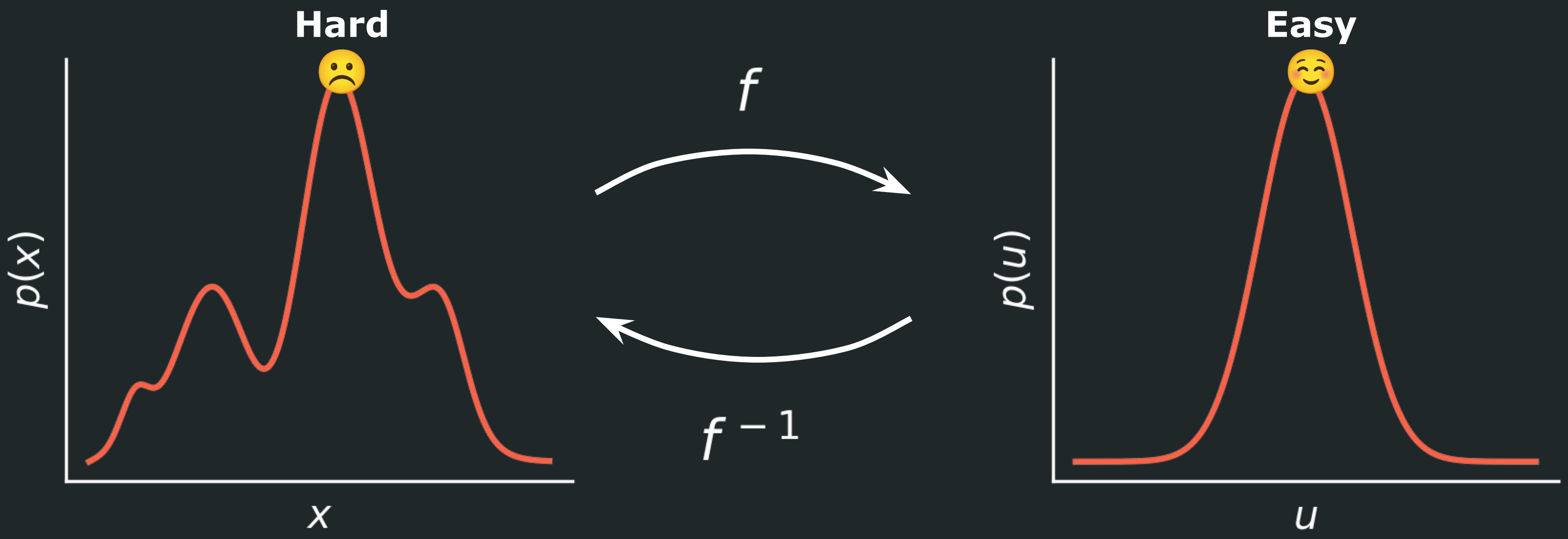
Normalizing Flows

John Franklin Crenshaw & Alex Gagliano

DESC MALTS

12-1-2022

Normalizing Flows



Normalizing Flows

Wie heißt du? $\xrightarrow{f}$ What is your name?

Normalizing Flows

f

Wie heißt du?

What is your name?

My name is John Franklin.

Normalizing Flows

f

Wie heißt du? → What is your name?

Ich heiße John Franklin. ← My name is John Franklin.

f^{-1}

Normalizing Flows

f

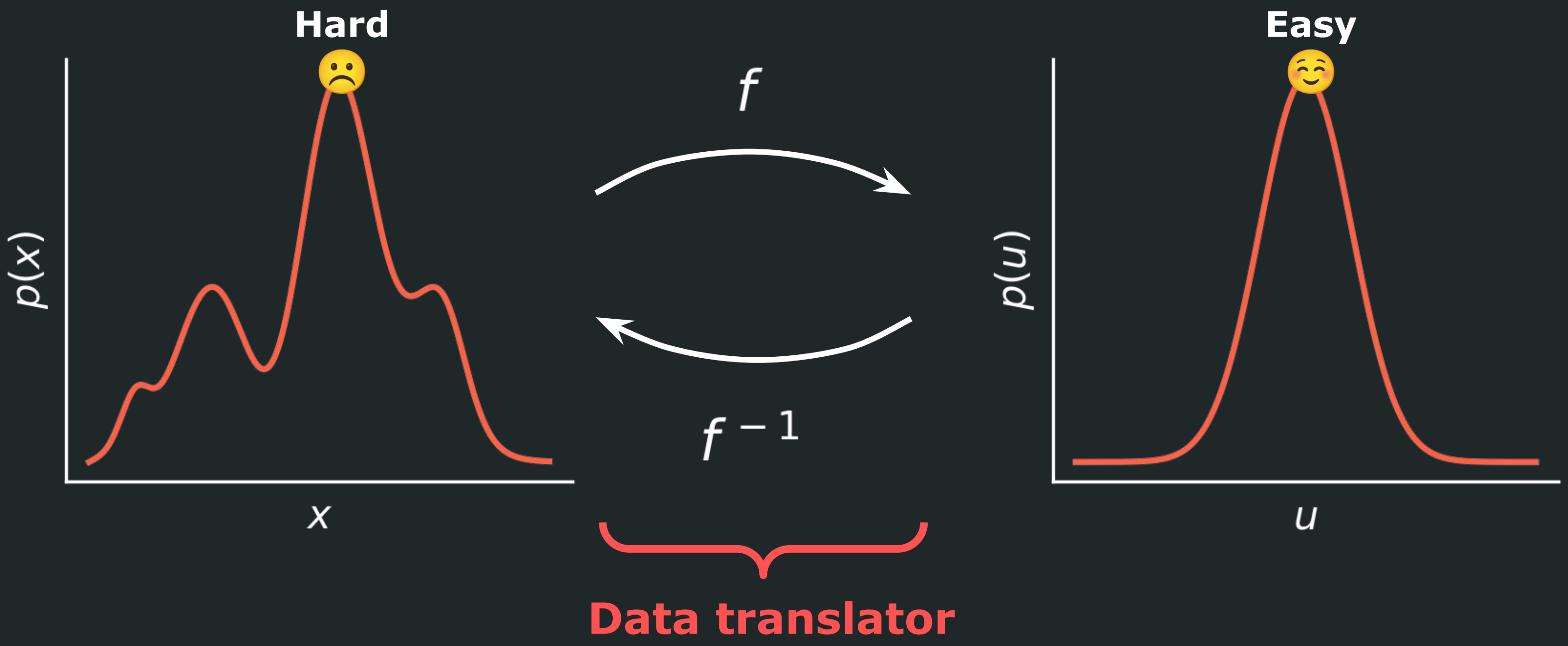
Wie heißt du? → What is your name?

Ich heiße John Franklin. ← My name is John Franklin.

f^{-1}

Language translator

Normalizing Flows



The Normalizing Flow likelihood

$$p_X(x) = p_U(u = f(x))\,|\det \nabla f(x)|,$$

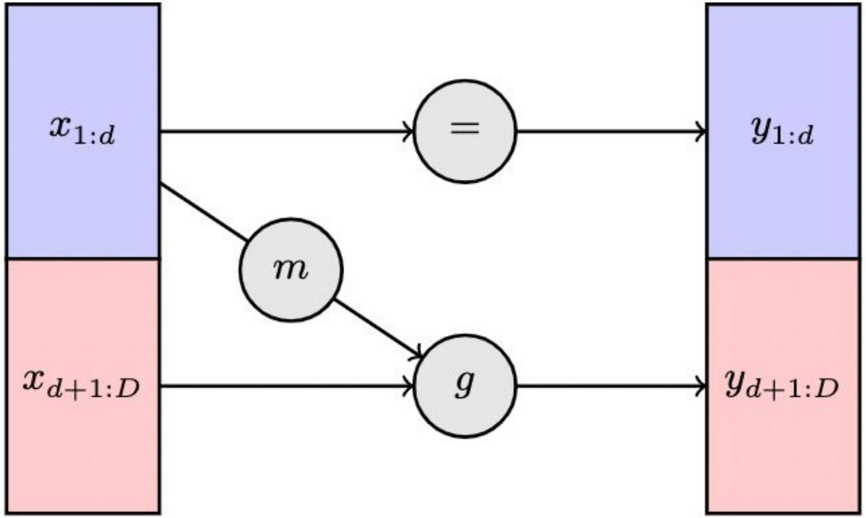
f maps data to latent space
Calculate likelihood in latent space

Multiply by the determinant of the Jacobian of the transformation

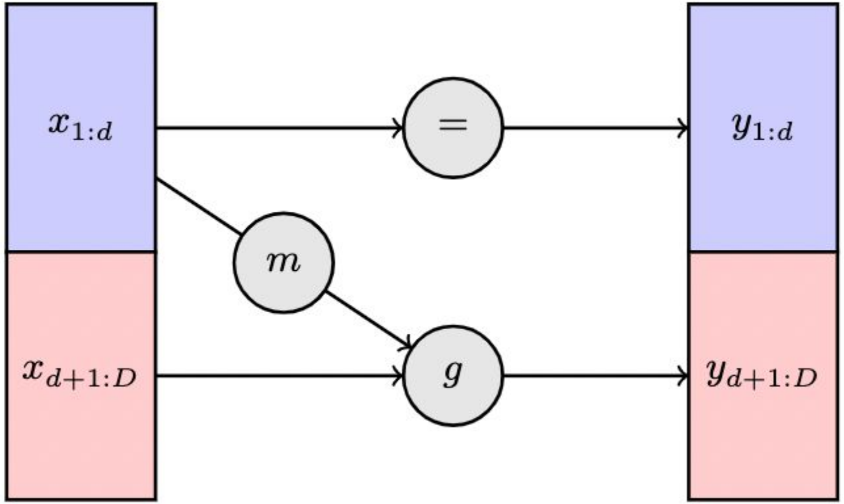
Coupling Layers

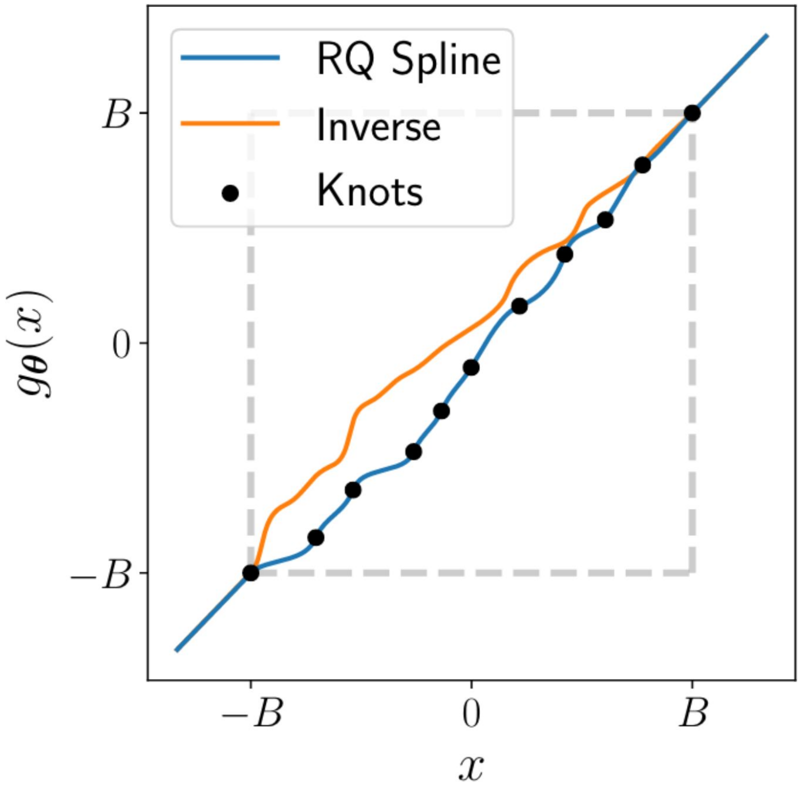
Transform a few of the dimensions as a function of the other dimensions

Stack a bunch of these back-to-back



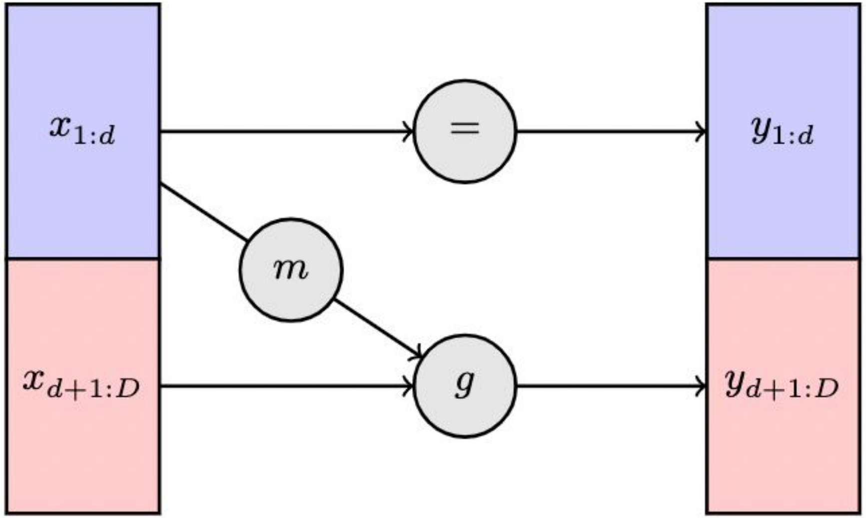
Coupling Layers





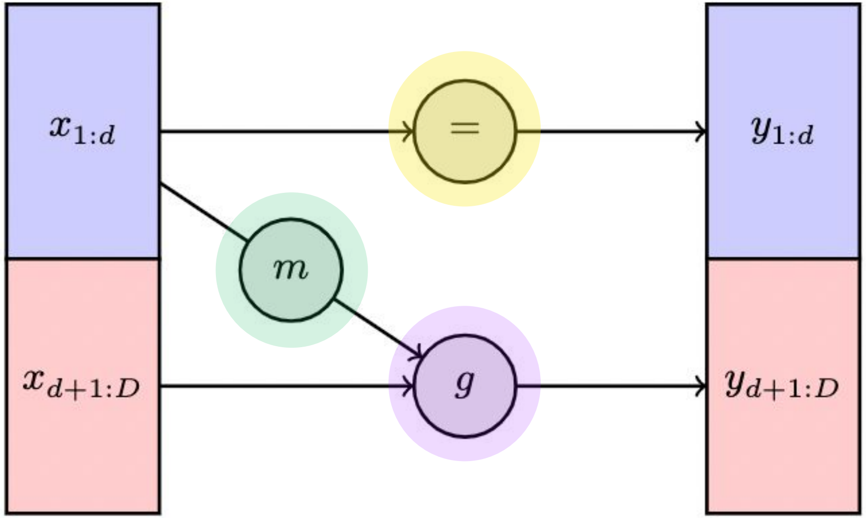
Coupling Layers

$$\frac{\partial y}{\partial x} = \begin{pmatrix} I_d & 0 \\ \frac{\partial y_{d+1:D}}{\partial x_{1:d}} & \frac{\partial y_{d+1:D}}{\partial x_{d+1:D}} \end{pmatrix}$$



Coupling Layers

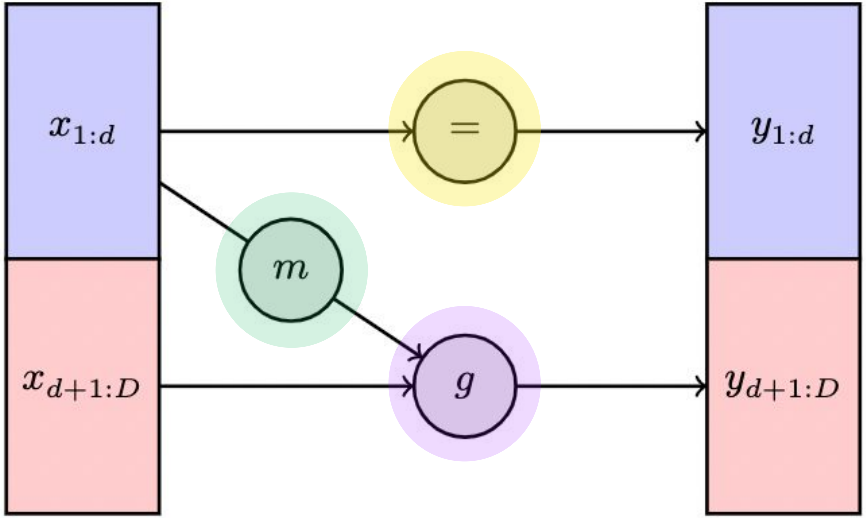
$$\frac{\partial y}{\partial x} = \begin{pmatrix} I_d & 0 \\ \frac{\partial y_{d+1:D}}{\partial x_{1:d}} & \frac{\partial y_{d+1:D}}{\partial x_{d+1:D}} \end{pmatrix}$$



Coupling Layers

$$\frac{\partial y}{\partial x} = \begin{pmatrix} I_d & 0 \\ \frac{\partial y_{d+1:D}}{\partial x_{1:d}} & \frac{\partial y_{d+1:D}}{\partial x_{d+1:D}} \end{pmatrix}$$

$$\det \frac{\partial y}{\partial x} = \det \frac{\partial y_{d+1:D}}{\partial x_{d+1:D}}.$$

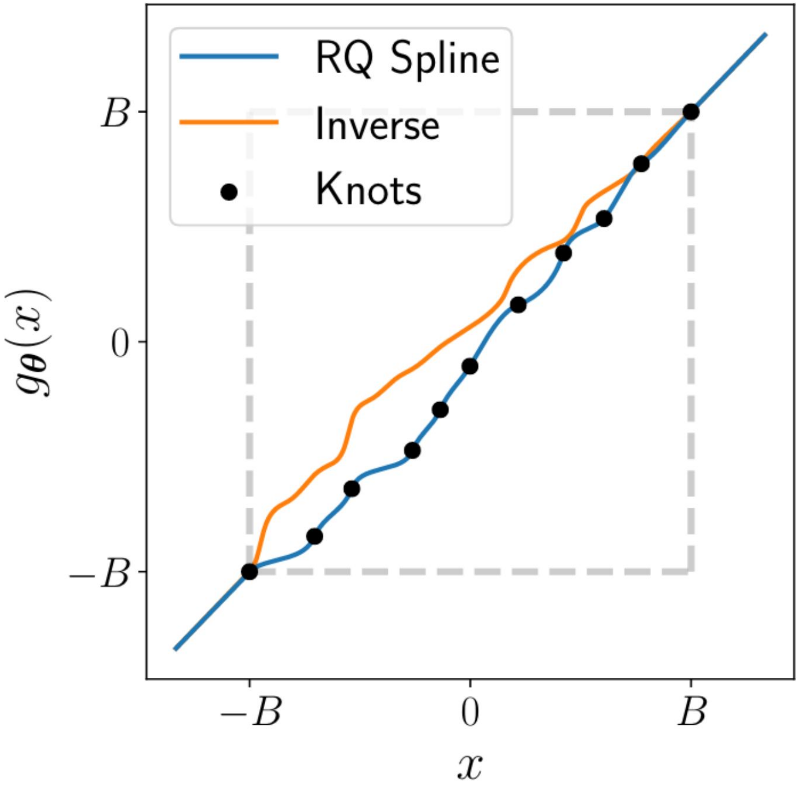


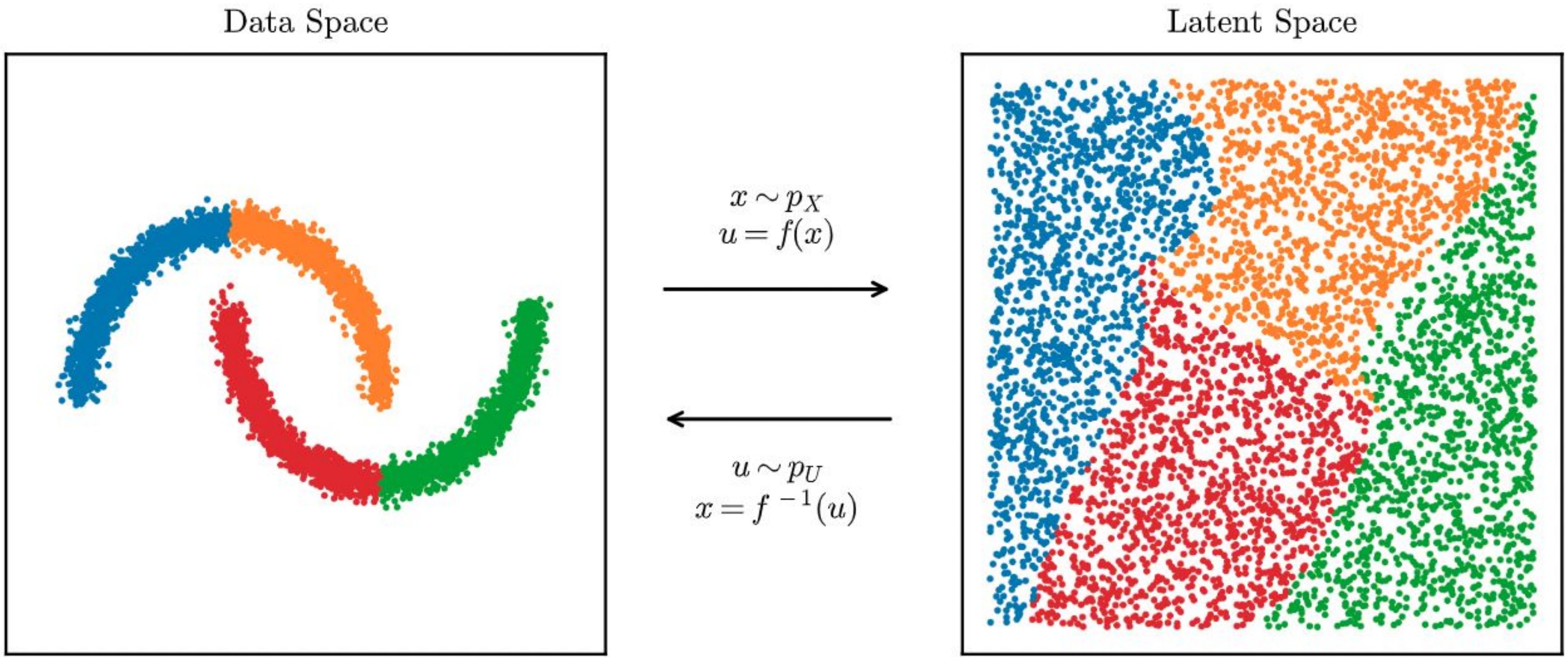
Rational Quadratic Splines

Very flexible, and invertible

Efficient in both directions!

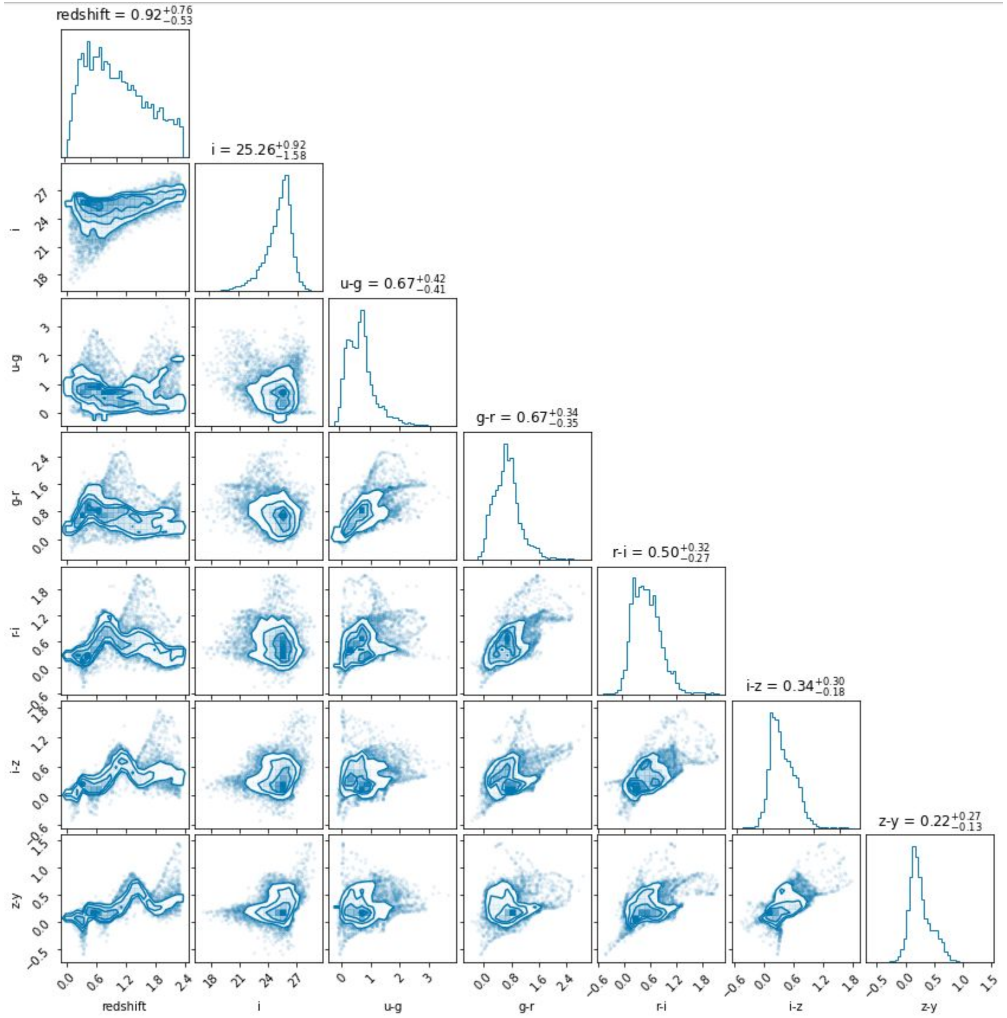
Sampling *and* likelihood evaluation!





Data Space
Latent Space
$x \sim p_X$
$u = f(x)$
$u \sim p_U$
$x = f^{-1}(u)$

Example: modeling a galaxy catalog



More PZFlow tutorials:
https://jfcrenshaw.github.io/pzflow/tutorials/

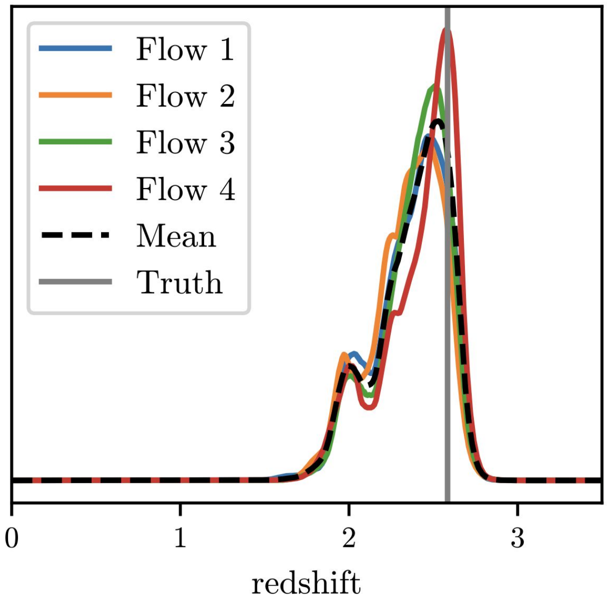
An Ensemble of Flows

```
from pzflow import Flow
flow = Flow(...)
```

→

```
from pzflow import FlowEnsemble
flowEns = FlowEnsemble(..., N=4)
```

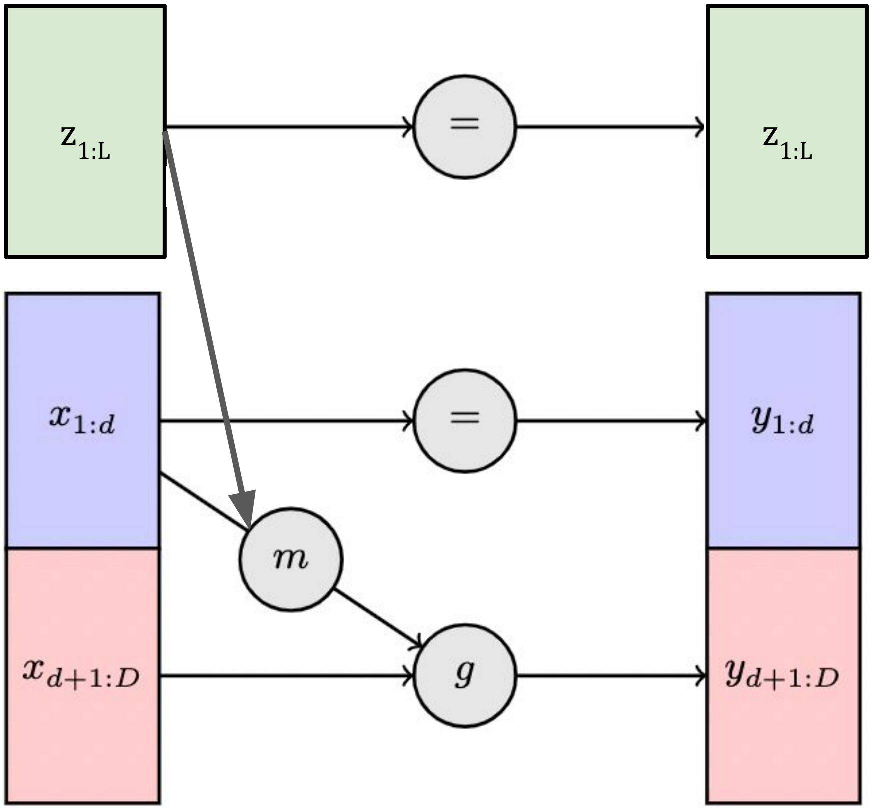
- Approximate Bayesian marginalization over neural network parameters
- Marginalize over multiple basins of attraction (most methods focus on marginalizing over a single basin)
- Can return averages or individual solutions



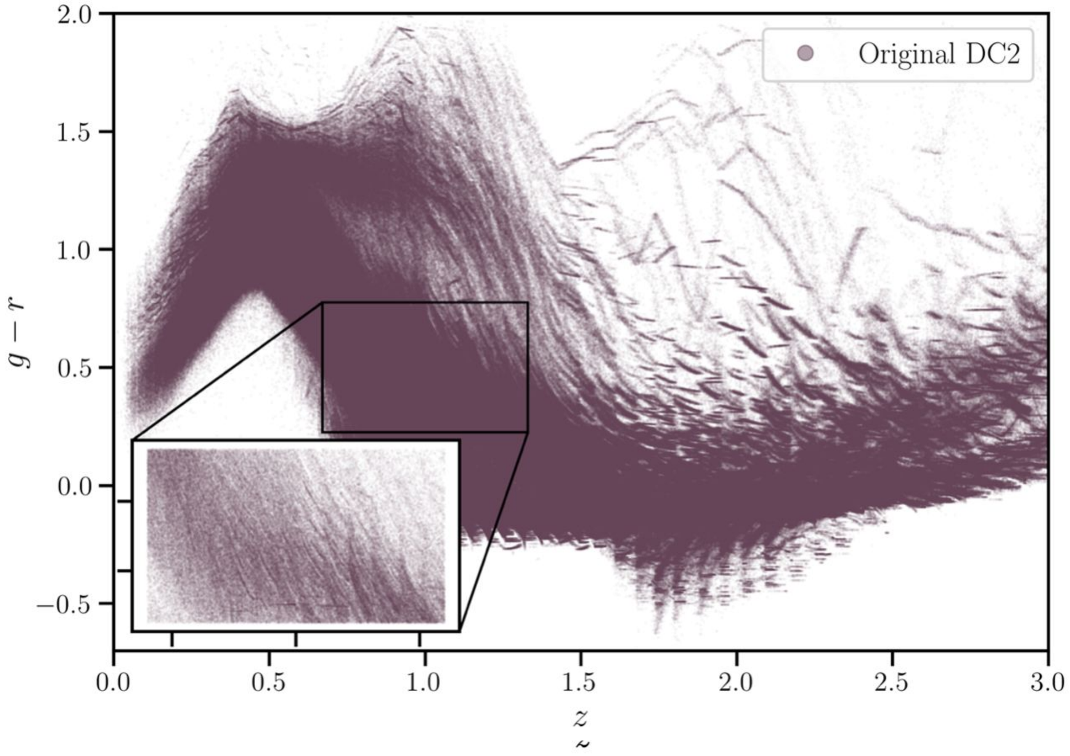
Conditional Flows

You can pass any other variables you want into the neural network

This allows the network to condition outputs on these values

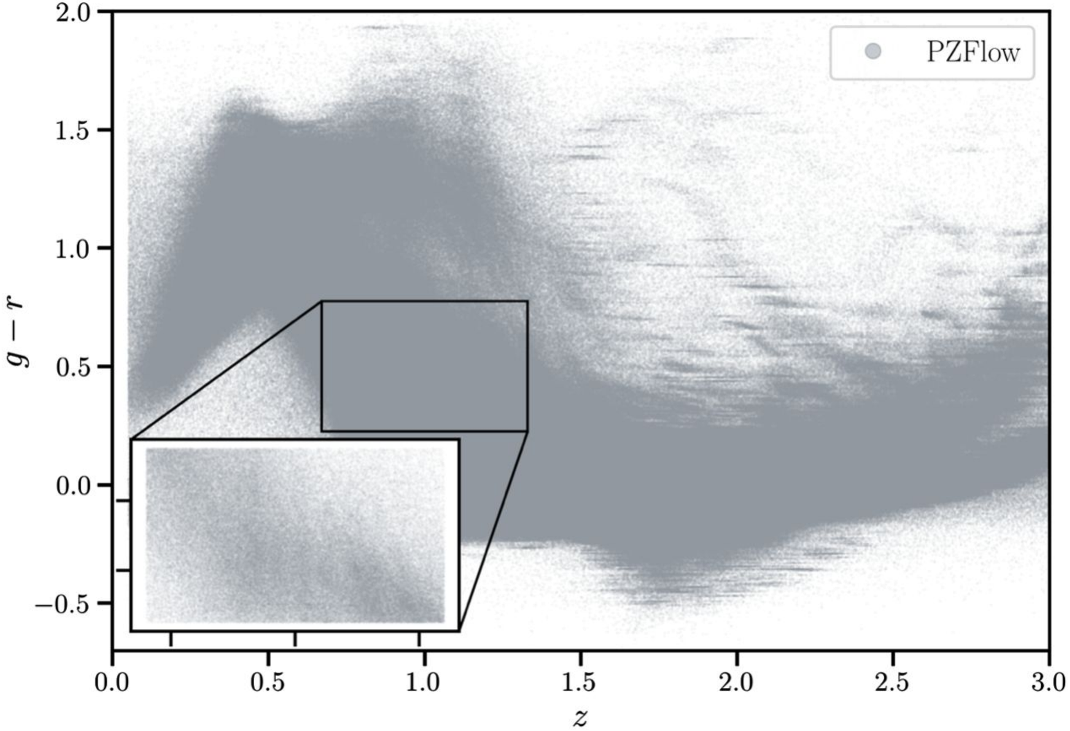


Overcoming Limitations of CosmoDC2



The Challenge: Learn a more physical version of this complex distribution: $p(z \,|\, u, g, r, i, z, y, M_*, \mathrm{SFR})$

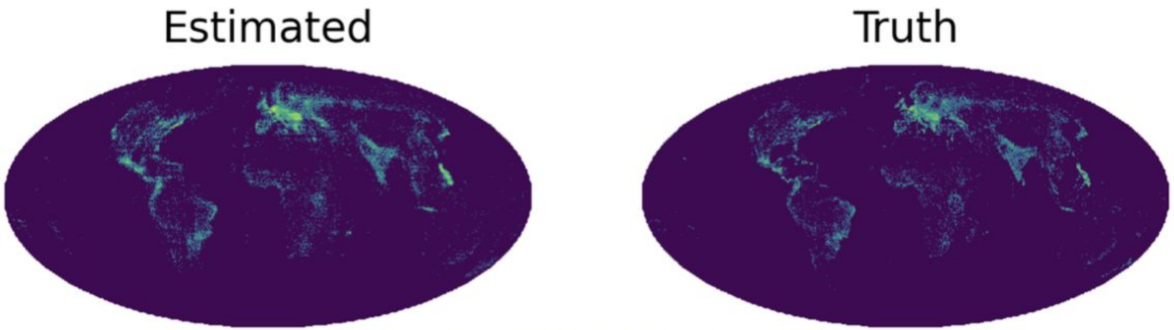
Overcoming Limitations of CosmoDC2



The Solution: Create a mapping between the complex distribution and a simpler one. Train to draw out broad physical correlations, and re-sample for new redshifts.

A Note on Compact Support/Periodic Topologies

https://github.com/LSSTDESC/transient-host-sims/notebooks/SCOTCH_walkthroughs.ipynb



Can be used to Simulate Observations Within a Survey Footprint!