UNIVERSITÀ DEGLI STUDI DI GENOVA

Istituto Nazionale di Fisica Nucleare
Sezione di Genova

Normalizing Flows for higher dimensional data sets.

(work in progress)

Humberto Reyes-González
University of Genoa

ACAT 2021
Daejeon, Korea - Virtual 30/11/2021

Motivation.

- In HEP we find complex Probability Distribution Functions (PDFs) **EVERYWHERE!**
- What do we want to do with them? -> (Re)-interpret, preserve, sample, combine, invert, …
- Can Normalizing Flows (NFs) help us on these endeavours?…

- **Normalizing flows** are a powerful brand of *generative models.*
- They map simple to complex distributions.
- They **allow for efficient sampling** of complex PDFs…
- … and include **density estimation** by construction!

(Some) Applications on HEP already on the market:

- Numerical integration (arXiv:2001.05486, arXiv:2001.05478)
- Unfolding (arXiv:2006.06685)
- Calorimeter shower simulation (arXiv:2106.05285)
- Event generation (arXiv:2001.10028, arXiv:2110.13632)
- …

IN THIS TALK:

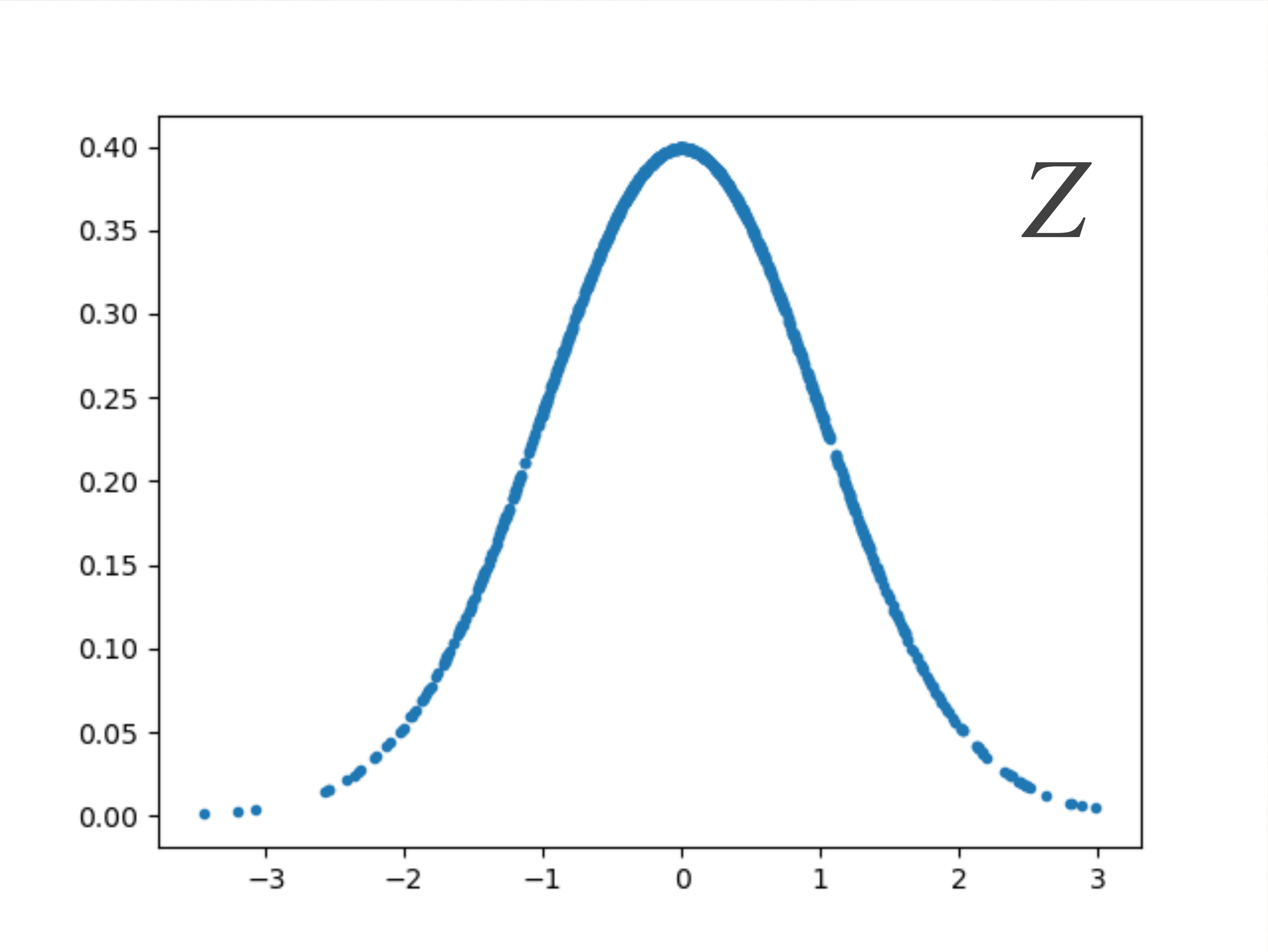
We want to find out…

- How far can we go dimension-wise?
- Can NFs learn the *high-dimensional Likelihood functions of LHC results*?

Introduction.

BASIC PRINCIPLE:

Following the change of variables formula, perform a series of **bijective, continuous, invertible** transformations on a *simple* probability density function (pdf) to obtain a *complex* one.

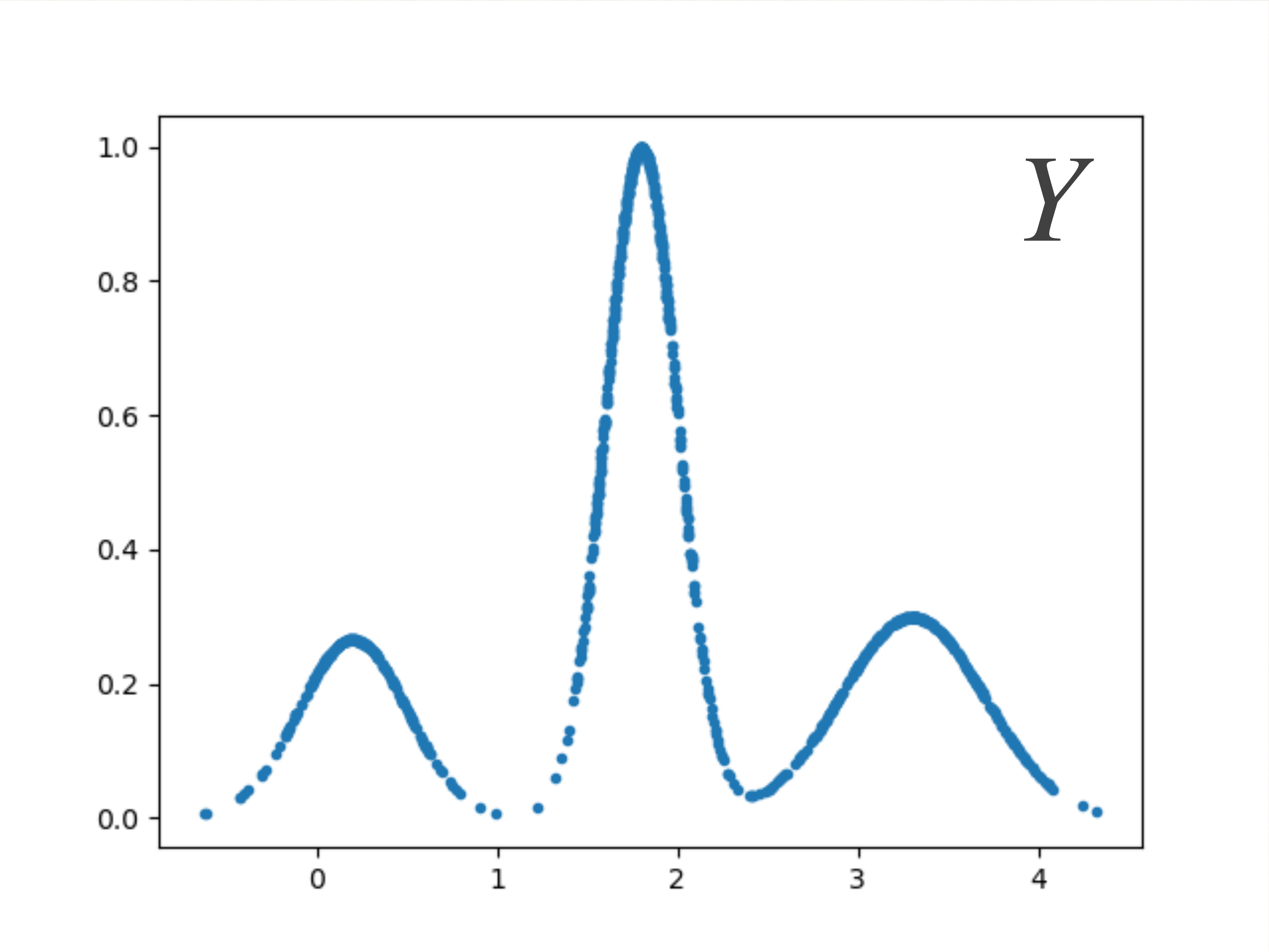


Normalizing direction

$Z = f(Y)$

Generative direction

$Y = g(Z)$



See review
Ivan K. et. al.
arXiv:1908.09257

Choosing the transformations

THE OBJECTIVE:

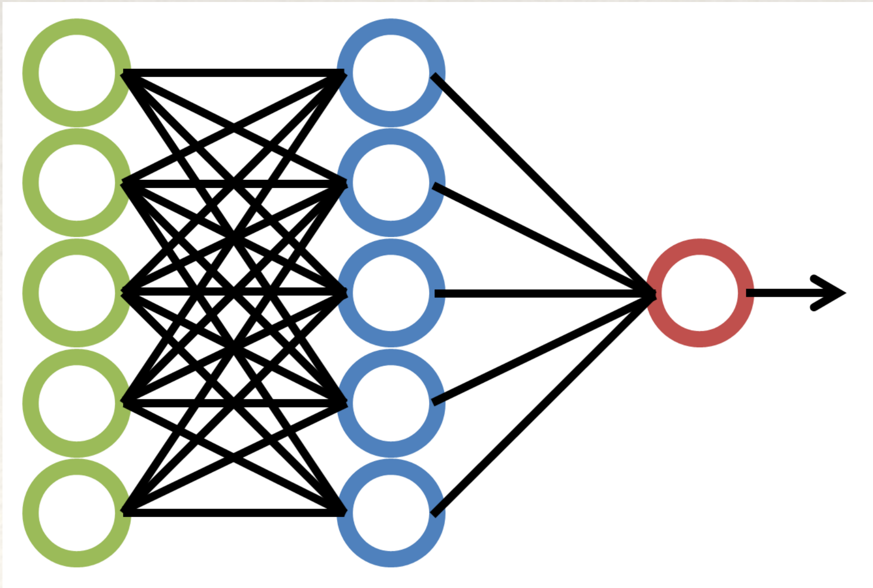
To perform the right transformations to accurately estimate the complex underlying distribution of some observed data.

THE RULES OF THE GAME:

- The transformations must be invertible
- They should be sufficiently expressive
- And computationally efficient (including Jacobian)

THE STRATEGY

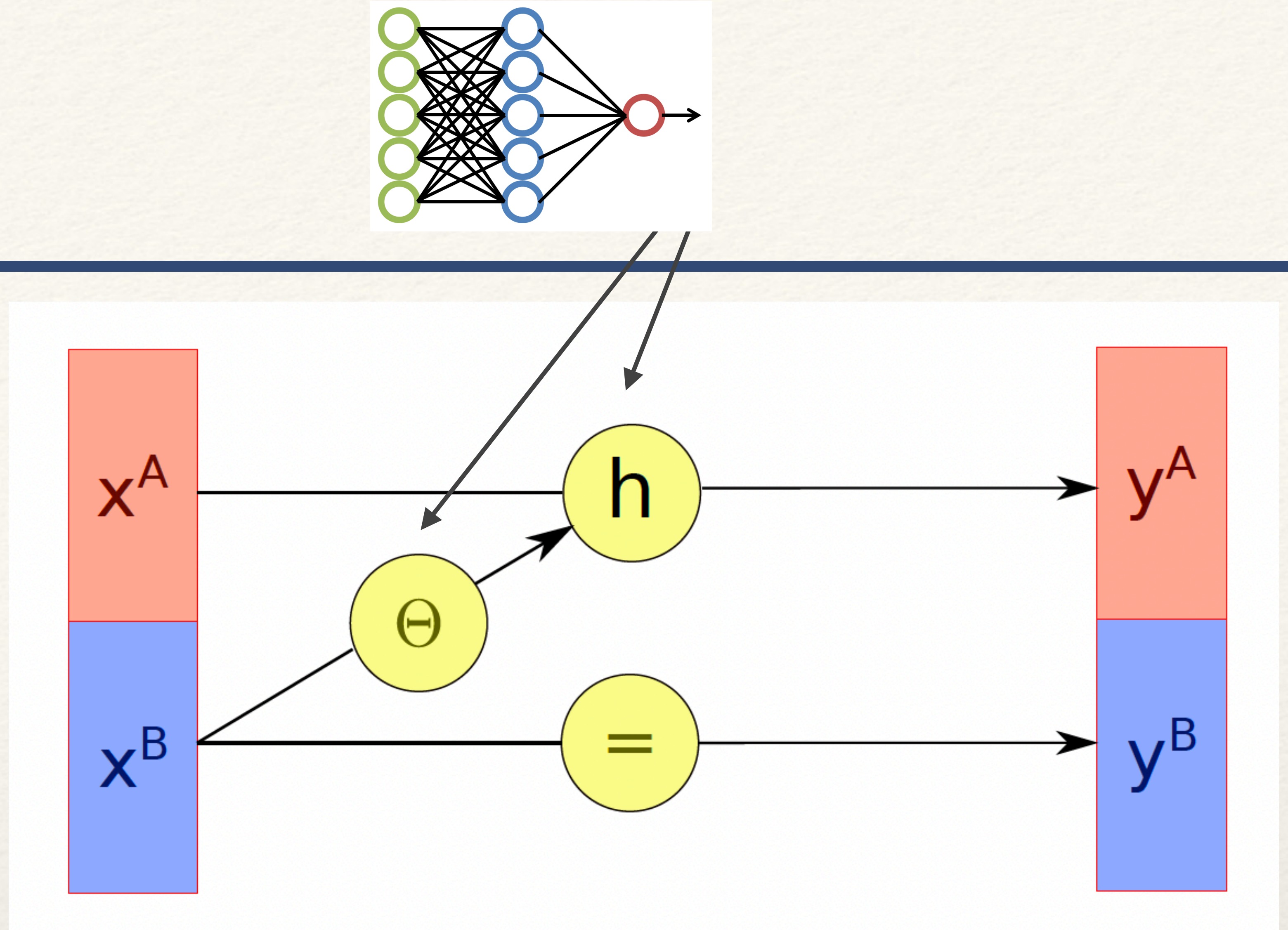
Let ***Neural Networks*** learn the parameters of ***Autoregressive Normalizing Flows.***



Autoregressive Flows

Coupling Flows :

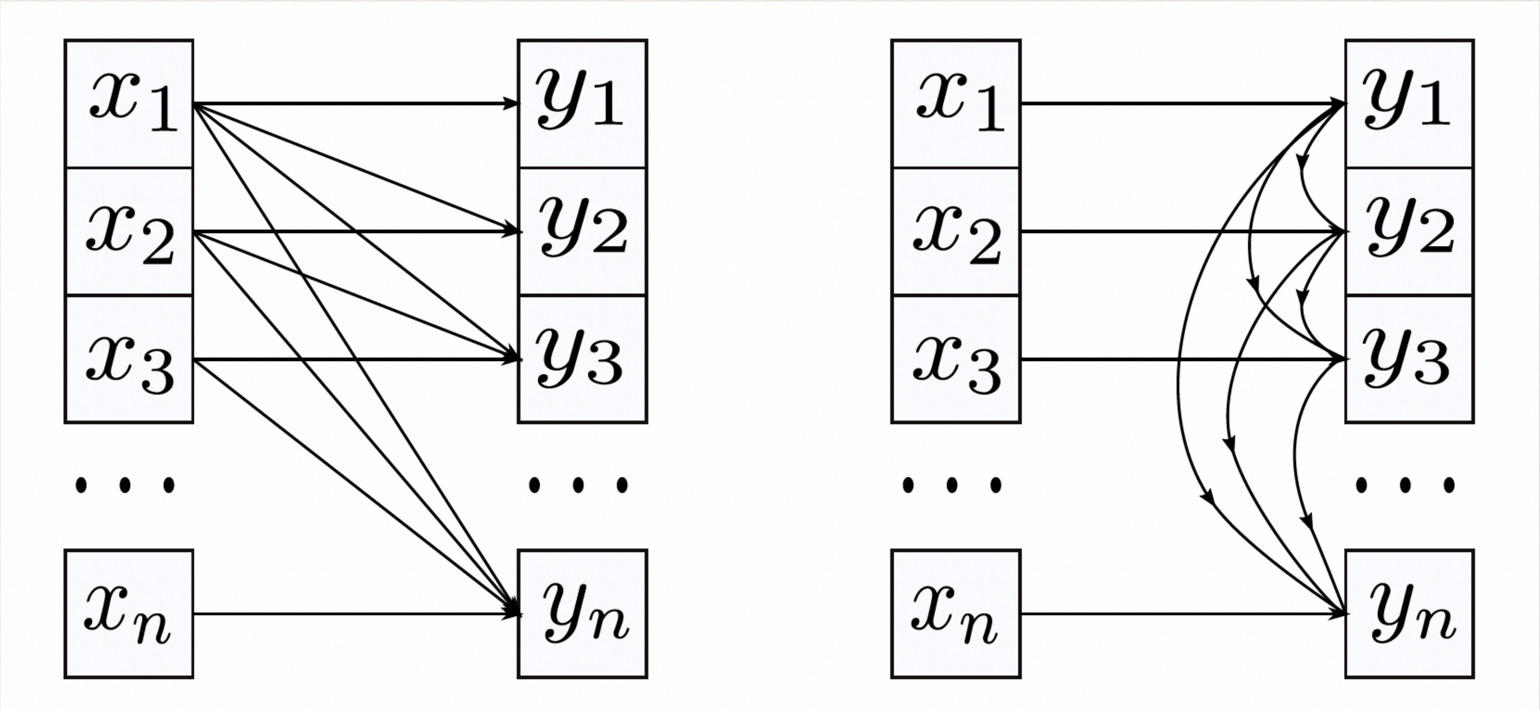
- Dimensions are divided in two sets: x^A and x^B
- We transform x^B with bijectors trained with x^A.
- The bijector parameters are functions of a NN.
- The Jacobian J is triangular -> $\det J = \Pi_i J_{ii}$
- **Jacobian is easily computed!**
- **Direct sampling AND density estimation.**
- **Less expressive.**



arXiv:1908.09257

Autoregressive Flows :

- Dimension x^i is transformed with bijectors trained with $y_{1:i-1}$
- Bijector parameters are trained with Autoregressive NNs.
- The Jacobian J is also triangular thus…
- **Jacobian is easily computed!**
- **Direct sampling OR density estimation.**
- **More expressive.**



arXiv:1908.09257

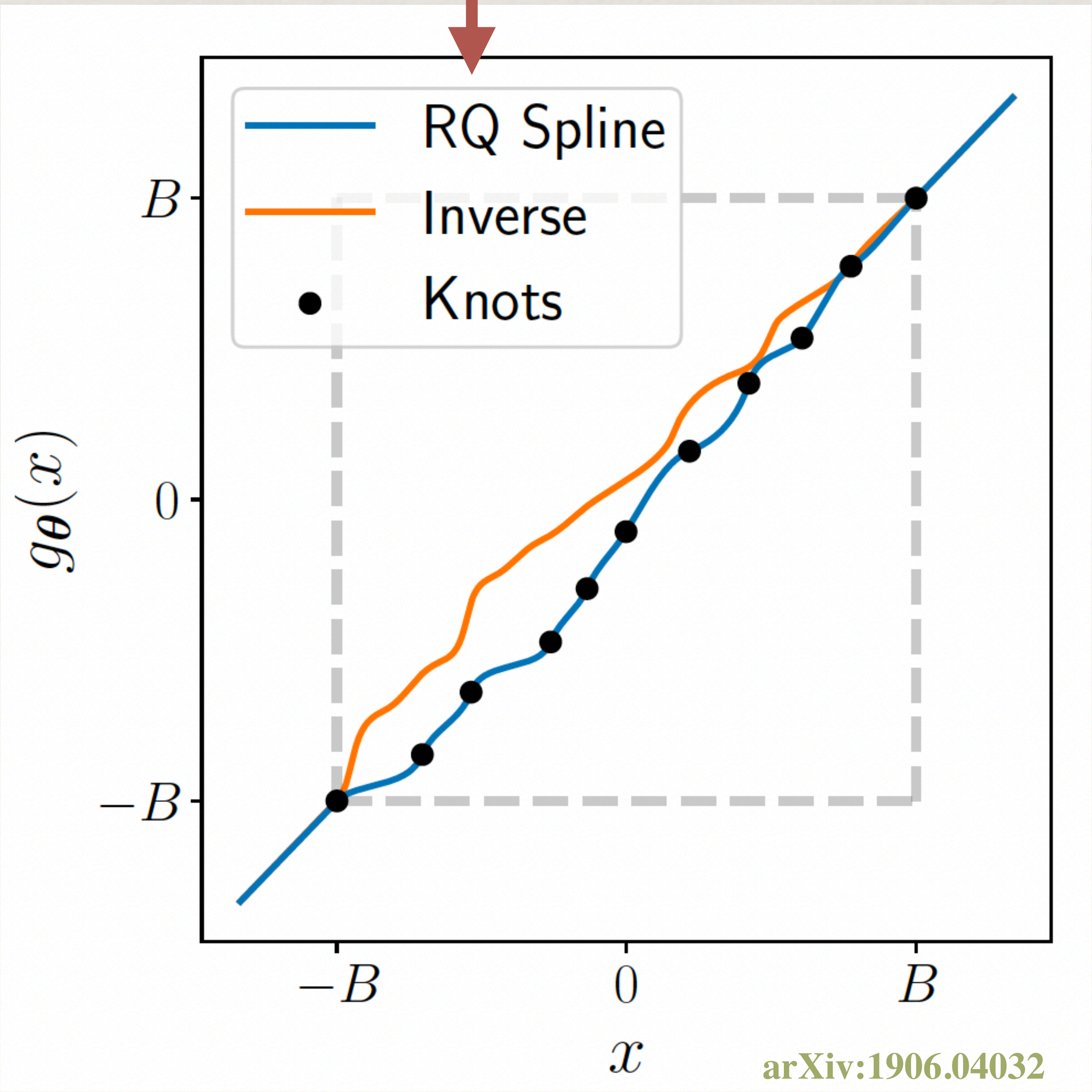
The loss function:

$-\log(p_{AF}(target_{dist}))$

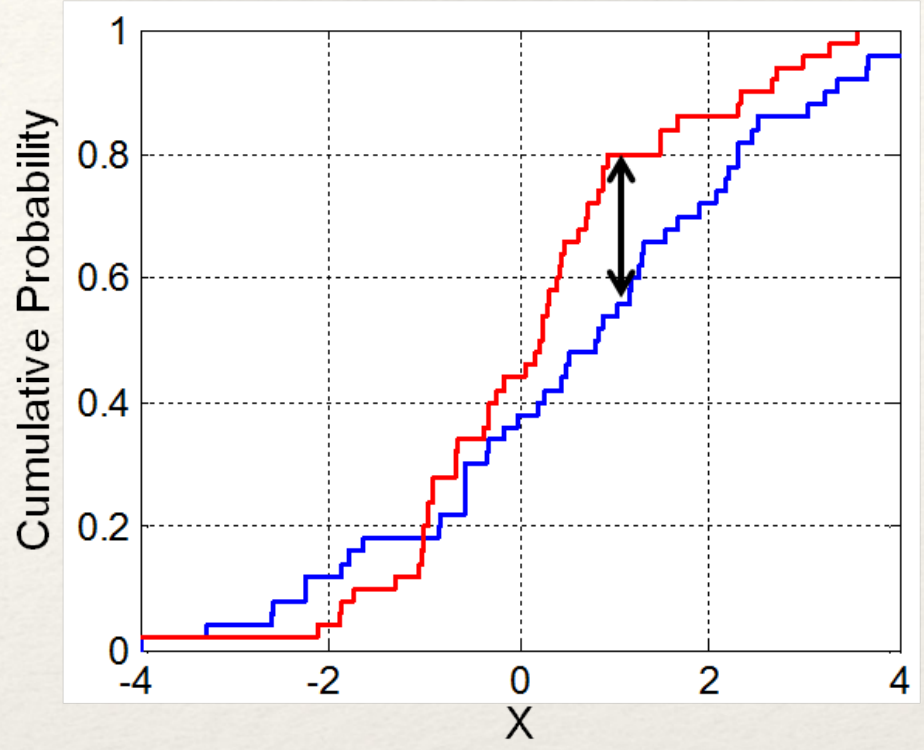
Our Autoregressive Flows

RealNVP	MAF	A-NSF
Real-Valued Non-Volume Preserving (arXiv:1605.08803)	Masked Autoregressive Flow (arXiv:1705.07057)	(Autoregressive) Neural Spline Flows (arXiv:1906.04032)
Coupling Flow	Autoregressive Flow	Autoregressive Flow
Affine	Affine	Rational Quadratic Spline

$$y(x; \mu, b) = \mu \cdot x + b$$



(PDF agnostic) metrics.



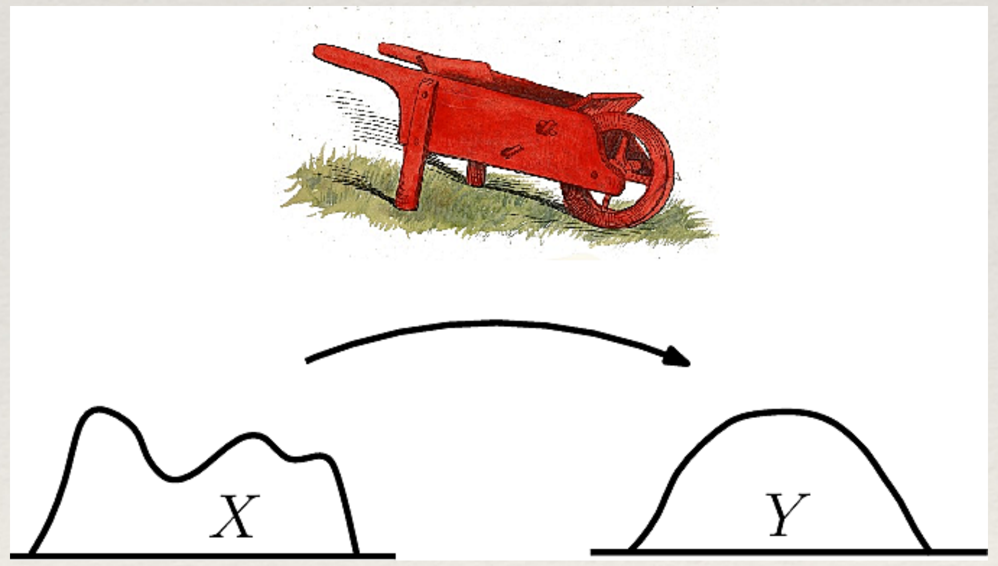
https://en.wikipedia.org/wiki/Kolmogorov-Smirnov_test

- Two-sample 1D Kolgomonov - Smirnov test (ks test):

$$D_{n,m} = \sup_x |F_n(x) - F_m(x)|$$

-Computes the p-value for two sets of 1D samples coming from the same *unknown* distribution.
-We average over ks test estimations and compute the median over dimensions.
-Optimal value 0.5

- 1D Wasserstein distance (Earth mover's distance)



https://sbl.inria.fr/doc/Earth_mover_distance-user-manual.html

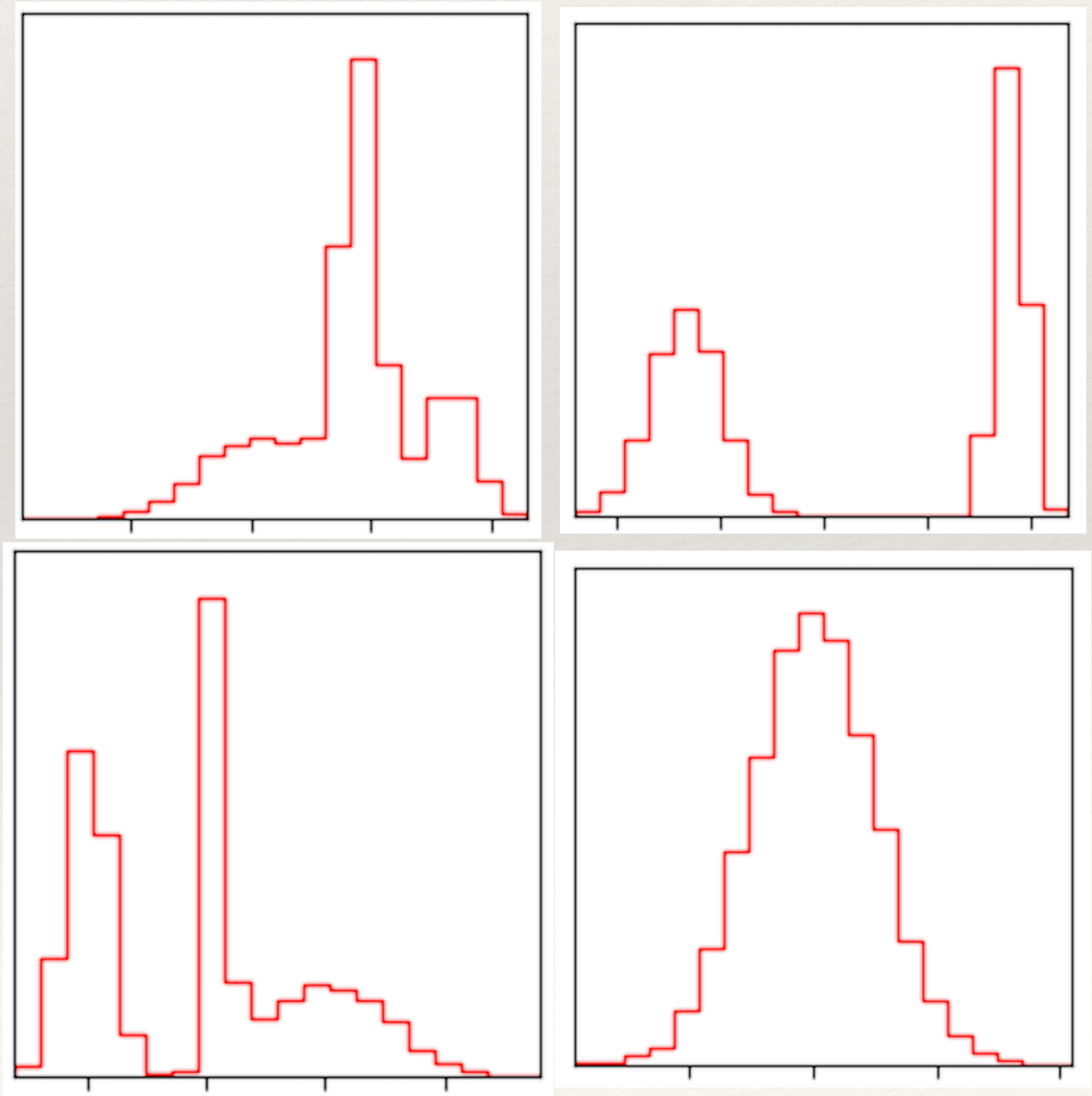
$$l(f_n, f_m) = \int_{-\infty}^{\infty} |F_n - F_m|$$

-Computes the minimum *energy* required to transform f_n into f_m
-We compute the median over dimensions.
-Optimal value 0.0

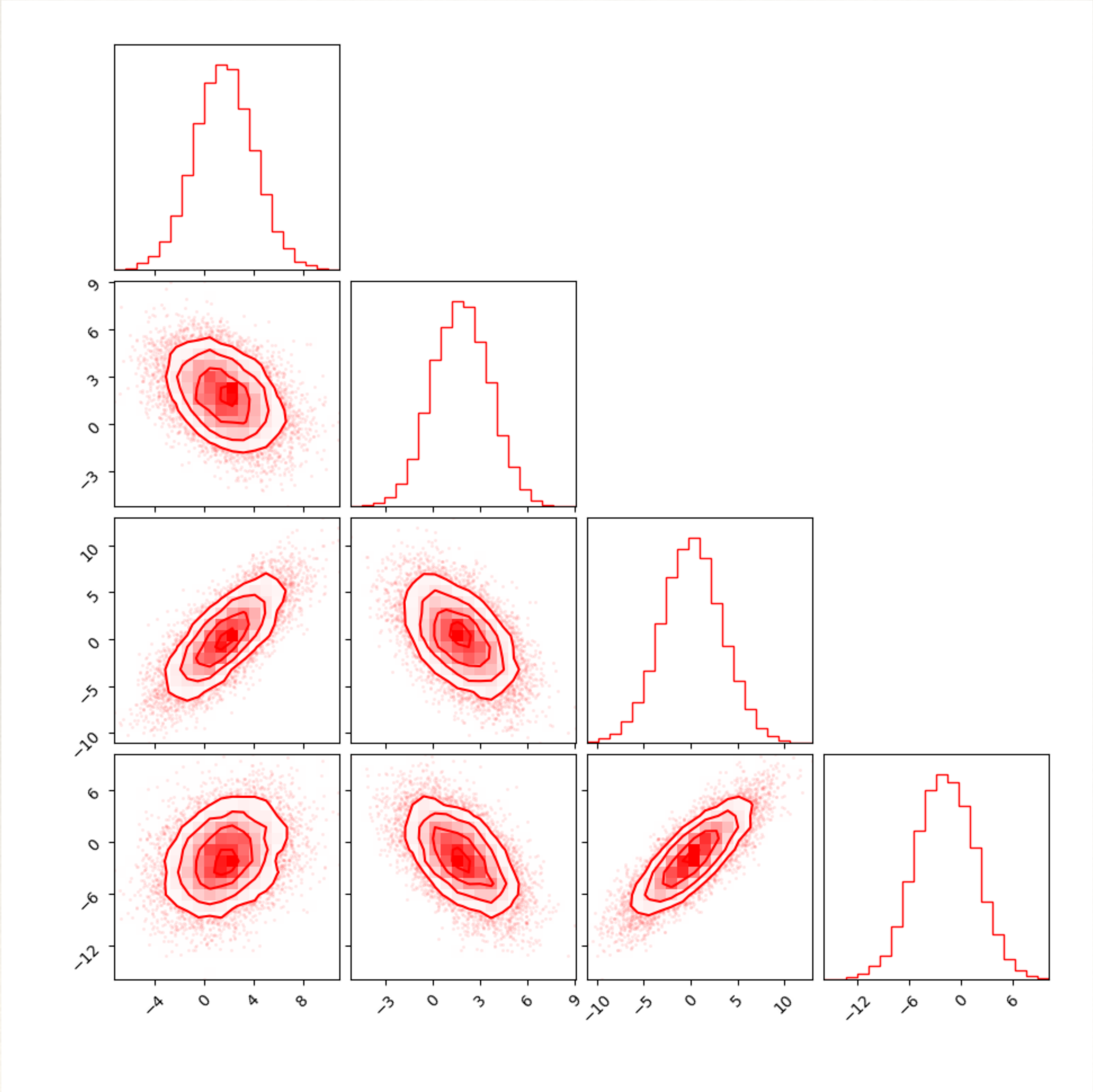
Testing the Flows.

Toy distributions from 4 to 100 dims:

Uncorrelated Mixture of Gaussians

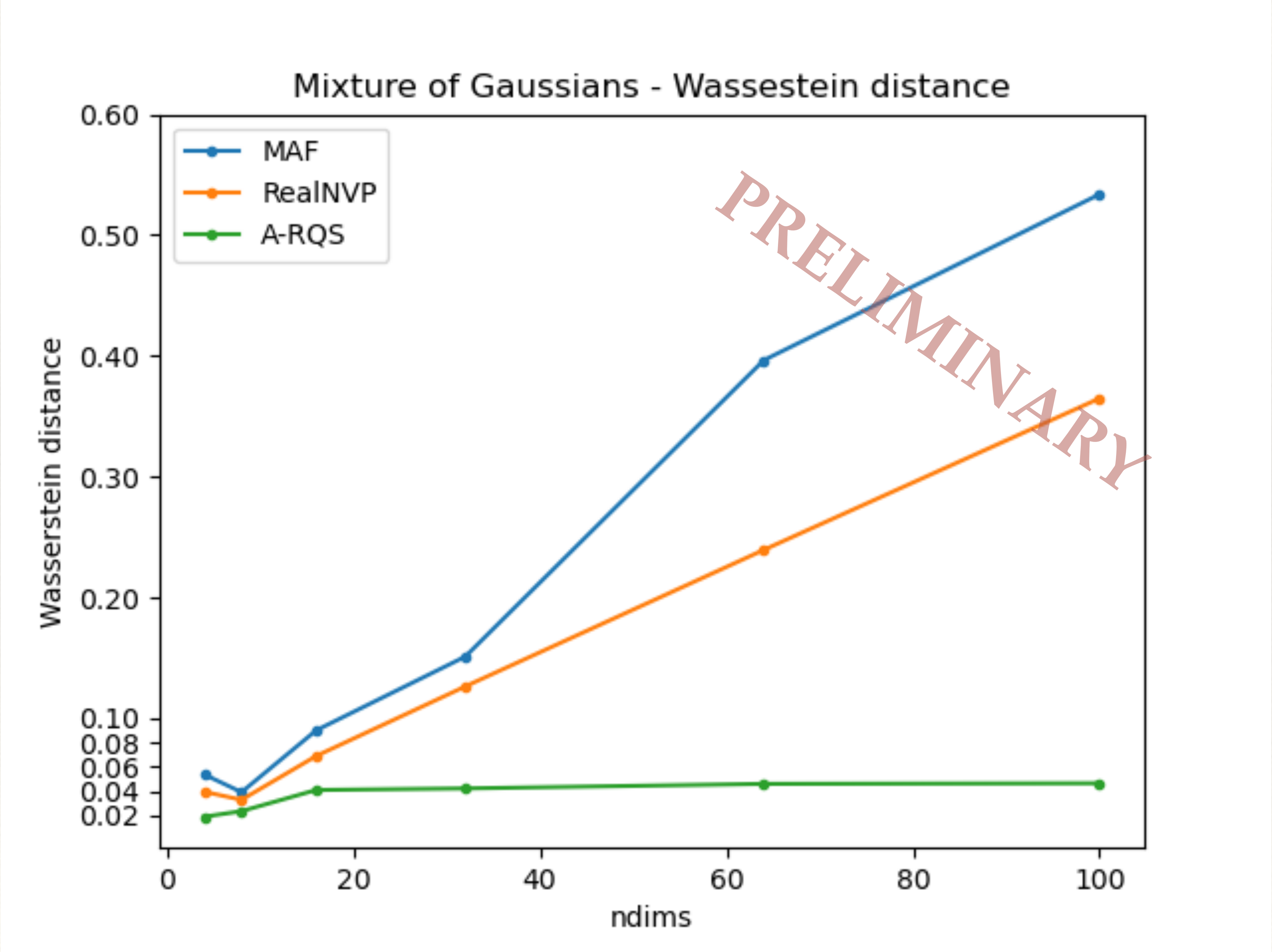


Correlated Gaussians

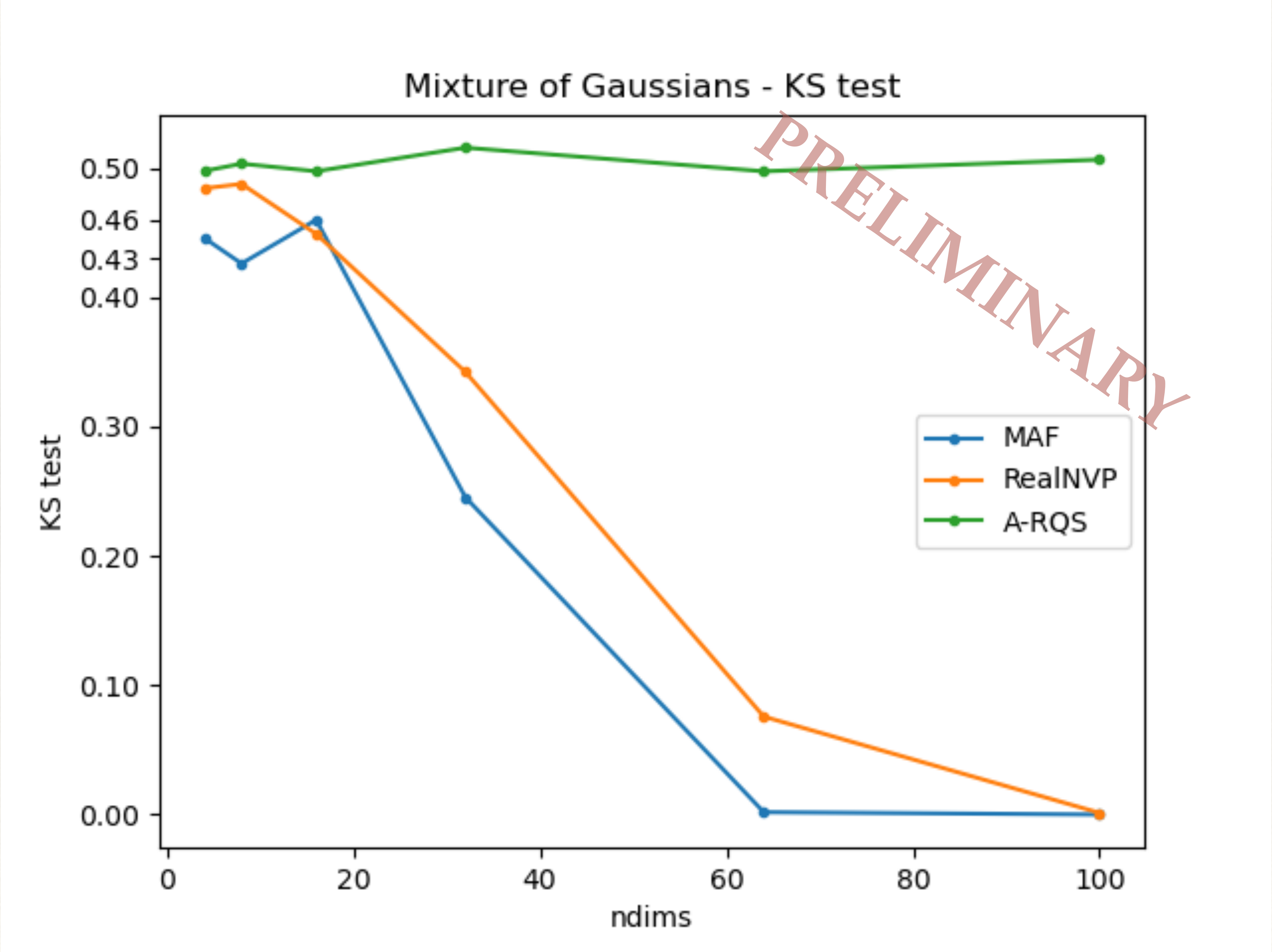


Uncorrelated Mixture of Gaussians

	N bijectors	Hidden layers	N samples
MAF	3, 5, 10	128x3, 256x3	100k, 300k
RealNVP	10	128x3, 256x3	100k, 300k
A-NSF (8knots)	2	128x3, 256x3	100k



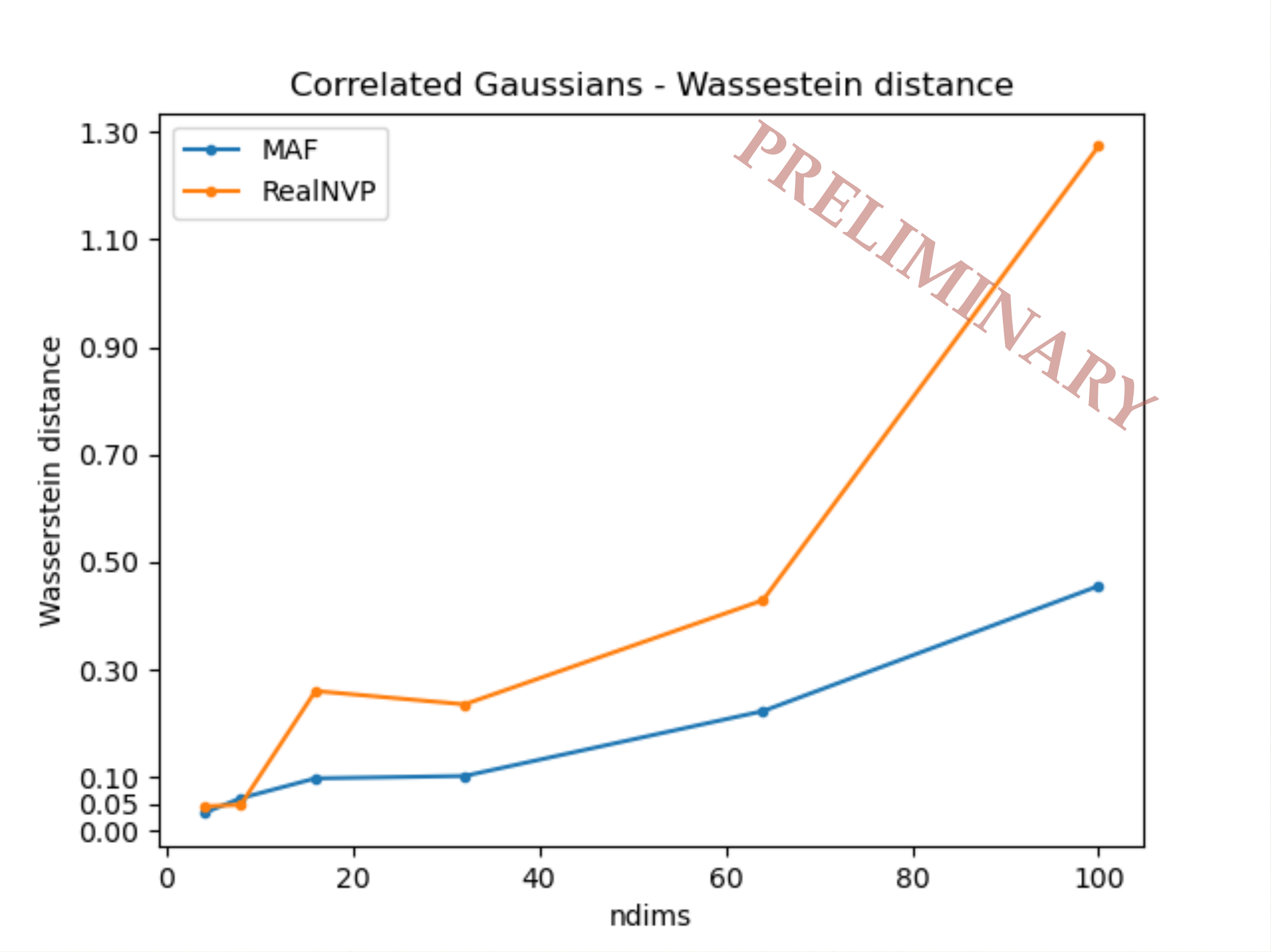
Wasserstein distance



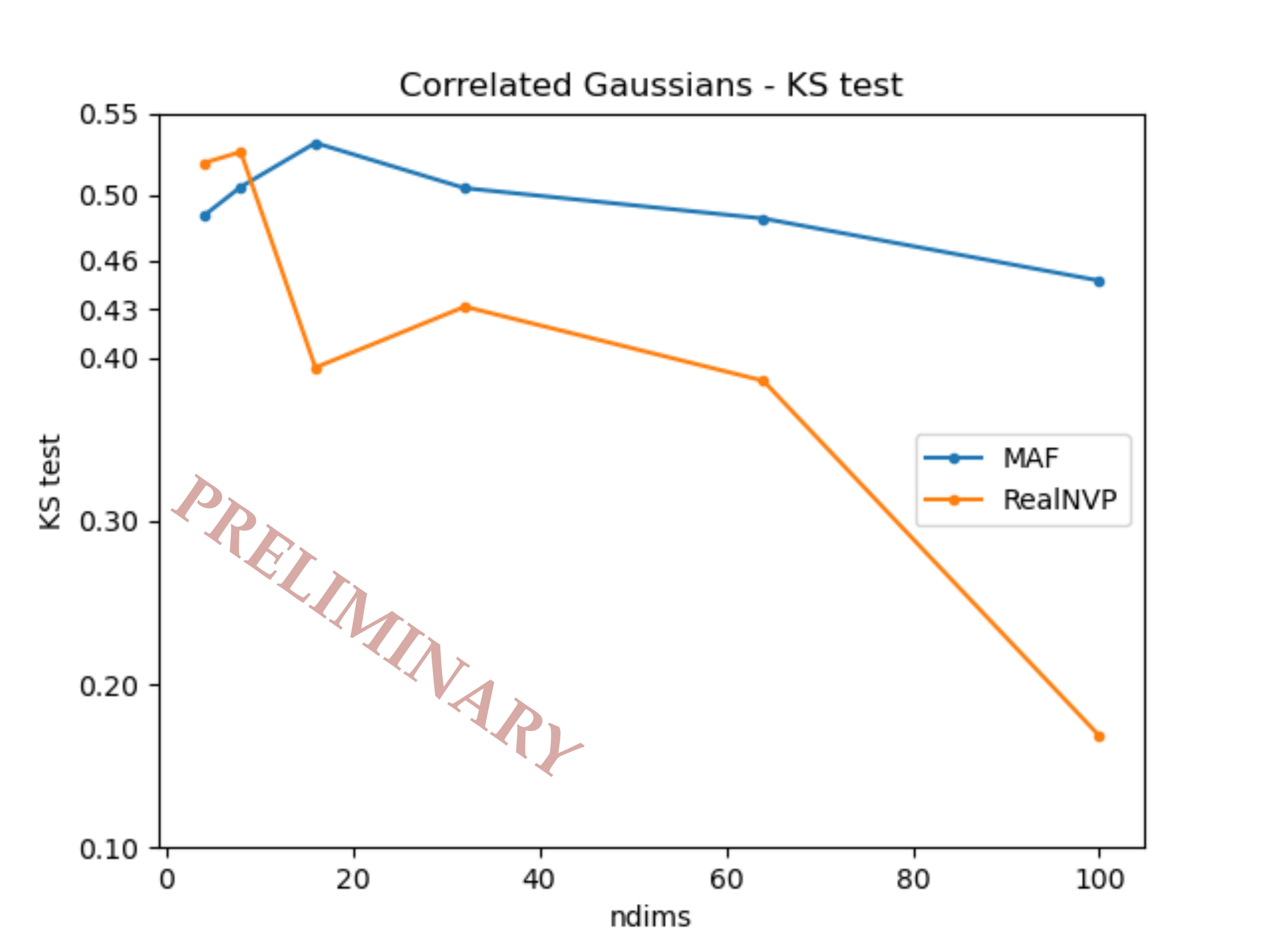
KS test

Correlated Gaussians

	N bijectors	Hidden layers	N samples
MAF	3, 10	32x3, 64x3, 128x3	100k, 300k
RealNVP	3, 10	32x3, 64x3, 128x3	100k, 300k



Wasserstein distance

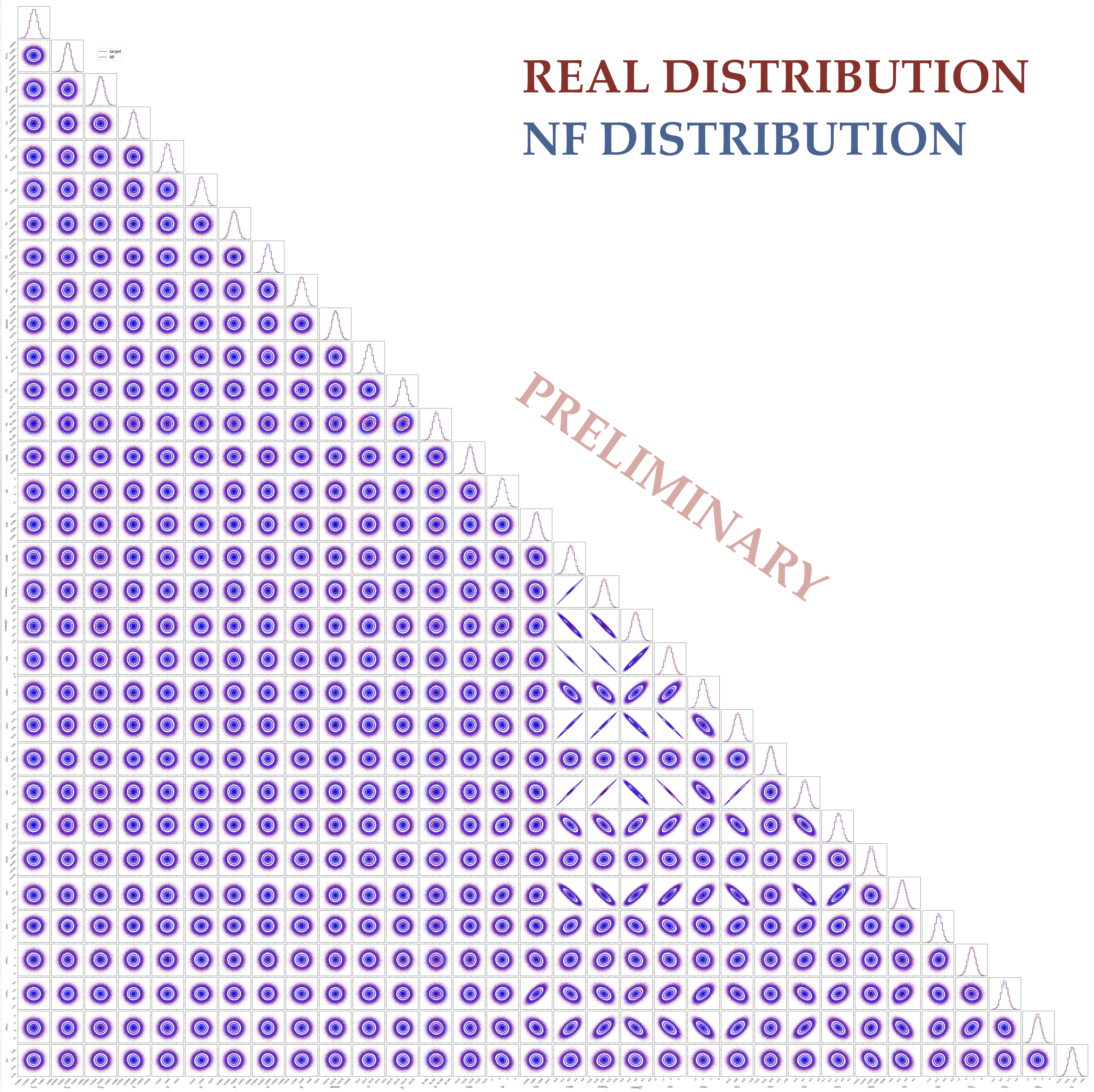


KS test

Let's get real…

Learning LHC Likelihoods.

EW-fit (32 dims)



Likelihood of global EW-fit at LHC:

18 parameters of interest (Wilson coefficients)

14 nuisance parameters (uncertainties)

Data provided by authors -> arXiv:1710.05402

Weapon of choice:

MAF, 3 Bijectors, 128x3 layers, 650k samples

Metrics:

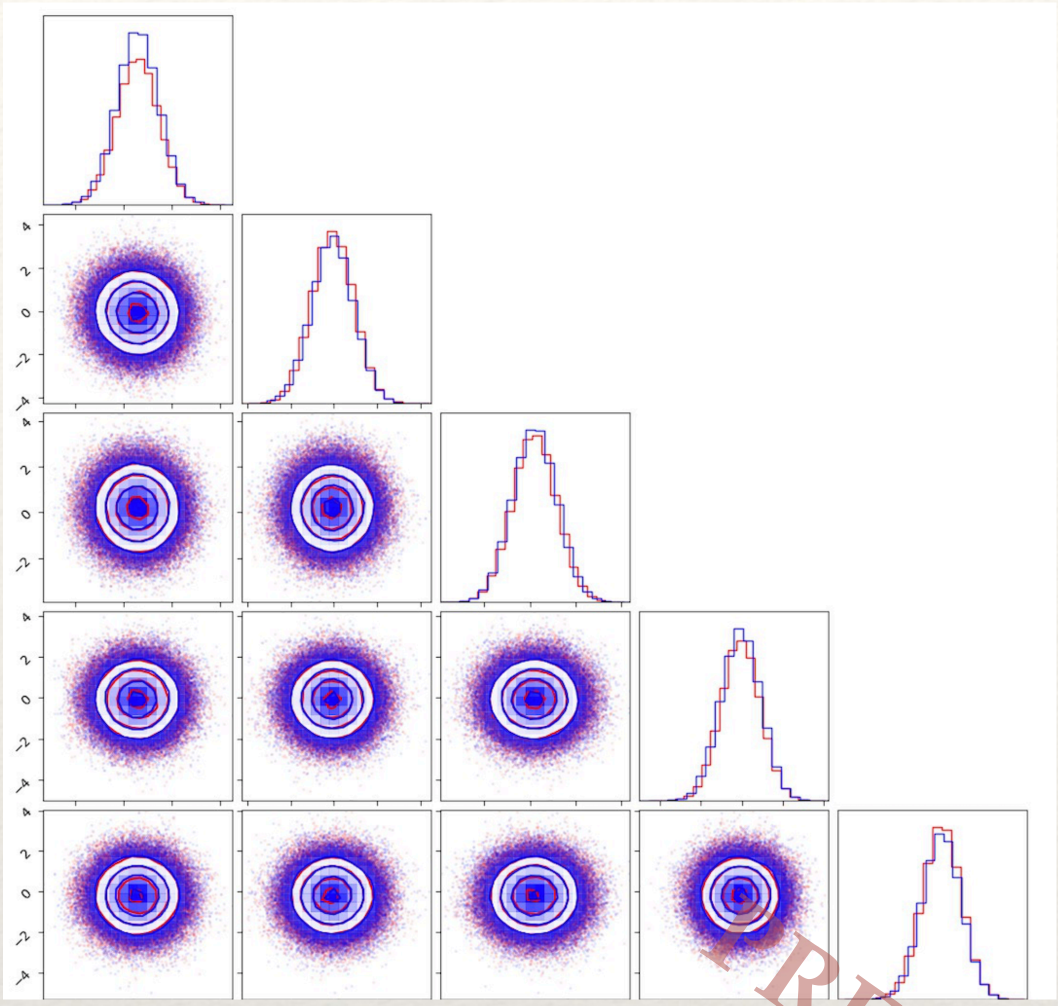
Wasserstein distance: .000315

KS test: 0.484

Training time: 2.8 hrs.

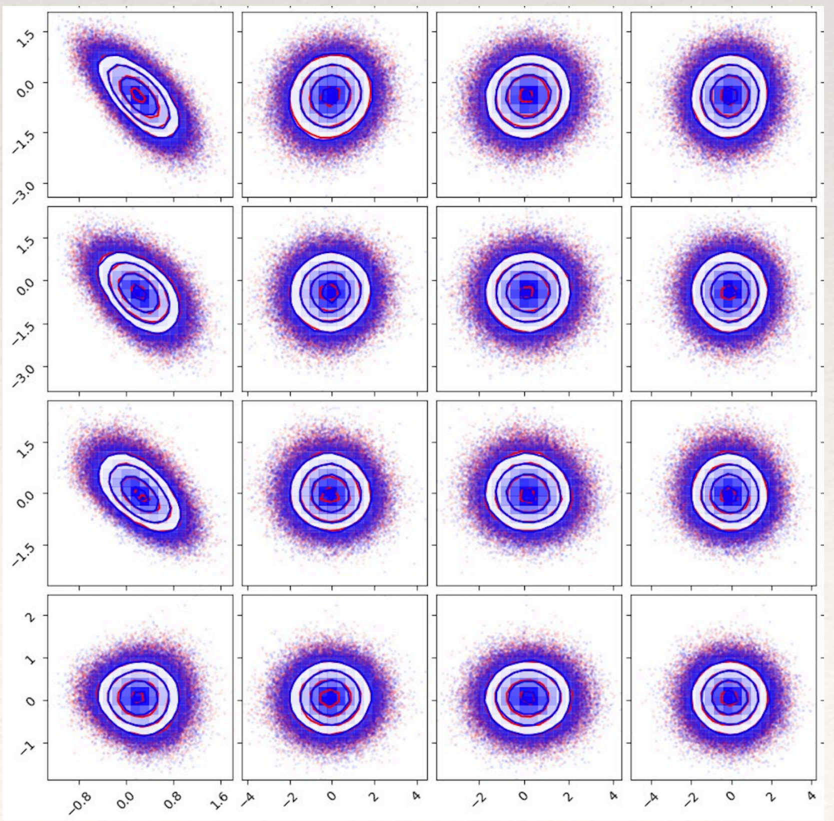
THE RESEMBLANCE IS GREAT!

LHC-like New Physics search (95 dims).

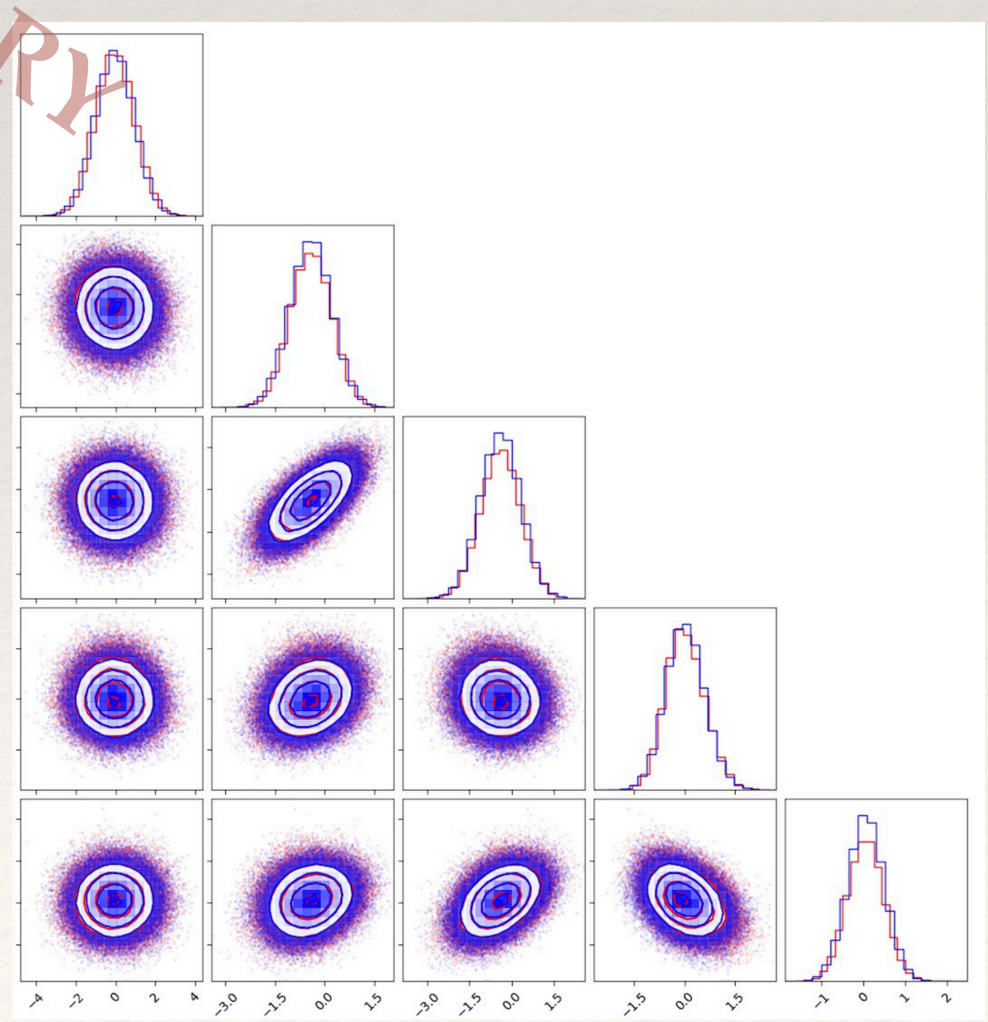


REAL DISTRIBUTION
NF DISTRIBUTION

⋮ …



…



Likelihood of LHC-like New Physics search:
1 parameter of interest.
94 nuisance parameters.

arXiv:1911.03305
arXiv:1809.05548

Weapon of choice:
MAF, 3 Bijectors, 128x3 layers, 500k samples

Metrics:
Wasserstein distance: .0067
KS test: 0.507
Training time: 9.3 mins

ANOTHER GREAT RESEMBLANCE!

Conclusion.

- Can Normalizing Flows (NFs) help us on these endeavours?…**YES**

At high dimensions…

- A-NSF can vey effectively describe (uncorrelated) high-dimensional complex distributions.
- Fully correlated distributions are harder to describe, but autoregressive flows hold their ground.

- **We presented for the first time Unsupervised Learning of LHC Likelihood functions using Normalizing Flows!!**

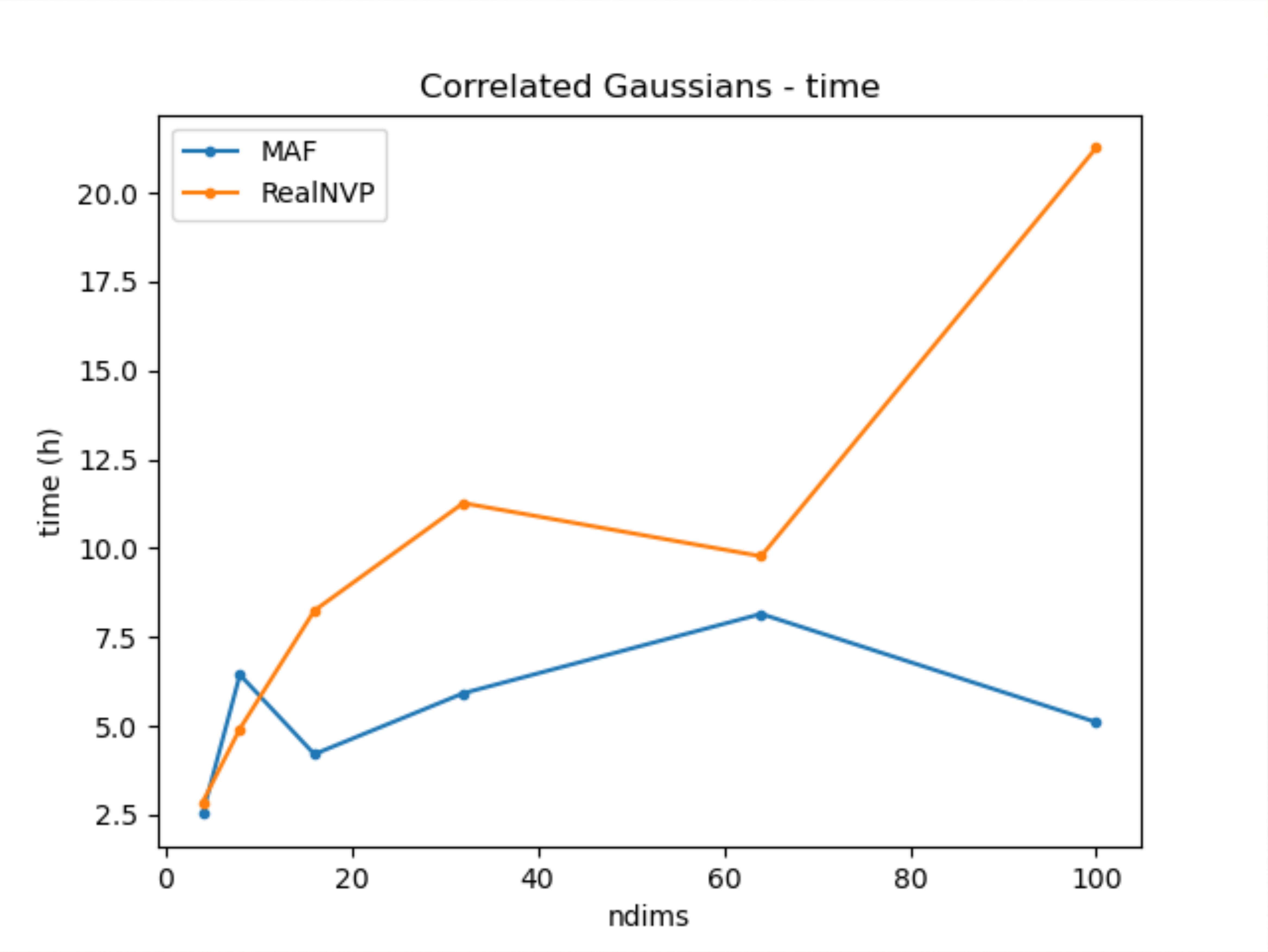
…more to come

THANK YOU!

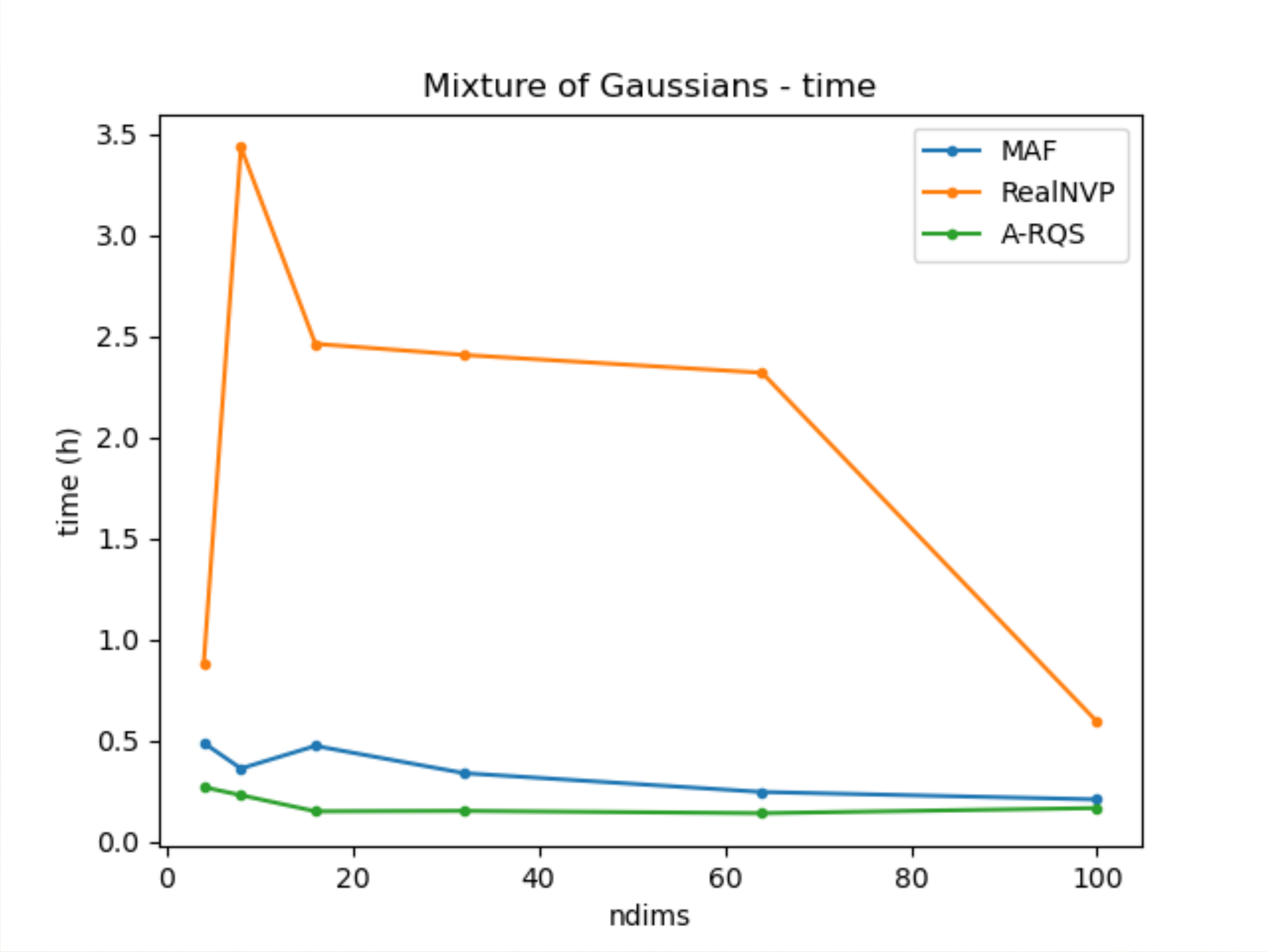
BACK UP

Training time.

Correlated Gaussians



Mixture of Gaussians



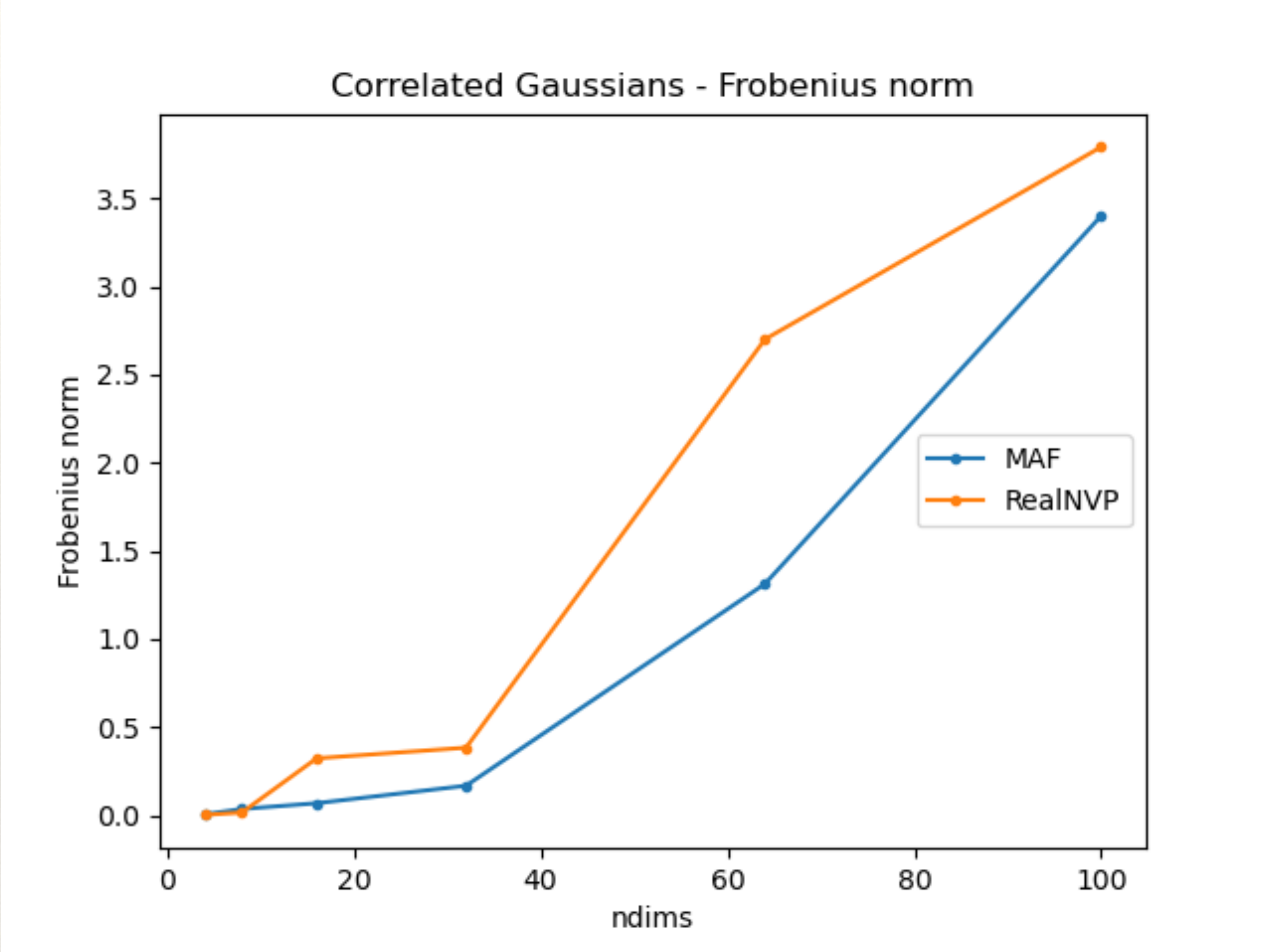
Are correlations well learned?

Frobenius norm.

$$||A|| = (\sum_{i,j} \text{abs}(a_{i,j})^2)^{1/2}$$

where

$$A = \text{Corr}_{NF} - \text{Corr}_{real}$$

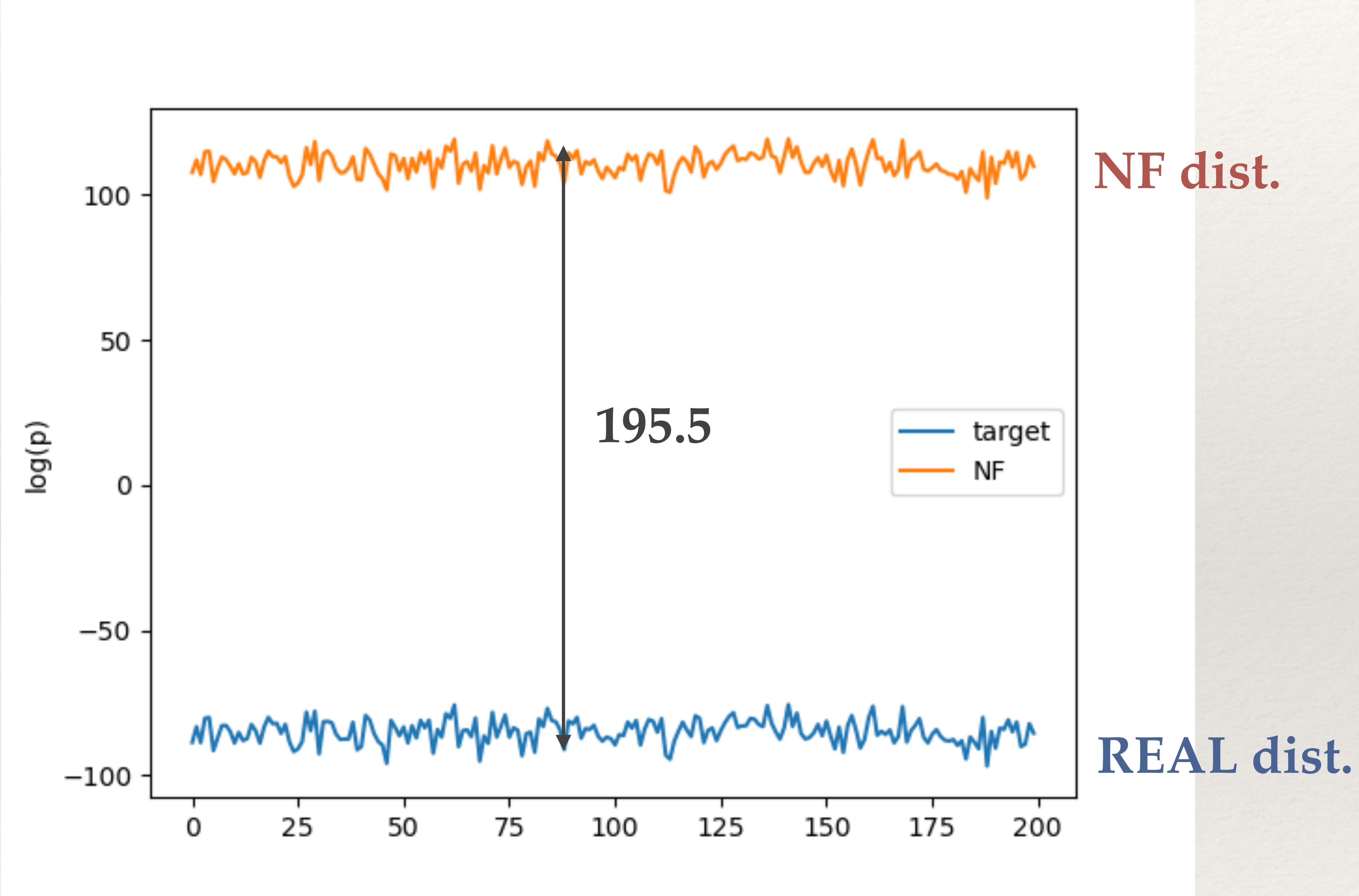


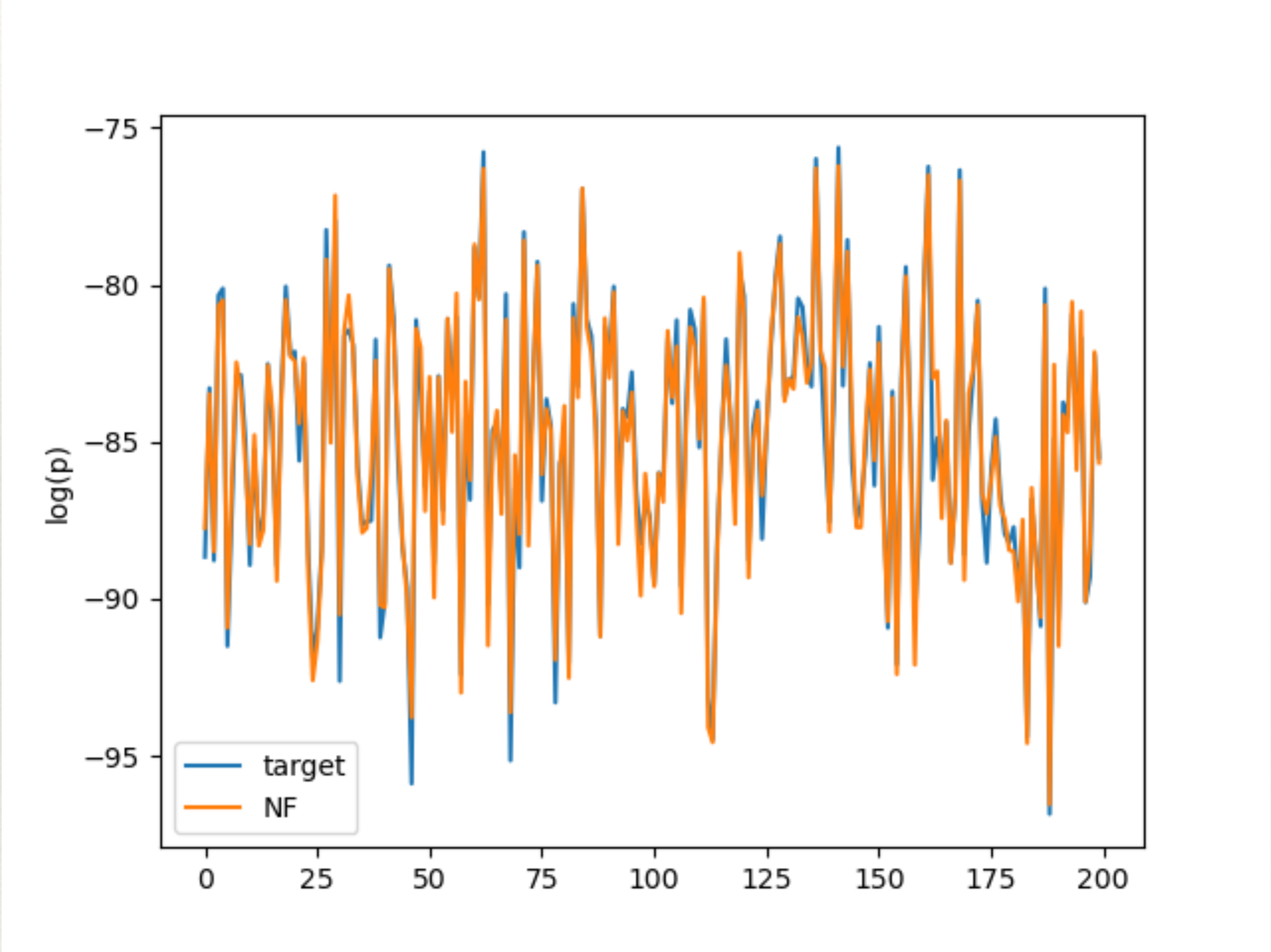
WARNING: Not dimension-scaled

On probability volumes

EXAMPLE: EW-Fit.

KL divergence=-195.5, however
Correlation between log probs=.996





Normalization might be required!

Introduction.

Let's get formal…

- If Z is a random variable with pdf p_Z, g is an invertible function such that $Y = g(Z)$ and $f = g^{-1}$, then we can obtain the pdf p_Y of the random variable Y as

$$p_Y(y) = p_Z(f(y))\,|\det(Df(y))| = p_Z(f(y))\,|\det(Dg(f(y)))|^{-1}$$

where $Dg(z) = \frac{\partial g}{\partial z}$ $Df(y) = \frac{\partial f}{\partial y}$

Jacobians

- N transformations are possible since…

$$f = f_1 \circ \ldots f_{N-1} \circ f_N$$

$$\det Df(y) = \Pi_{i=1}^{N} \det(Df_i(x_i))$$

where $x_i = g_i \circ \ldots \circ g_1(z) = f_{i+1} \circ \ldots \circ f_N(y)$

- Since p_Z is parametrised by ϕ and the bijector g by θ, we can compute the **log probability** of some measured data $\mathcal{D} = \{y^{(i)}\}_{i=1}^{M}$ given the parameters $\Theta = (\theta, \phi)$ as

$$\log p(\mathcal{D}\,|\,\Theta) = \sum_{i=1}^{M} \log p_Y(y^{(i)}\,|\,\Theta) = \sum_{i=1}^{M} \log p_Z(f(y^{(i)}\,|\,\theta)\,|\,\phi) + \log|\det Df(y^{(i)}\,|\,\theta)|$$