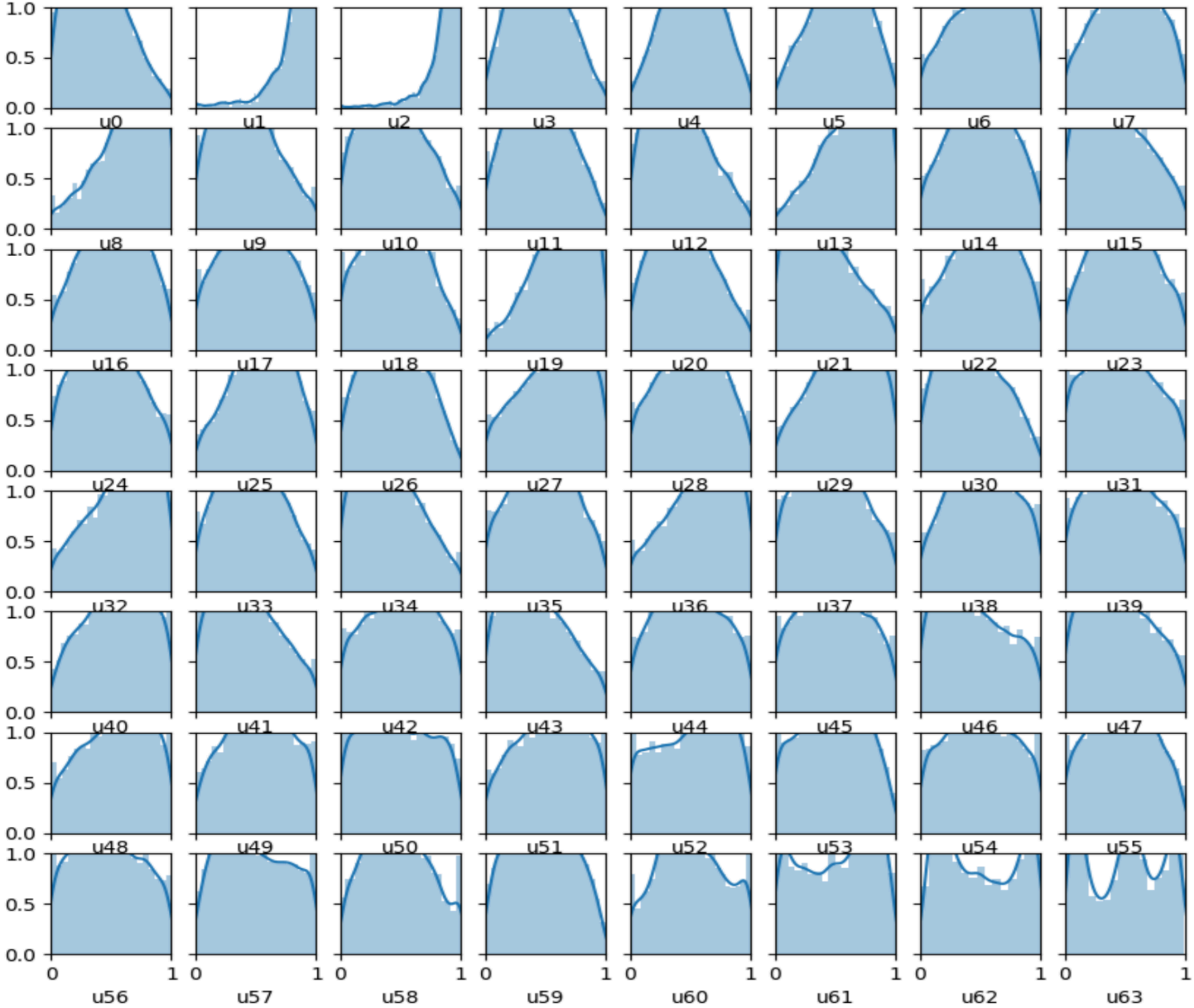
can be tuned

Donut

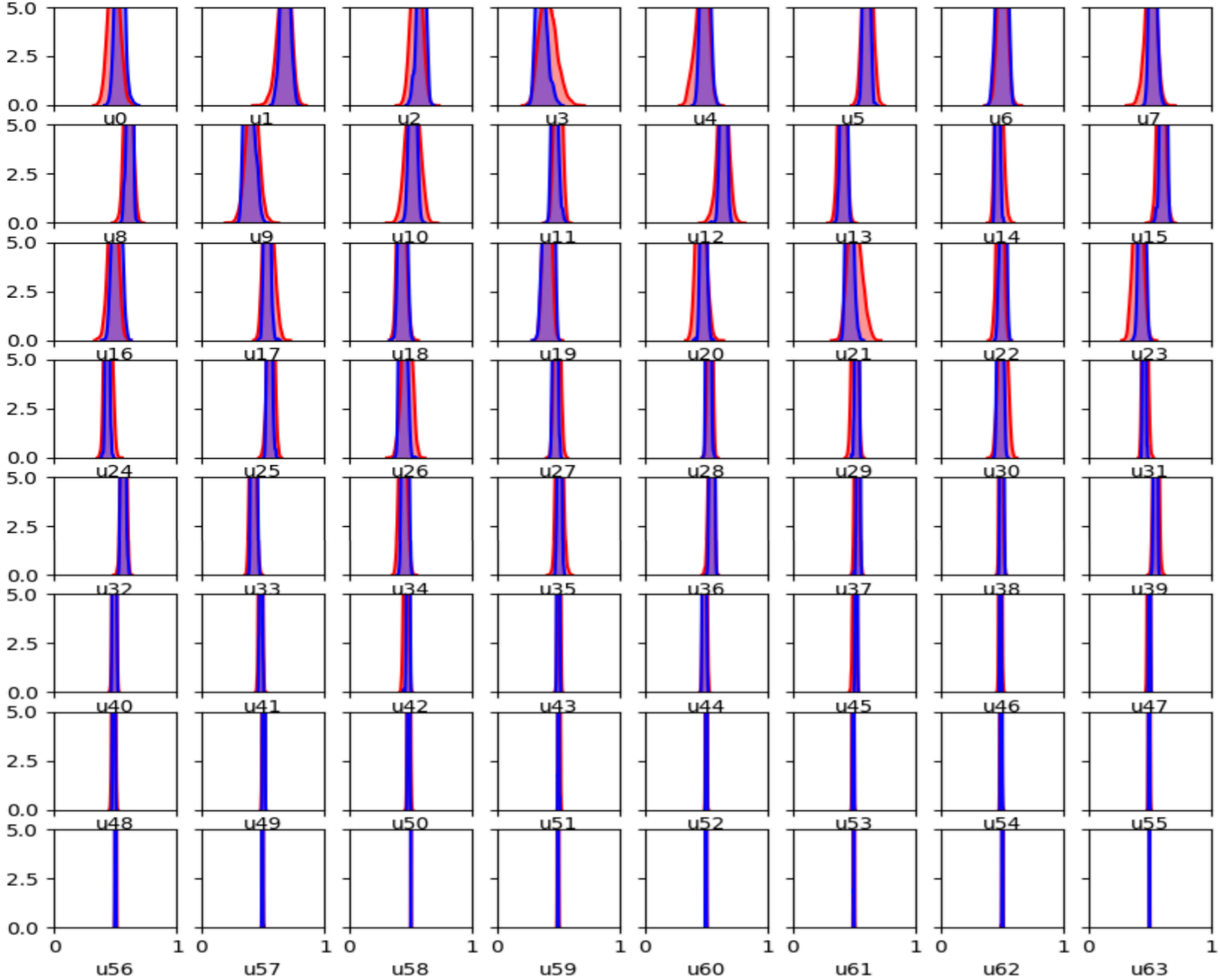

normal
novel_in
novel_out
normal_uni
novel_in_uni
novel_out_uni
normal
novel_in
novel_out
normal_uni
novel_in_uni
novel_out_uni
normal_uni_test
novel_in_uni_test
novel_out_uni

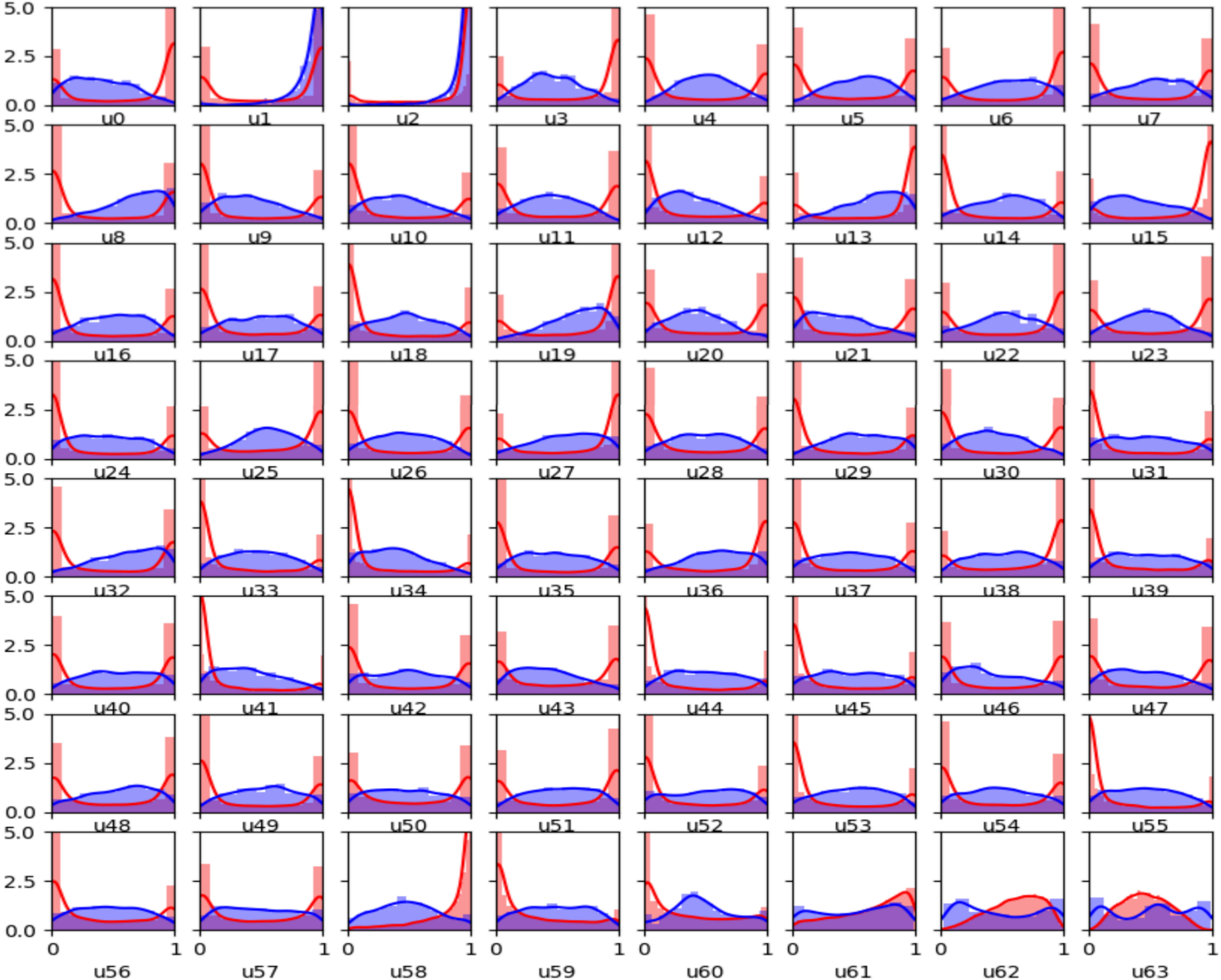


1.0
0.5
0.0
0
1
u0
u1
u2
u3
u4
u5
u6
u7
u8
u9
u10
u11
u12
u13
u14
u15
u16
u17
u18
u19
u20
u21
u22
u23
u24
u25
u26
u27
u28
u29
u30
u31
u32
u33
u34
u35
u36
u37
u38
u39
u40
u41
u42
u43
u44
u45
u46
u47
u48
u49
u50
u51
u52
u53
u54
u55
u56
u57
u58
u59
u60
u61
u62
u63

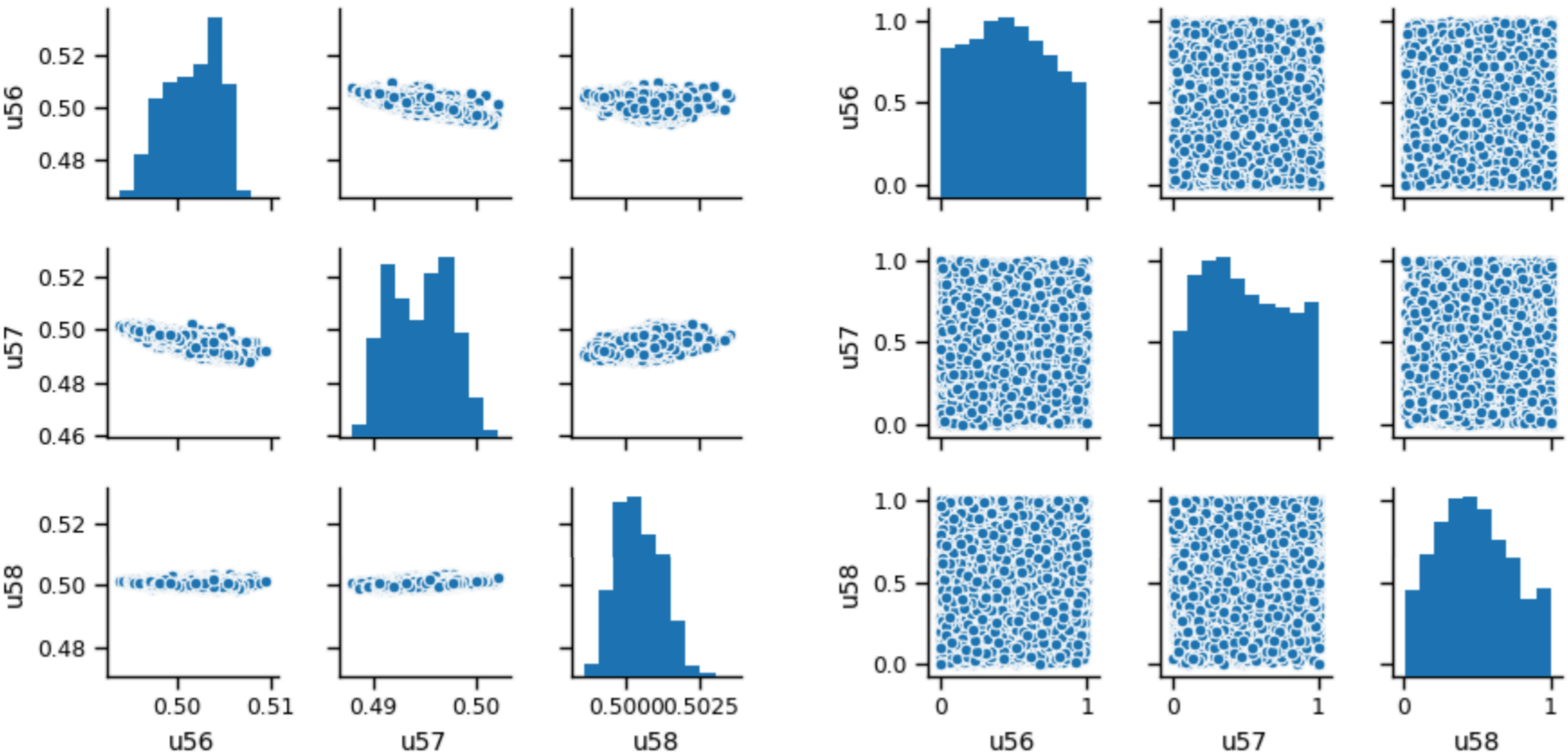


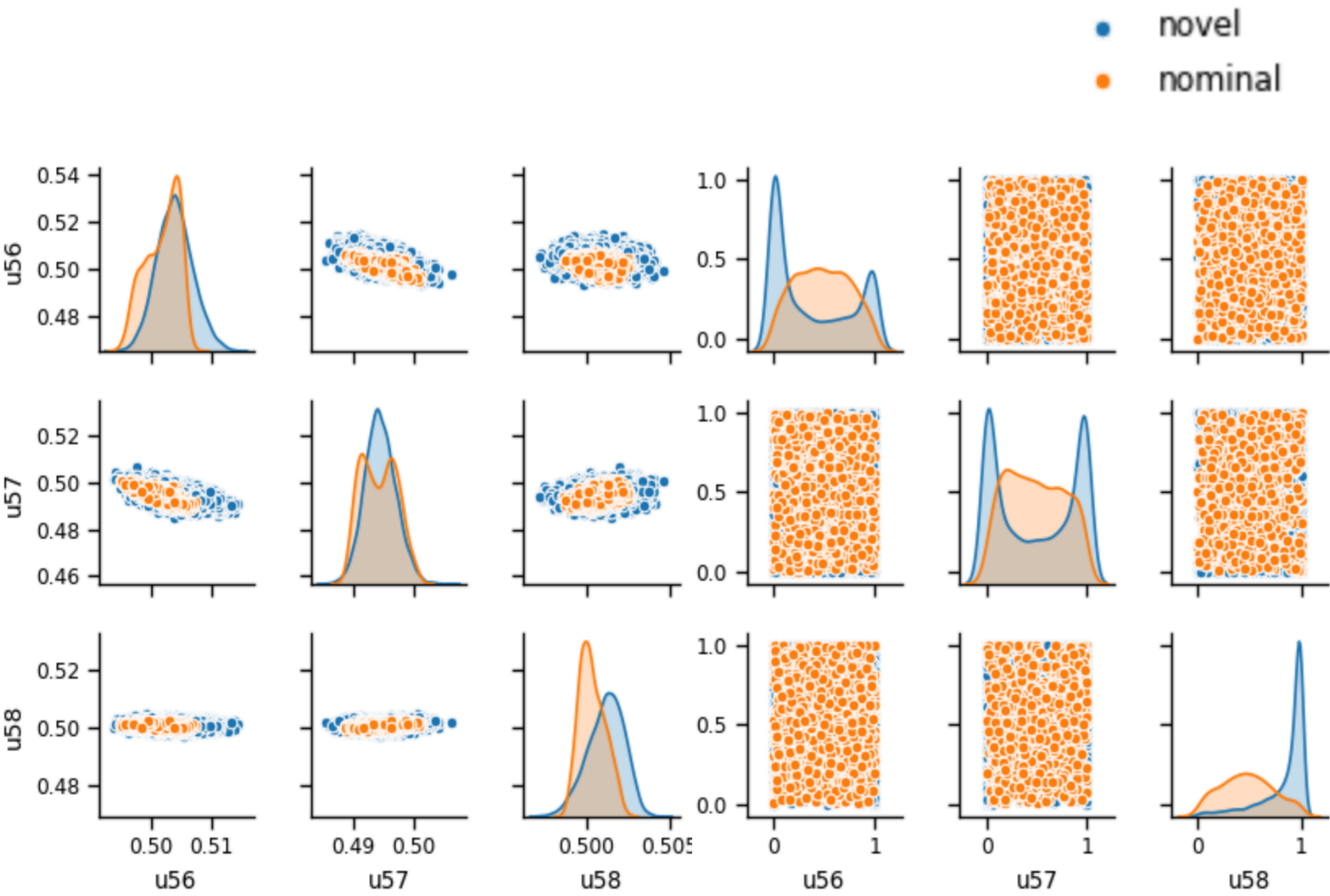
u0
u1
u2
u3
u4
u5
u6
u7
u8
u9
u10
u11
u12
u13
u14
u15
u16
u17
u18
u19
u20
u21
u22
u23
u24
u25
u26
u27
u28
u29
u30
u31
u32
u33
u34
u35
u36
u37
u38
u39
u40
u41
u42
u43
u44
u45
u46
u47
u48
u49
u50
u51
u52
u53
u54
u55
u56
u57
u58
u59
u60
u61
u62
u63


5.0
2.5
0.0
0
1
u0
u1
u2
u3
u4
u5
u6
u7
u8
u9
u10
u11
u12
u13
u14
u15
u16
u17
u18
u19
u20
u21
u22
u23
u24
u25
u26
u27
u28
u29
u30
u31
u32
u33
u34
u35
u36
u37
u38
u39
u40
u41
u42
u43
u44
u45
u46
u47
u48
u49
u50
u51
u52
u53
u54
u55
u56
u57
u58
u59
u60
u61
u62
u63



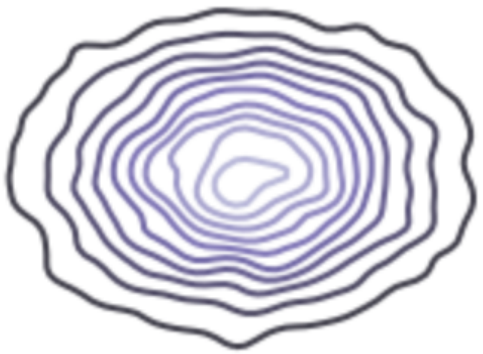
5.0
2.5
0.0
0
1
u0
u1
u2
u3
u4
u5
u6
u7
u8
u9
u10
u11
u12
u13
u14
u15
u16
u17
u18
u19
u20
u21
u22
u23
u24
u25
u26
u27
u28
u29
u30
u31
u32
u33
u34
u35
u36
u37
u38
u39
u40
u41
u42
u43
u44
u45
u46
u47
u48
u49
u50
u51
u52
u53
u54
u55
u56
u57
u58
u59
u60
u61
u62
u63


u56
u57
u58
0.52
0.50
0.48
0.46
0.50
0.51
0.49
0.50000.5025
1.0
0.5
0.0
0
1

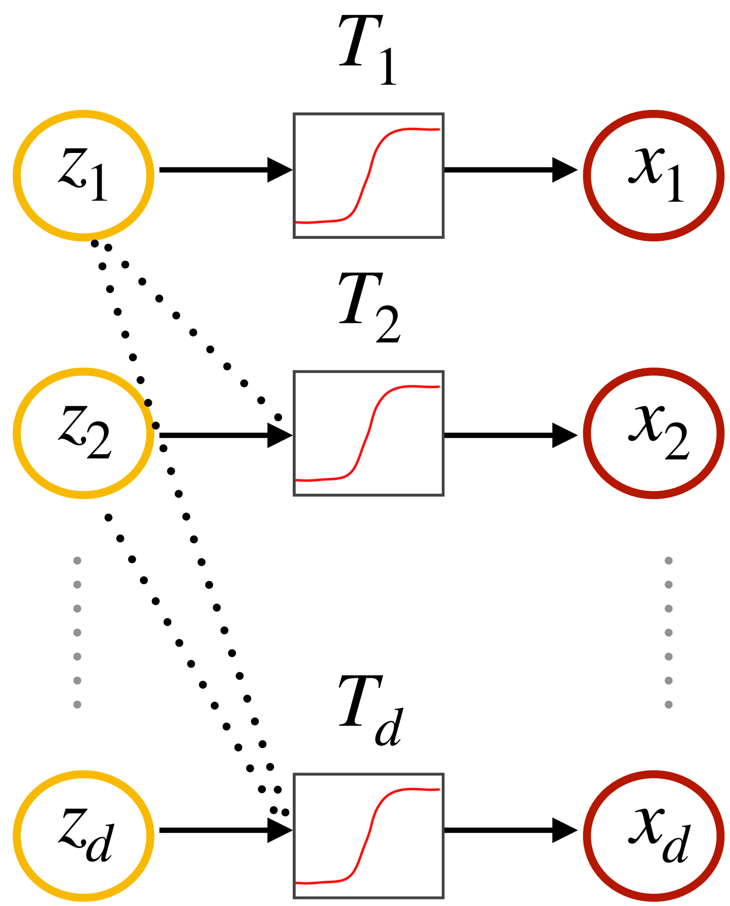


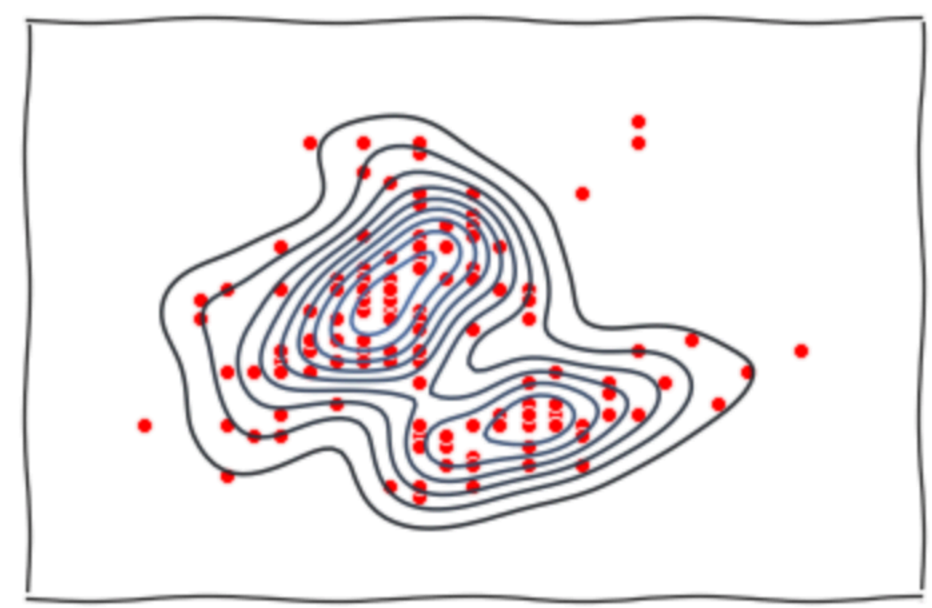
novel
nominal
u56
u57
u58
0.54
0.52
0.50
0.48
0.46
0.50 0.51
0.49 0.50
0.500 0.505
1.0
0.5
0.0
0 1

Summary



$z \sim q(z)$





$x \sim p(x)$

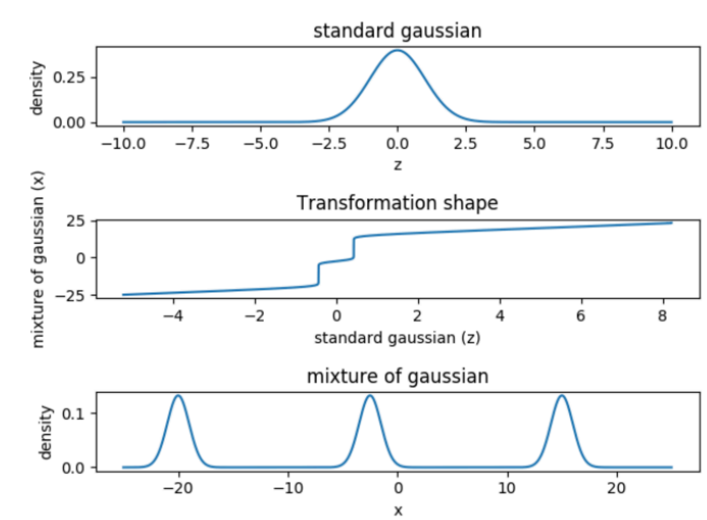
A probabilistic object

deterministic map

A probabilistic object

Complexity

Properties



[JKYB]. Tails of Lipschitz Triangular Flows. ICML, 2020

Spantini, Bigoni and Marzouk. Inference via low-dimensional couplings. JMLR, 2018

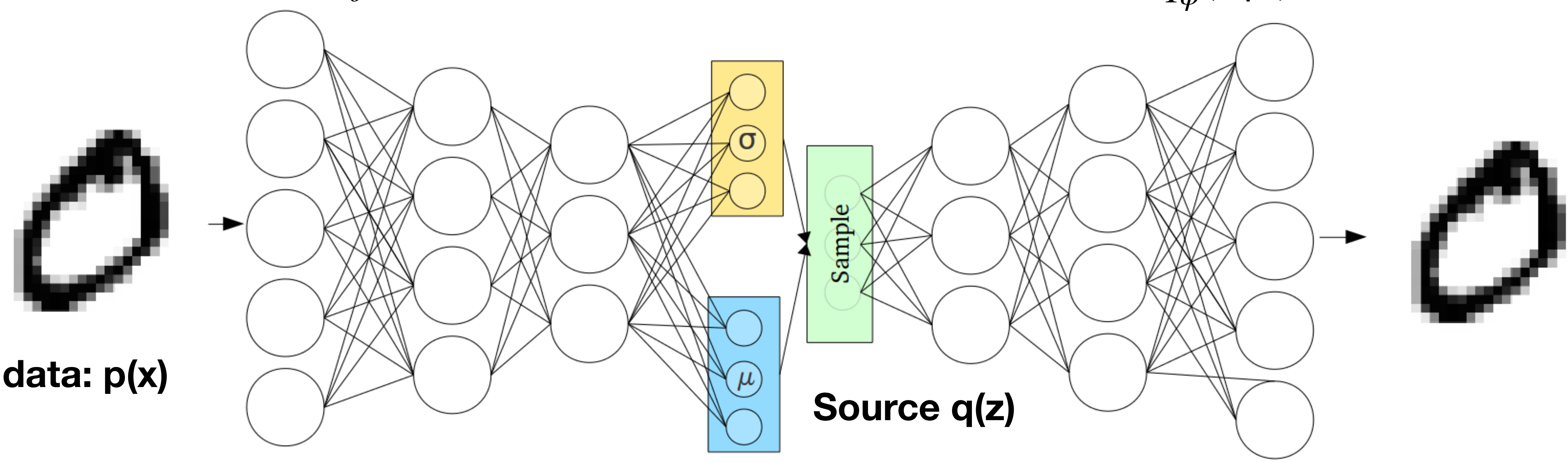
Variational Auto-Encoder

Given data $\mathbf{x}_1, \ldots, \mathbf{x}_n$, estimate p(x)

$$\min_{\theta} \min_{\phi} \ \mathrm{KL}\left[p(x) p_\theta(z \mid x) \middle\| q(z) q_\phi(x \mid z) \right]$$

Encoder: $p_\theta(z \mid x)$

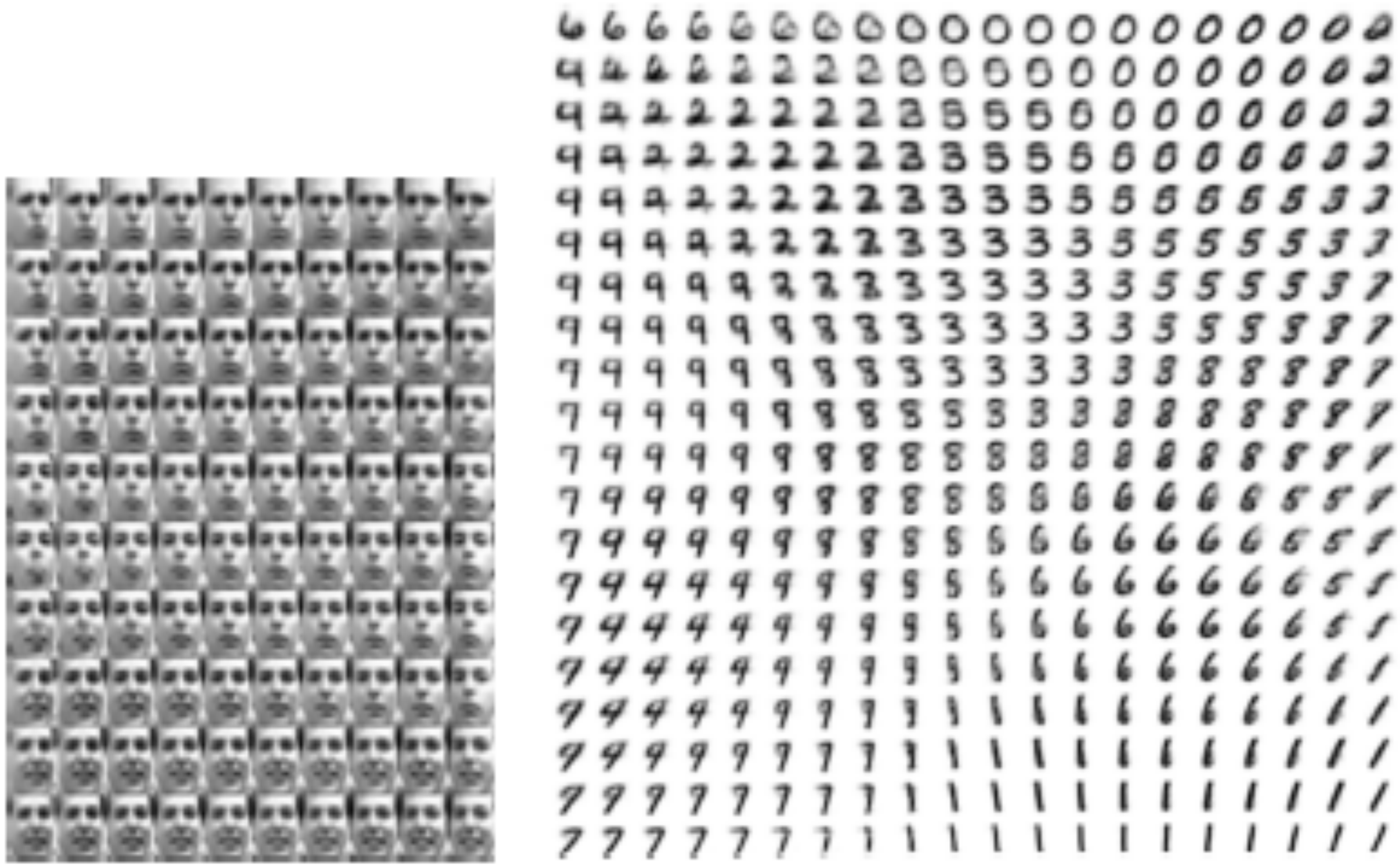
Decoder: $q_\phi(x \mid z)$



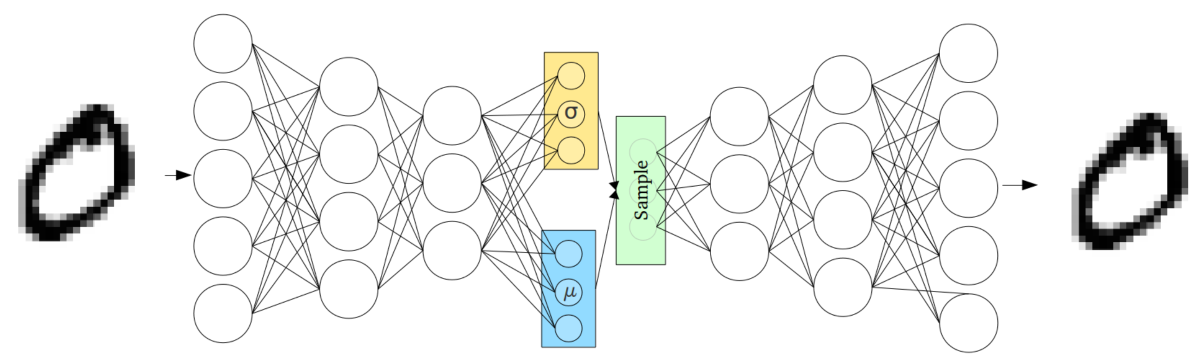
Kingma, D and Welling, M. Auto-Encoding Variational Bayes, ICLR, 2014

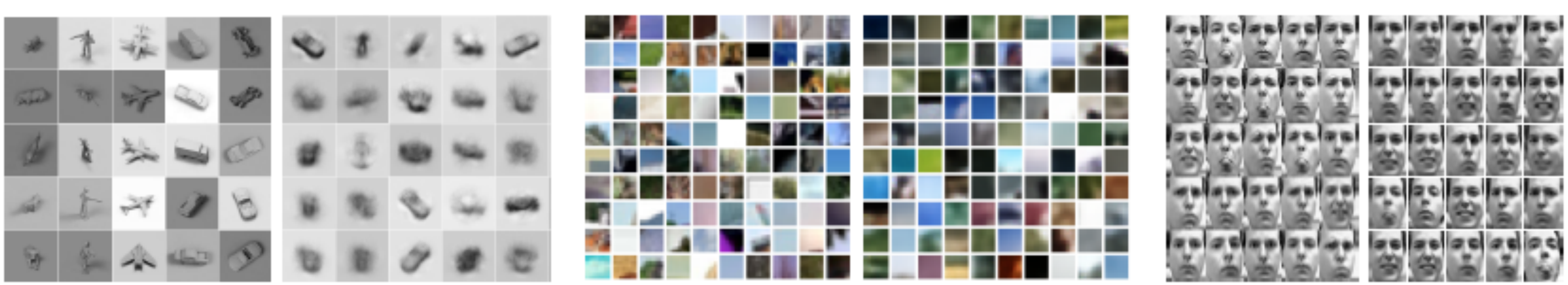
Rezende, D. et.al. Stochastic backpropagation and approximate inference in deep generative models, ICML, 2014

VAE examples



(a) Learned Frey Face manifold (b) Learned MNIST manifold





(a) NORB (b) CIFAR (c) Frey

Kingma, D and Welling, M. Auto-Encoding Variational Bayes, ICLR, 2014

Rezende, D. et.al. Stochastic backpropagation and approximate inference in deep generative models, ICML, 2014