Flexible conditional normalizing-flow distributions over manifolds: the jammy-flows toolkit

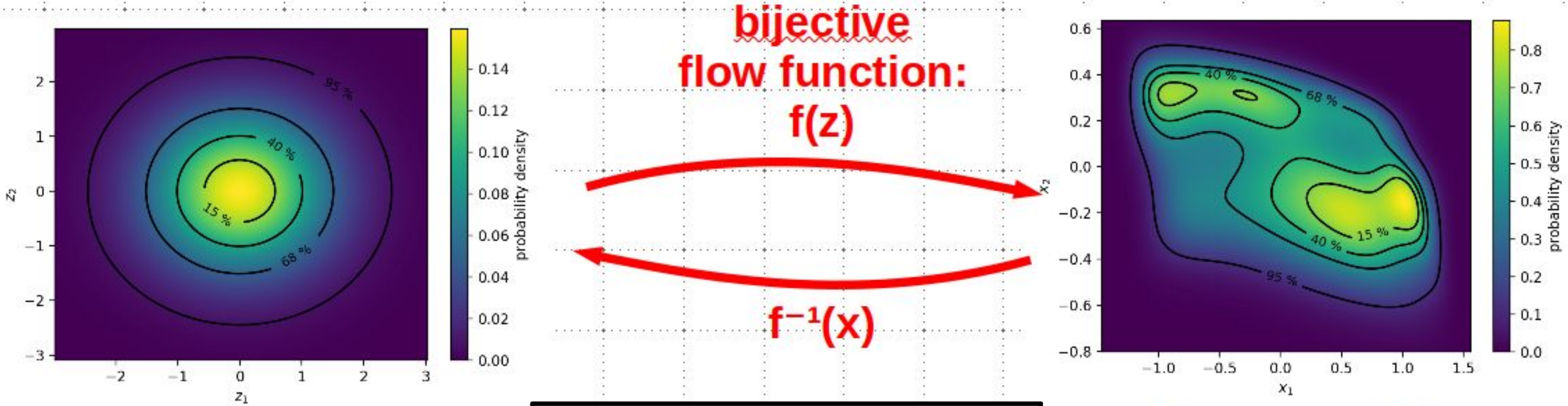
Thorsten Glüsenkamp, May 1st 2024, EuCAIFCon flash talk

UPPSALA UNIVERSITET

CARL TRYGGERS STIFTELSE
FÖR VETENSKAPLIG FORSKNING

EUROPEAN AI FOR FUNDAMENTAL PHYSICS CONFERENCE
EuCAIFCon 2024

Normalizing flows are great, but….



Auxiliary base space "z":

$p_0(z) = \mathcal{N}(z; \mathbf{0}, \mathbb{1})$

Target space "x":

$p_\theta(x|y) = p_0(f^{-1}_{g_\theta(y)}(x)) \cdot |\det J^{-1}_{g_\theta(y)}(x)|$

… implementations are

- often scattered around in their own repositories
- often not really suited for physics ("image-focused flows")
- often complicated to set up, especially the conditioning

Poster No: 110

Jammy Flows

([1] https://github.com/thoglu/jammy_flows)

Joint **A**utoregressive **M(MY)**anifold normalizing **flows**

setup complex normalizing flows in 1 line of code [1]

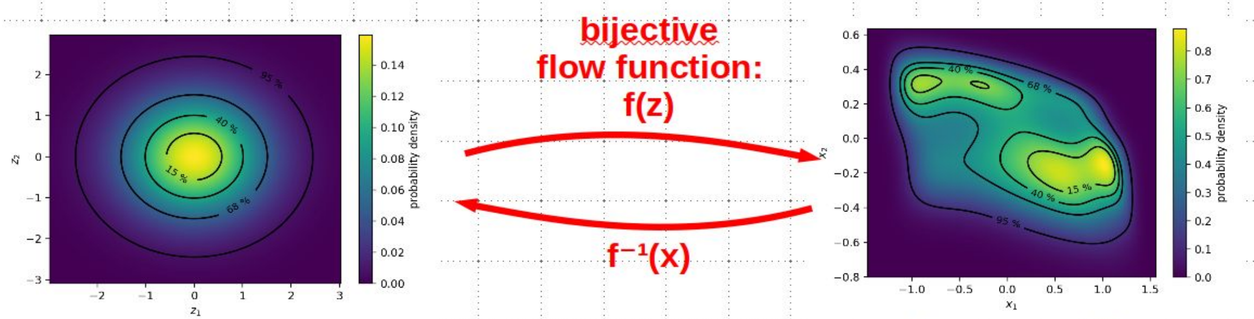
Euclidean ($\mathbb{R}^4$)

$$f(\vec{z}) = \Phi_t(\Phi_g(\Phi_g(\vec{z})))$$

Arg 1: Manifold

Arg 2: Flow function f(z)

```
pdf=jammy_flows.pdf("e4", "ggt")
```



Auxiliary base space "z":

$p_0(z) = \mathcal{N}(z; \mathbf{0}, \mathbb{1})$

Target space "x":

$p_\theta(x|y) = p_0(f^{-1}_{g_\theta(y)}(x)) \cdot |\det J^{-1}_{g_\theta(y)}(x)|$

Poster No: 110

Supports various manifolds + autoregressive linking

Joint Autoregressive M(MY)anifold normalizing flows
setup complex normalizing flows in 1 line of code [1]

Euclidean ($\mathbb{R}^4$)

$$f(\vec{z}) = \Phi_t(\Phi_g(\Phi_g(\vec{z})))$$

Arg 1: Manifold

Arg 2: Flow function f(z)

```
pdf=jammy_flows.pdf("e4", "ggt")
```

Autoregressive ($S^2 \times \Delta^2 \times \mathbb{R}^3$)

$$f(\vec{z}) = \begin{bmatrix} \Phi_n(\Phi_n(\vec{z_1})) \\ \Phi_{w,x1}(\vec{z_2}) \\ \Phi_{g,x1,x2}(\Phi_{g,x1,x2}(\Phi_{g,x1,x2}(\Phi_{g,x1,x2}(\vec{z_3})))) \end{bmatrix}$$

```
pdf=jammy_flows.pdf("s2+c2+e4", "nn+w+gggg")
```

2-sphere (S^2)

$$f(\vec{z}) = \Phi_v(\Phi_v(\Phi_v(\vec{z})))$$

```
pdf=jammy_flows.pdf("s2", "vvv")
```

Interval (-3 to 3)

$$f(\vec{z}) = \Phi_r(\Phi_r(\Phi_r(\Phi_r(\Phi_r(\vec{z})))))$$

```
pdf=jammy_flows.pdf("i1_-3_3", "rrrrr")
```

Probability simplex (Δ^3)

$$f(\vec{z}) = \Phi_w(\Phi_w(\vec{z}))$$

```
pdf=jammy_flows.pdf("c3", "ww")
```

Syntax reflects normalizing flow-defining function f(z)

Flow abbreviations:

- "t" – affine flow
- "g" – Gaussianzation flows [2]
- "v" – Exponential map flows [3]
- "n" – Recursive flows on the 2-sphere [5]
- "w" – Simplex flow [6]
- "r" – neural spline flows [4]

+ a few more in the package

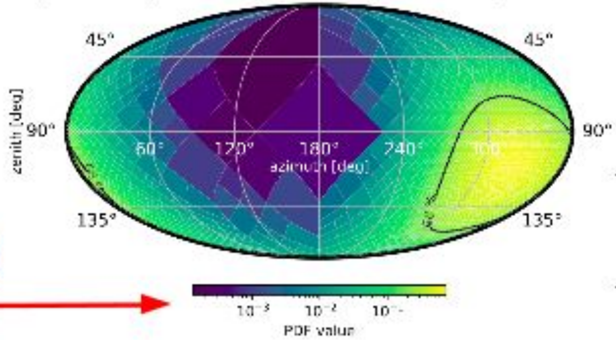
Poster No: 110

Features

Many state-of-the-art normalizing flows that can be autoregressively linked, customized, and made conditional. Autoregressive routing generalizes structure from inverse autoregressive flows [7].

Convenience functions for ...

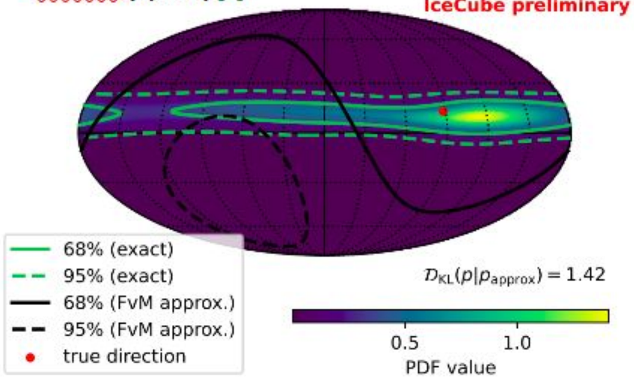
- $1^{st} + 2^{nd}$ moments
- asymmetry measure (higher order moments) (see [9])
- entropy
- coverage (Gaussian base used for all manifolds – see [8])
- plotting (adaptive healpix for spherical PDFs)
- easy to add new layers

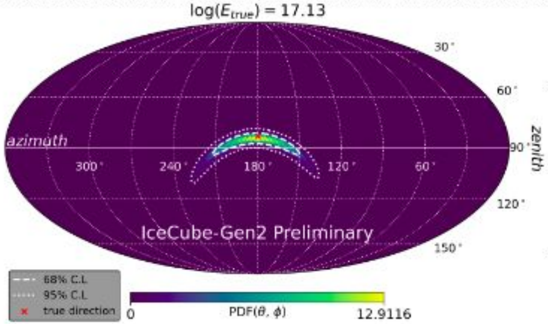


Poster No: 110

Applied in IceCube and IceCube-Gen2 for per-event posterior inference

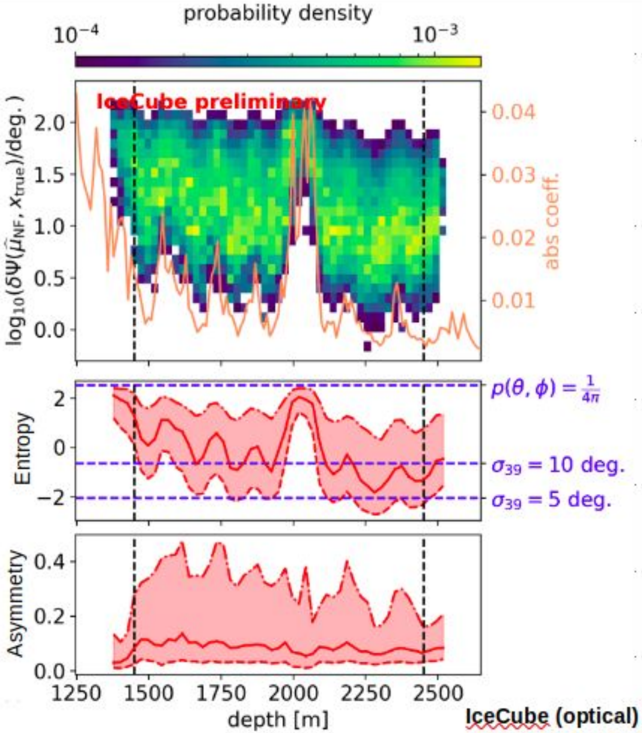
IceCube (optical) [9]





IceCube-Gen2 (radio) [10]

Directional posterior resolution vs depth (~50k electron neutrinos)



IceCube (optical) [9]

Posterior properties reflect detector properties (South Pole ice properties vary with depth)

Poster No: 110