Madrid Innovation Lab

【NORMALIZING FLOWS: MADRID MACHINE LEARNING MEETUP】

〈20.03.2024 / 18.30h 〉

Madrid Innovation Lab

↳ **Javier Fabre**

Research Engineer

Desilico Labs.

Calle Bravo Murillo 37-39, Madrid

ARQUIMEA

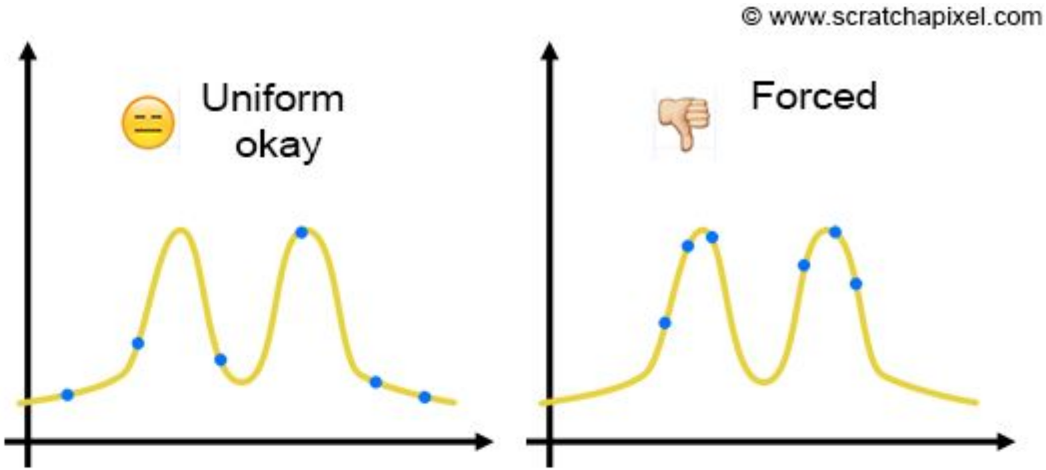
Universidad Rey Juan Carlos

POLITÉCNICA

apolo

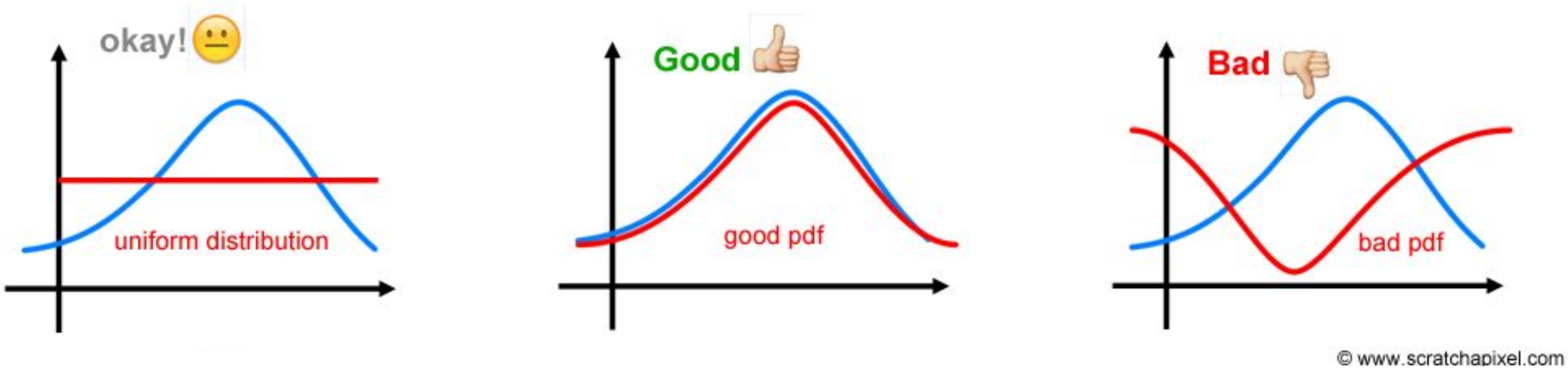
Why importance sampling?

- More samples where we have more information
- Useful for:
  - Monte Carlo methods
  - Selecting similar data to provided one



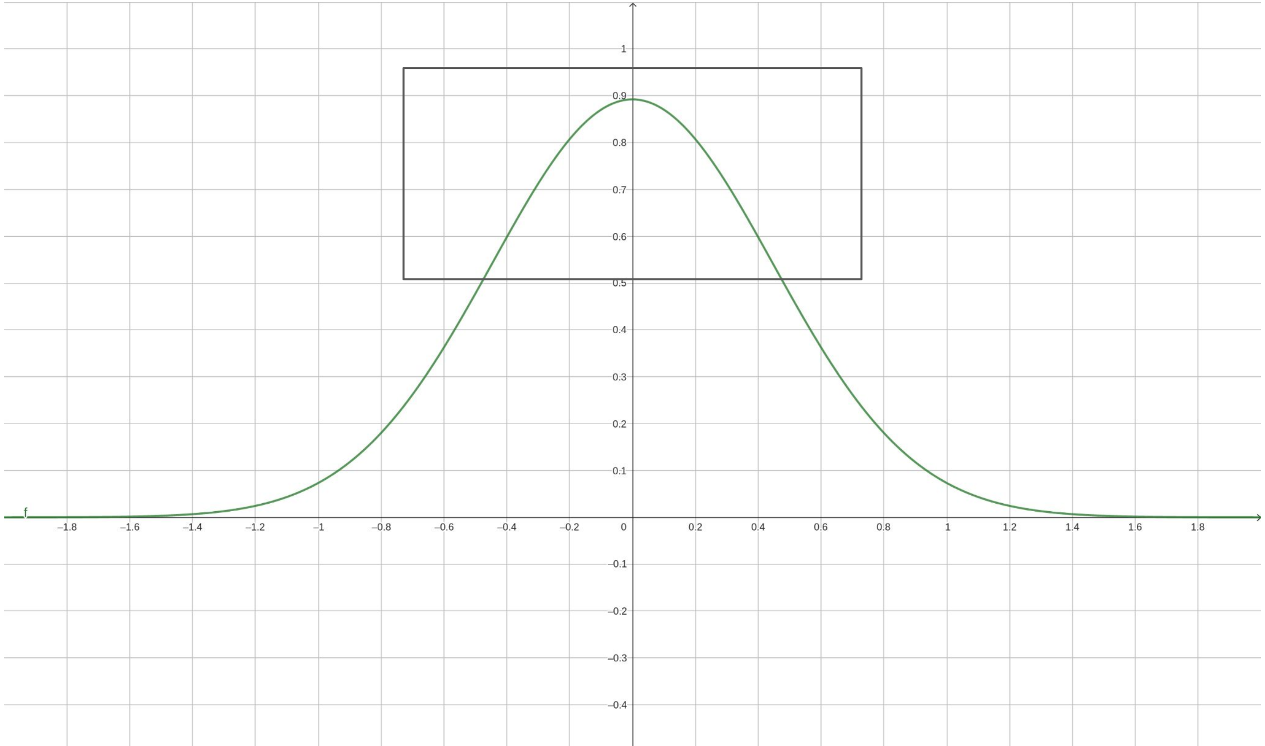
Why importance sampling?

- More samples where we have more information
- Useful for:
  - Monte Carlo methods
  - Selecting similar data to provided one



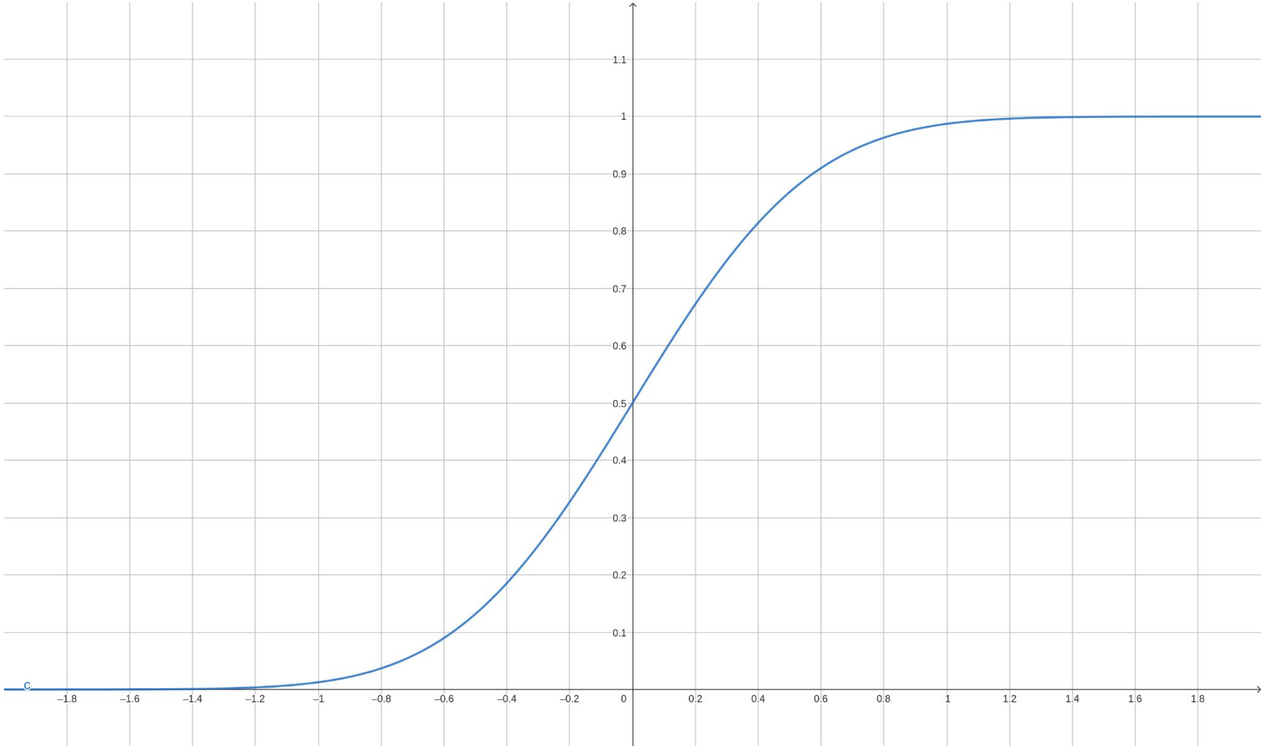
How we sample?

- Cumulative Distribution Function



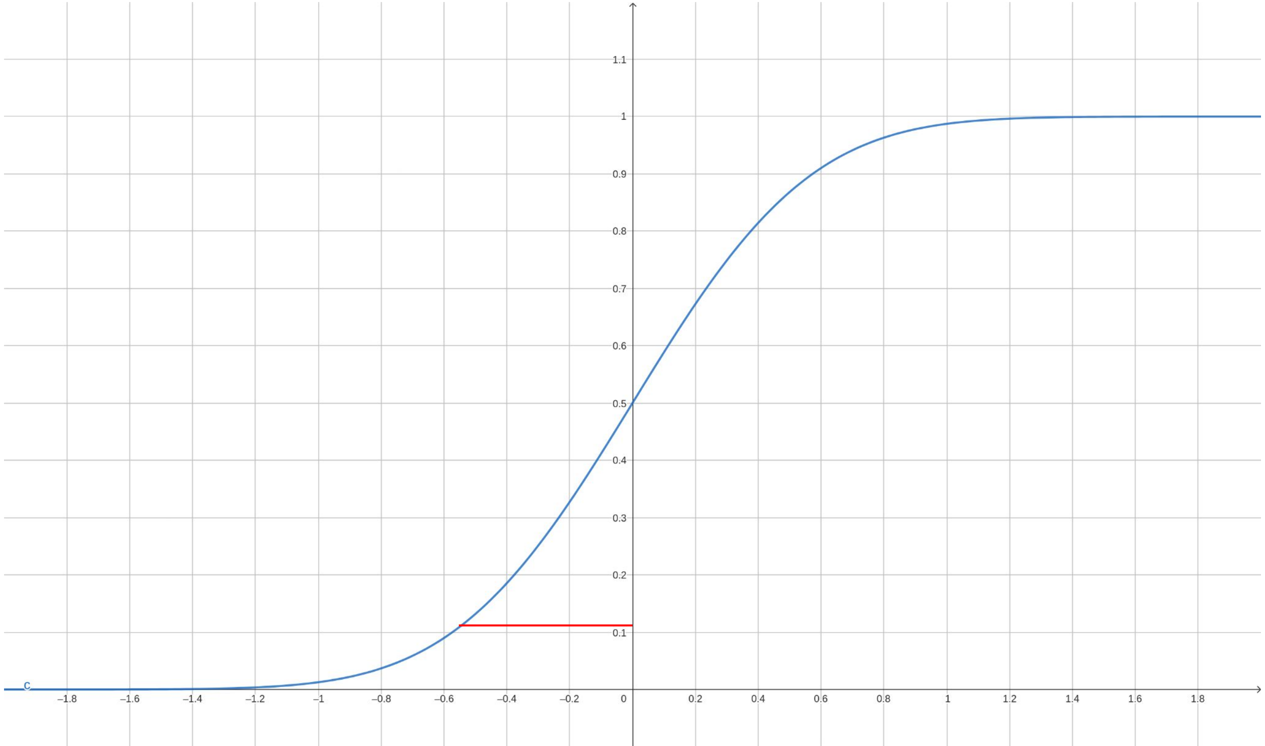
How we sample?

- Cumulative Distribution Function



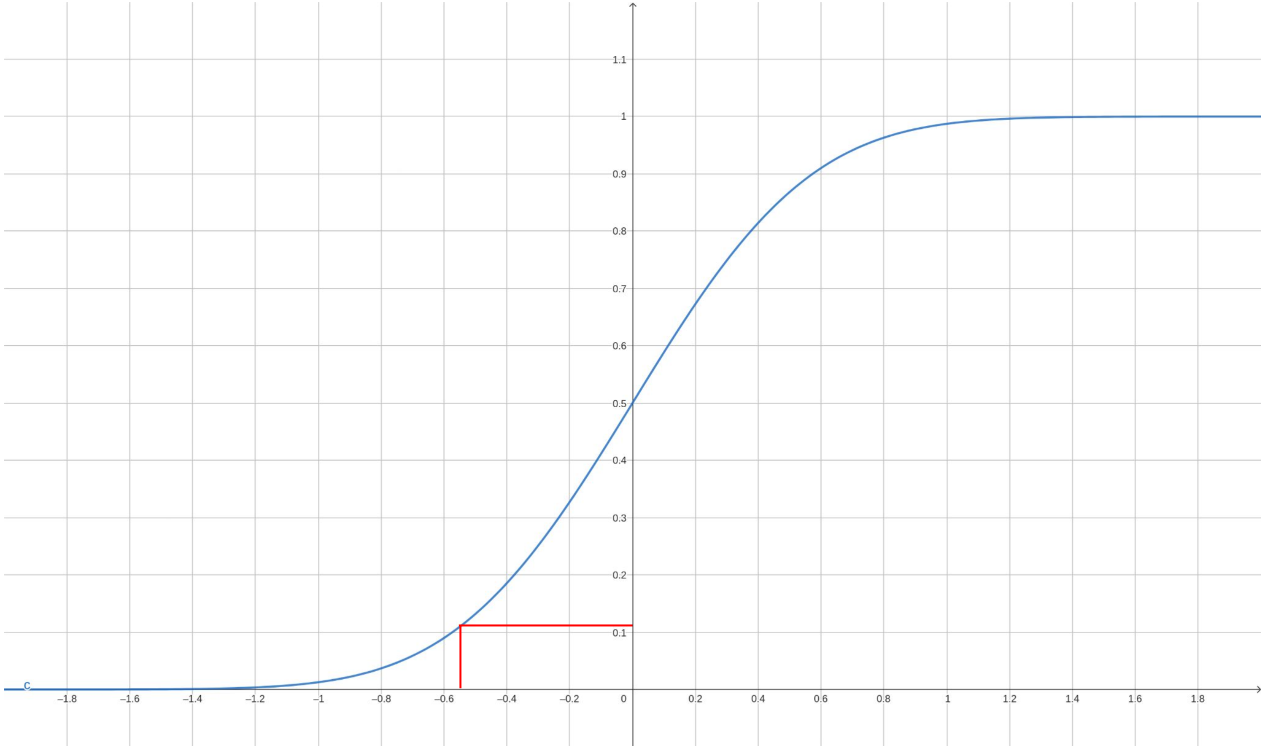
How we sample?

- Cumulative Distribution Function



How we sample?

- Cumulative Distribution Function



How we sample?

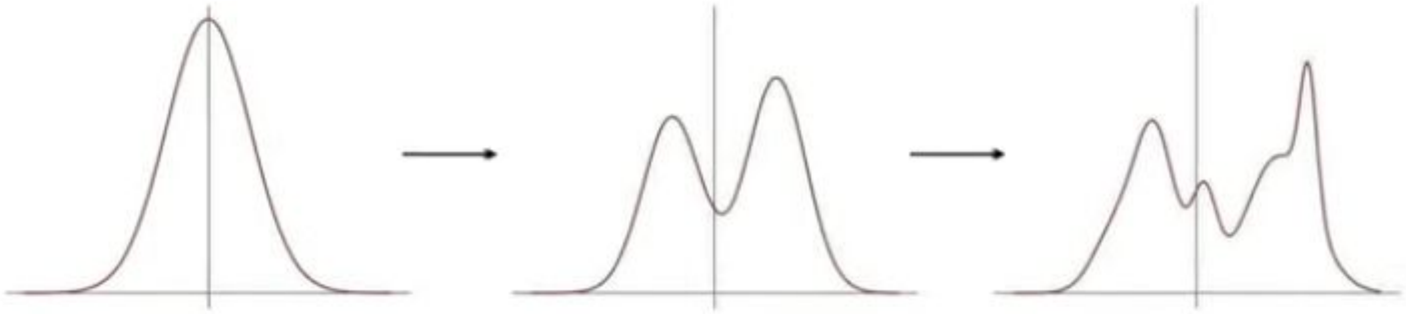
- Cumulative Distribution Function
  - What if we don’t know the function?
  - What if the CDF is not analytic??
  - Maybe it’s too complex???

How we sample?

- Cumulative Distribution Function
- Piecewise-Constant Distribution (see related links)
  - Binary search
  - Expensive

Normalizing flows

- We know a nice an easy function with nice properties
- We want to fit a complex function
- We get the complex one as transformations of the simple function



Normalizing flows

- We know a nice an easy function with nice properties
- We want to fit a complex function
- We get the complex one as transformations of the simple function

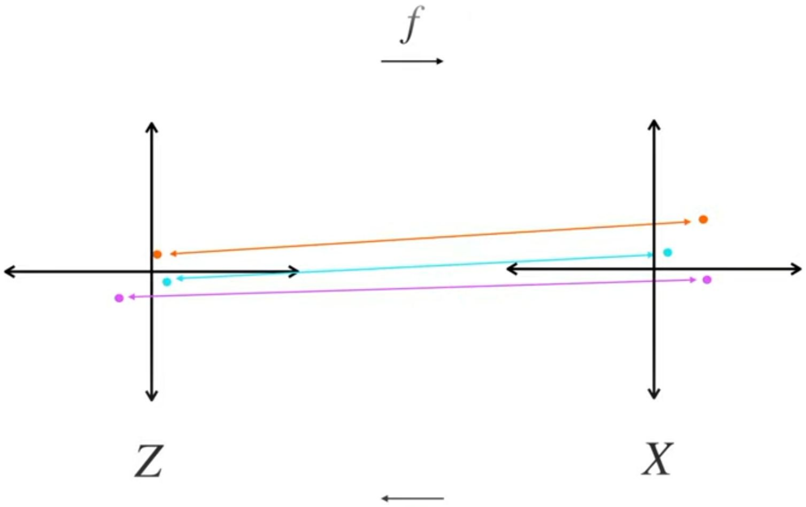
$$z \sim p_\theta(z) = \mathcal{N}(z; 0, I)$$

$$x = f_\theta(z) = f_K \circ \ldots \circ f_2 \circ f_1(z)$$

each f_i is invertible (bijective)

Normalizing flows

- We know a nice an easy function with nice properties
- We want to fit a complex function
- We get the complex one as transformations of the simple function



$$p_\theta(x) \stackrel{?}{=} p_\theta(f_\theta^{-1}(x))$$

Normalizing flows

- We know a nice an easy function with nice properties
- We want to fit a complex function
- We get the complex one as transformations of the simple function

$f : Z \to X$, f is invertible

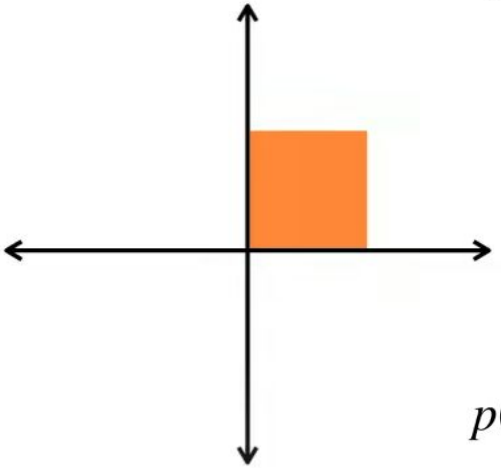
$p_\theta(z)$ defined over $z \in Z$

Change of variables formula:

$$p_\theta(x) = p_\theta(z) \left| \det\left(\frac{\partial z}{\partial x} \right) \right|$$

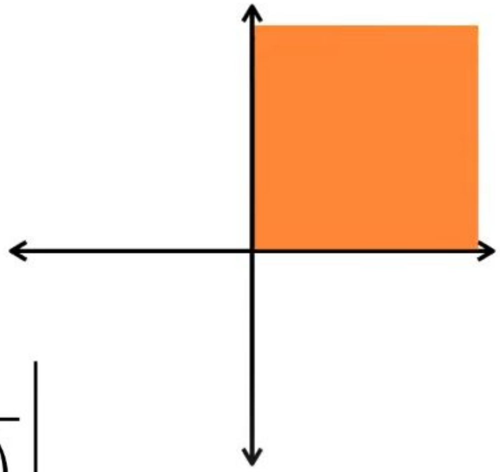
Normalizing flows

$$J = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$



$$p(x) = p(z)\left|\frac{1}{\det\left(\frac{\partial x}{\partial z}\right)}\right|$$

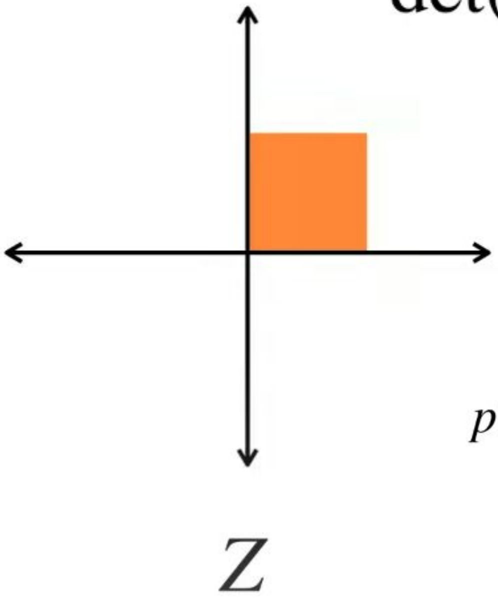
Z



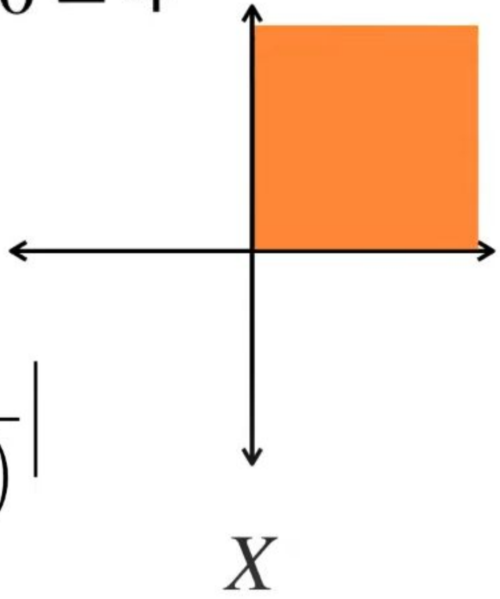
X

Normalizing flows

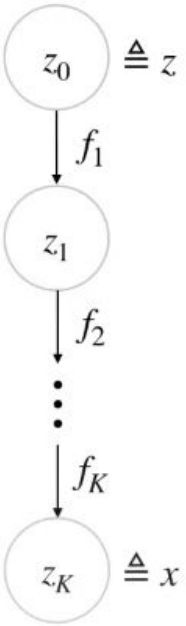
$$\det(J) = 2 \cdot 2 - 0 \cdot 0 = 4$$



$$p(x) = p(z) \left| \frac{1}{\det\left(\frac{\partial x}{\partial z}\right)} \right|$$



Normalizing flows



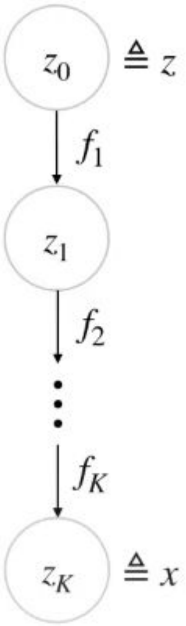
"normalizing flow"

$$z \sim p_\theta(z)$$

$$x = f_\theta(z) = f_K \circ \ldots \circ f_2 \circ f_1(z)$$

$$p_\theta(x) = p_\theta(z) \prod_1^K \left| \det\left(\frac{\partial f_i^{-1}}{\partial z_i} \right) \right| = p_\theta(z) \left| \det\left(\frac{\partial f^{-1}}{\partial x} \right) \right|$$

Normalizing flows



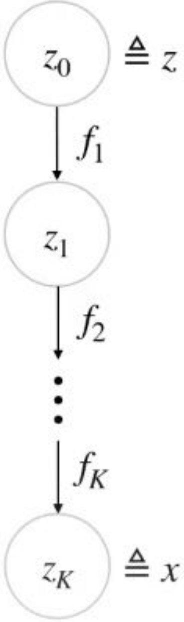
“normalizing flow”

$$z \sim p_\theta(z)$$

$$x = f_\theta(z) = f_K \circ \ldots \circ f_2 \circ f_1(z)$$

$$\log p_\theta(x) = \log p_\theta(z) + \log \left| \det\left(\frac{\partial f^{-1}}{\partial x} \right) \right|$$

Normalizing flows



"normalizing flow"

$$z \sim p_\theta(z)$$

$$x = f_\theta(z) = f_K \circ \ldots \circ f_2 \circ f_1(z)$$

$$\log p_\theta(x) = \log p_\theta(z) + \sum_{i=1}^{K} \log \left| \det\left(\frac{\partial f_i^{-1}}{\partial z_i} \right) \right|$$

Normalizing flows

Variational Autoencoders

Kingma et al., 2014

- lower bound on log-likelihood (ELBO)
- approximate posterior: $q_\phi(z \mid x)$

Generative Adversarial Networks

Goodfellow et al., 2014

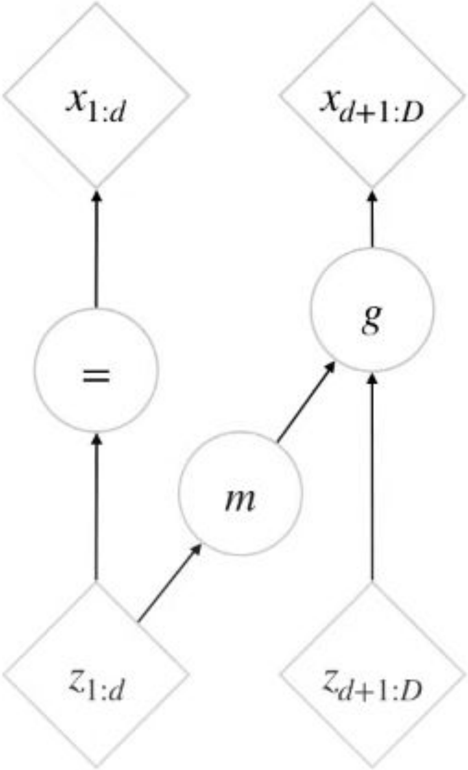
- no log-likelihood evaluation
- no latent variable inference

$$\log p_\theta(x) = \log p_\theta(z) + \sum_{i=1}^{K} \log \left| \det\left(\frac{\partial f_i^{-1}}{\partial z_i} \right) \right|$$

exact log-likelihood evaluation

exact posterior inference (via $z = f^{-1}(x)$)

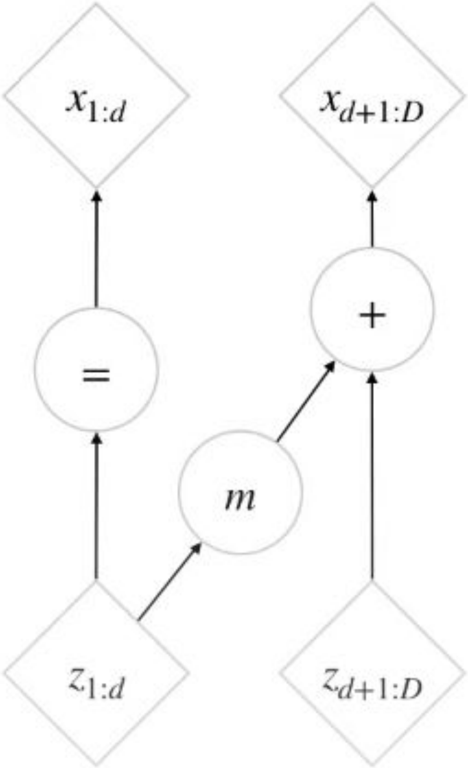
Coupling layer



$$x_{1:d} = z_{1:d}$$

$$x_{d+1:D} = g(z_{d+1:D}; m(z_{1:d}))$$

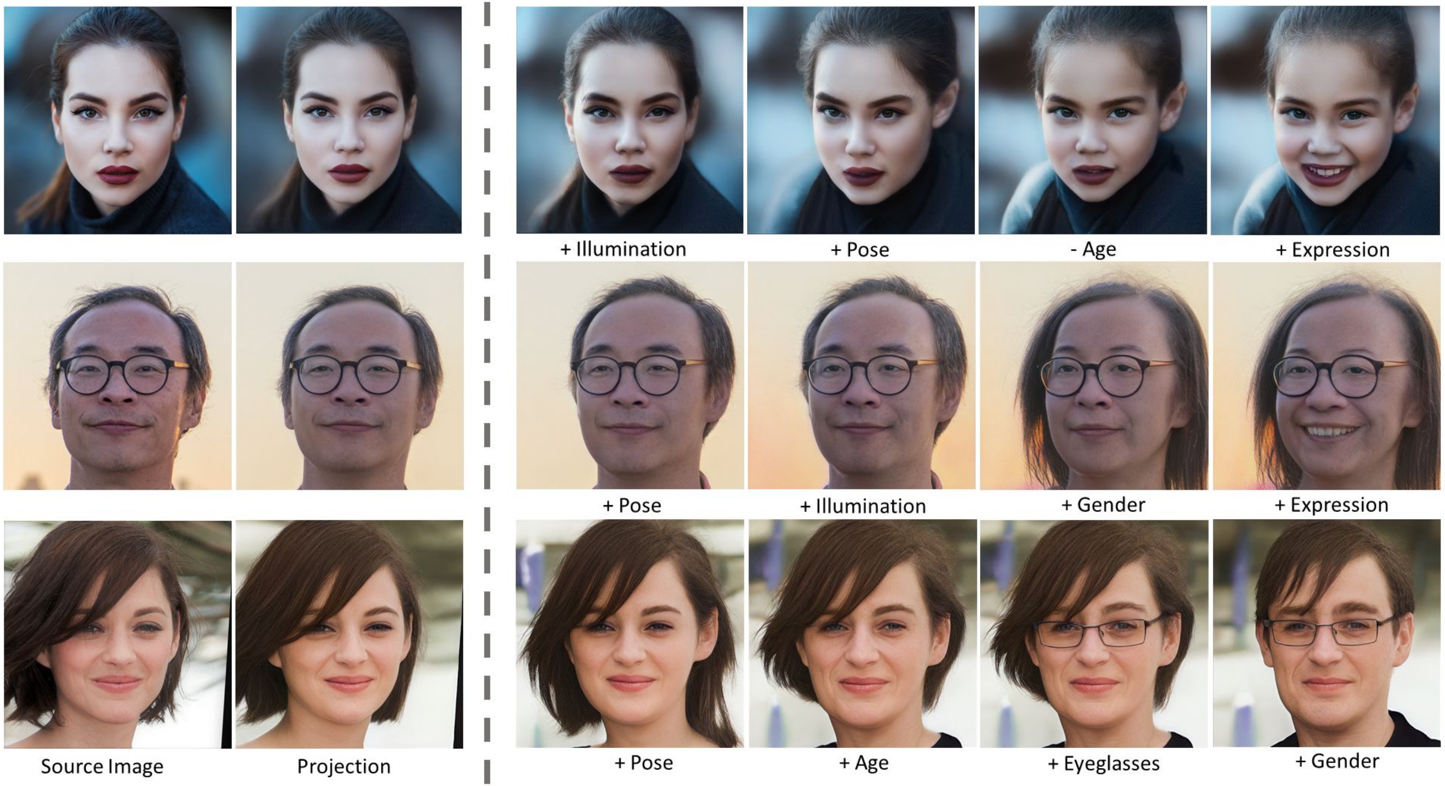
Additive coupling layer



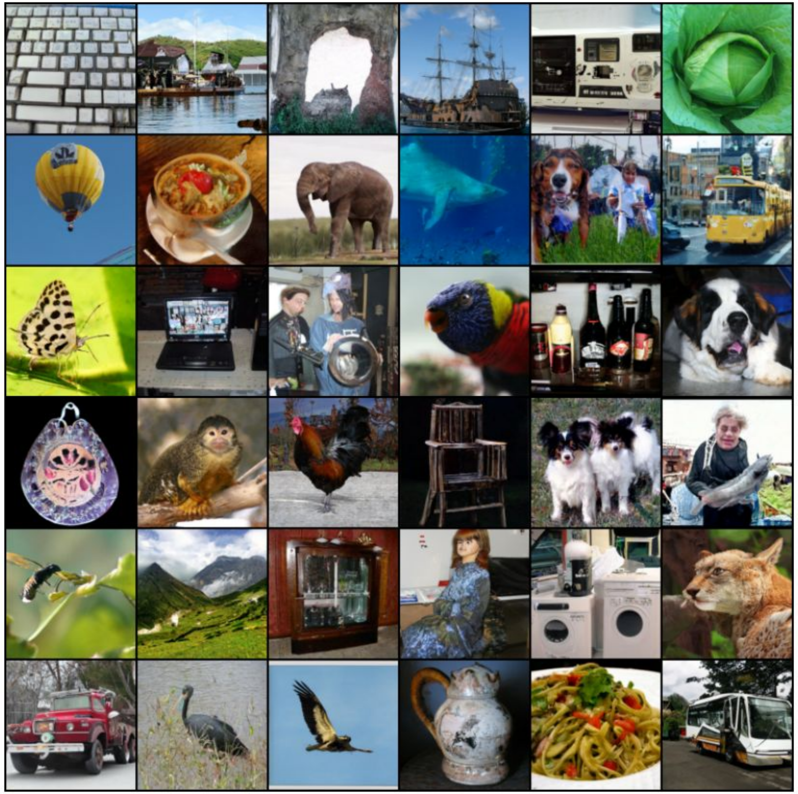
$$x_{1:d} = z_{1:d}$$

$$x_{d+1:D} = z_{d+1:D} + m(z_{1:d})$$

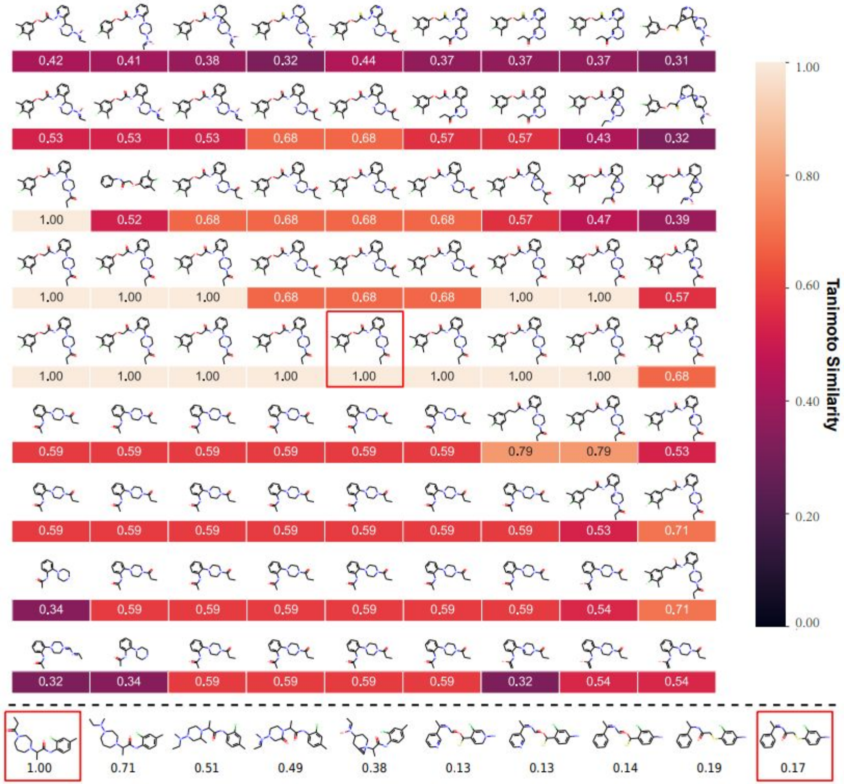
Use cases: StyleFlow



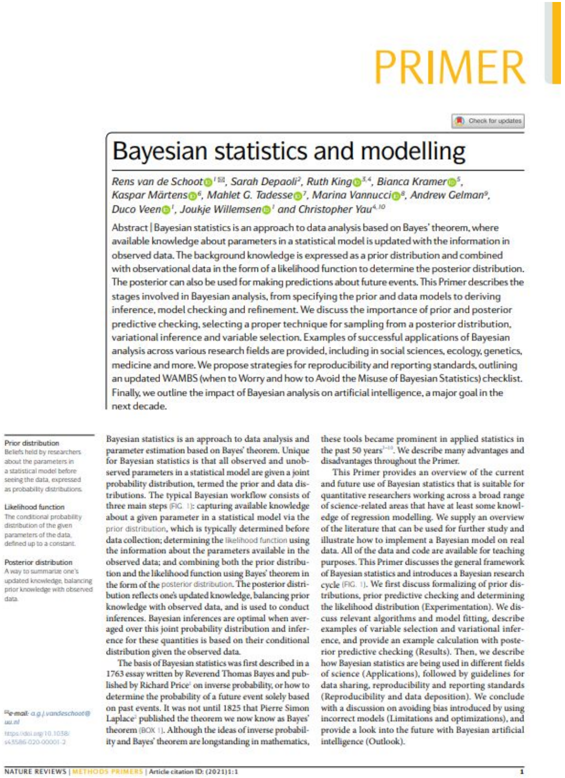
Use cases: Image Generation



Use cases: [Molecular Graphs]



Use cases: Bayesian modeling


PRIMER

Check for updates

Bayesian statistics and modelling

Rens van de Schoot[1]✉, Sarah Depaoli[2], Ruth King[3,4], Bianca Kramer[5], Kaspar Märtens[6], Mahlet G. Tadesse[7], Marina Vannucci[8], Andrew Gelman[9], Duco Veen[1], Joukje Willemsen[1] and Christopher Yau[4,10]

Abstract | Bayesian statistics is an approach to data analysis based on Bayes' theorem, where available knowledge about parameters in a statistical model is updated with the information in observed data. The background knowledge is expressed as a prior distribution and combined with observational data in the form of a likelihood function to determine the posterior distribution. The posterior can also be used for making predictions about future events. This Primer describes the stages involved in Bayesian analysis, from specifying the prior and data models to deriving inference, model checking and refinement. We discuss the importance of prior and posterior predictive checking, selecting a proper technique for sampling from a posterior distribution, variational inference and variable selection. Examples of successful applications of Bayesian analysis across various research fields are provided, including in social sciences, ecology, genetics, medicine and more. We propose strategies for reproducibility and reporting standards, outlining an updated WAMBS (when to Worry and how to Avoid the Misuse of Bayesian Statistics) checklist. Finally, we outline the impact of Bayesian analysis on artificial intelligence, a major goal in the next decade.

Prior distribution
Beliefs held by researchers about the parameters in a statistical model before seeing the data, expressed as probability distributions.

Likelihood function
The conditional probability distribution of the given parameters of the data, defined up to a constant.

Posterior distribution
A way to summarize one's updated knowledge, balancing prior knowledge with observed data.

Bayesian statistics is an approach to data analysis and parameter estimation based on Bayes' theorem. Unique for Bayesian statistics is that all observed and unobserved parameters in a statistical model are given a joint probability distribution, termed the prior and data distributions. The typical Bayesian workflow consists of three main steps (FIG. 1): capturing available knowledge about a given parameter in a statistical model via the prior distribution, which is typically determined before data collection; determining the likelihood function using the information about the parameters available in the observed data; and combining both the prior distribution and the likelihood function using Bayes' theorem in the form of the posterior distribution. The posterior distribution reflects one's updated knowledge, balancing prior knowledge with observed data, and is used to conduct inferences. Bayesian inferences are optimal when averaged over this joint probability distribution and inference for these quantities is based on their conditional distribution given the observed data.

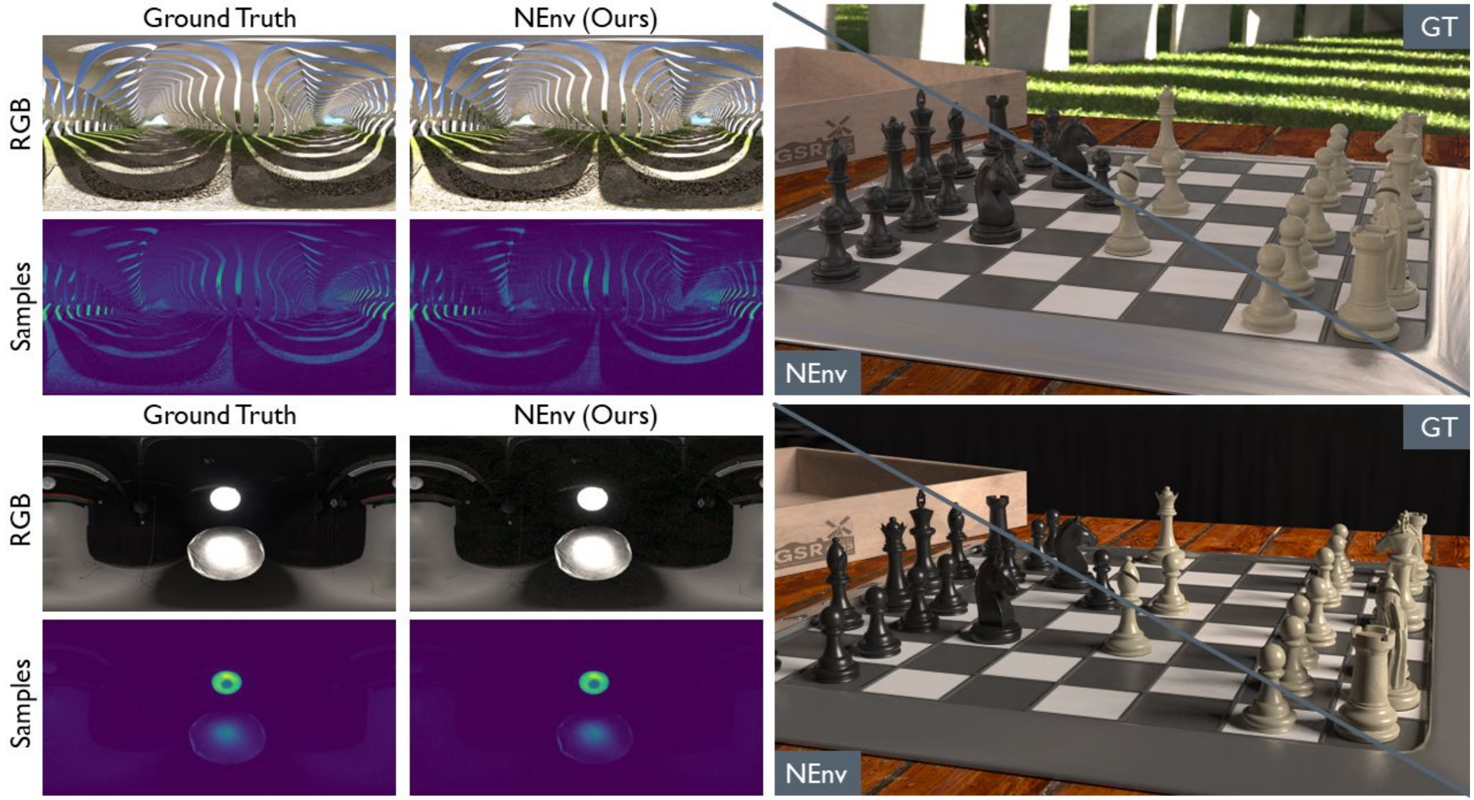
The basis of Bayesian statistics was first described in a 1763 essay written by Reverend Thomas Bayes and published by Richard Price[1] on inverse probability, or how to determine the probability of a future event solely based on past events. It was not until 1825 that Pierre Simon Laplace[2] published the theorem we now know as Bayes' theorem (BOX 1). Although the ideas of inverse probability and Bayes' theorem are longstanding in mathematics, these tools became prominent in applied statistics in the past 50 years[3–10]. We describe many advantages and disadvantages throughout the Primer.

This Primer provides an overview of the current and future use of Bayesian statistics that is suitable for quantitative researchers working across a broad range of science-related areas that have at least some knowledge of regression modelling. We supply an overview of the literature that can be used for further study and illustrate how to implement a Bayesian model on real data. All of the data and code are available for teaching purposes. This Primer discusses the general framework of Bayesian statistics and introduces a Bayesian research cycle (FIG. 1). We first discuss formalizing of prior distributions, prior predictive checking and determining the likelihood distribution (Experimentation). We discuss relevant algorithms and model fitting, describe examples of variable selection and variational inference, and provide an example calculation with posterior predictive checking (Results). Then, we describe how Bayesian statistics are being used in different fields of science (Applications), followed by guidelines for data sharing, reproducibility and reporting standards (Reproducibility and data deposition). We conclude with a discussion on avoiding bias introduced by using incorrect models (Limitations and optimizations), and provide a look into the future with Bayesian artificial intelligence (Outlook).

✉e-mail: a.g.j.vandeschoot@uu.nl

https://doi.org/10.1038/s43586-020-00001-2

NEnv: Neural Environment Maps for Global Illumination



NEnv: Neural Environment Maps for Global Illumination

NEnv: Neural Environment Maps for Global Illumination




NEnv: Neural Environment Maps for Global Illumination

- Carlos Rodríguez-Pardo
  - Postdoctoral researcher at Politecnico di Milano
  - Previously at SEDDI and URJC
  - Work in Machine Learning + Computer Vision + Graphics

NEnv: Neural Environment Maps for Global Illumination

- Francisco Javier Fabre
  - Research Engineer at SEDDI
  - Ongoing Industrial PhD at URJC
  - Work in Offline Rendering + Volumetric Rendering + Material Appearance.

NEnv: Neural Environment Maps for Global Illumination

- Elena Garces
  - Senior Researcher & Director at SEDDI
  - Juan de la Cierva Fellow at URJC
  - Previously at Adobe Research & Technicolor R&D
  - Applied Machine Learning in scene reconstruction and digitalization of fabric optics

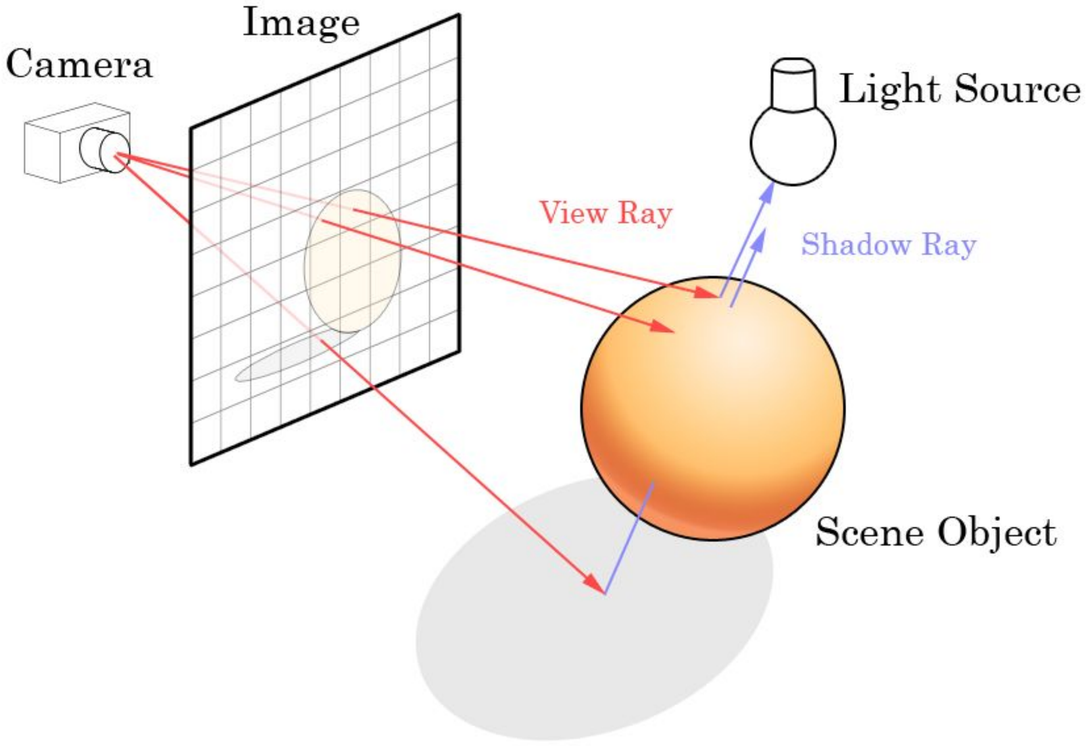
NEnv: Neural Environment Maps for Global Illumination

- Jorge López-Moreno
  - Chief Science Officer at SEDDI
  - Associate professor at URJC
  - Previously Adobe Research & Universidad de Zaragoza
  - Surface reconstruction, appearance models, offline and real–time rendering

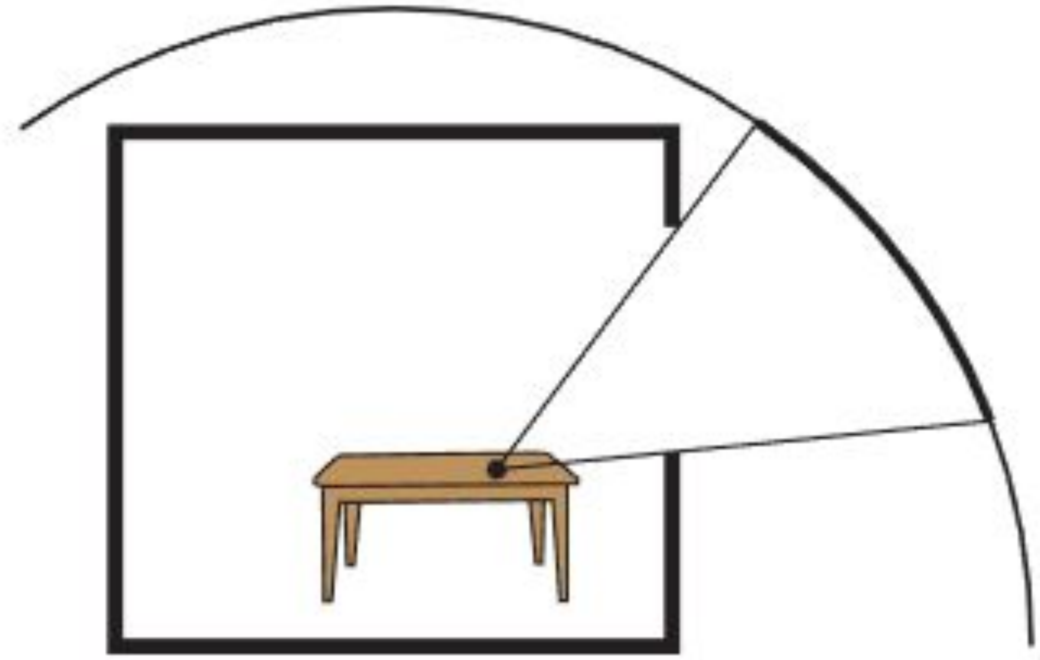
Normalizing flows for render

What is Rendering?

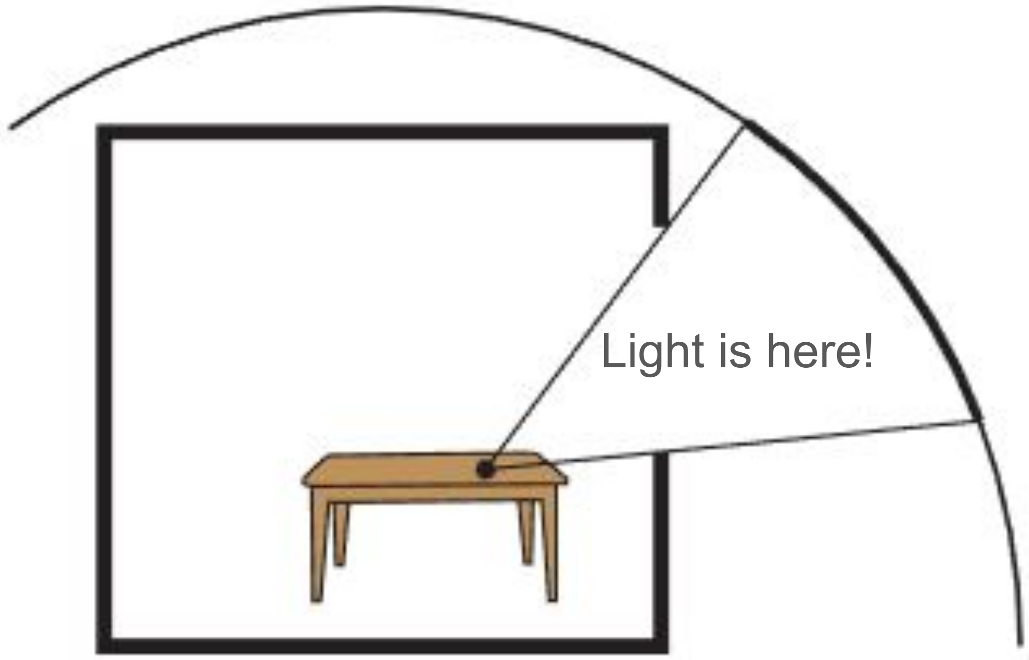
Normalizing flows for render



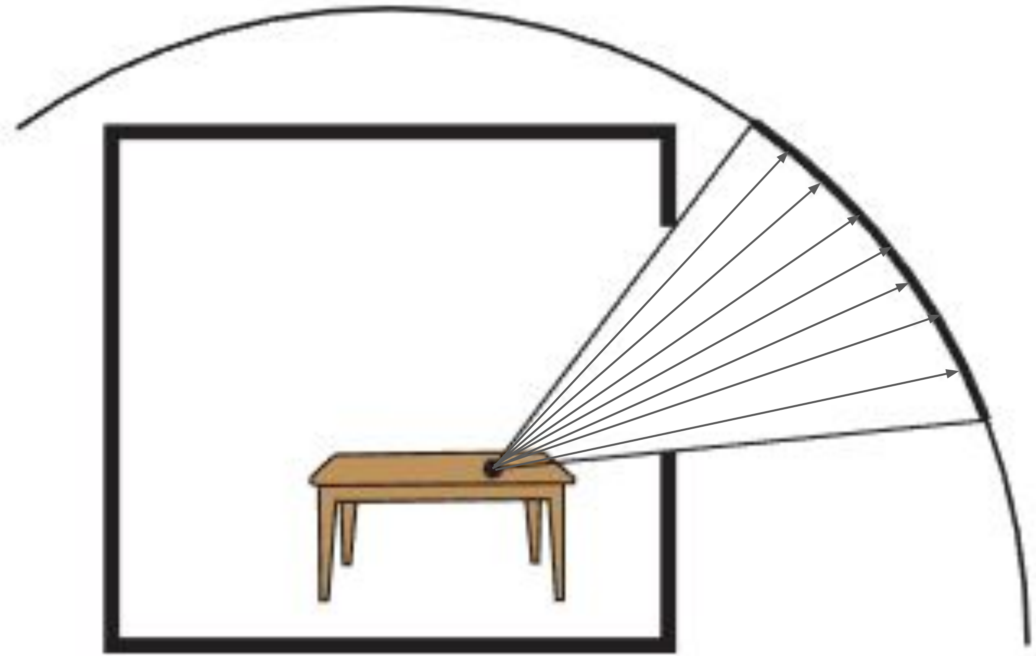
Normalizing flows for render



Normalizing flows for render



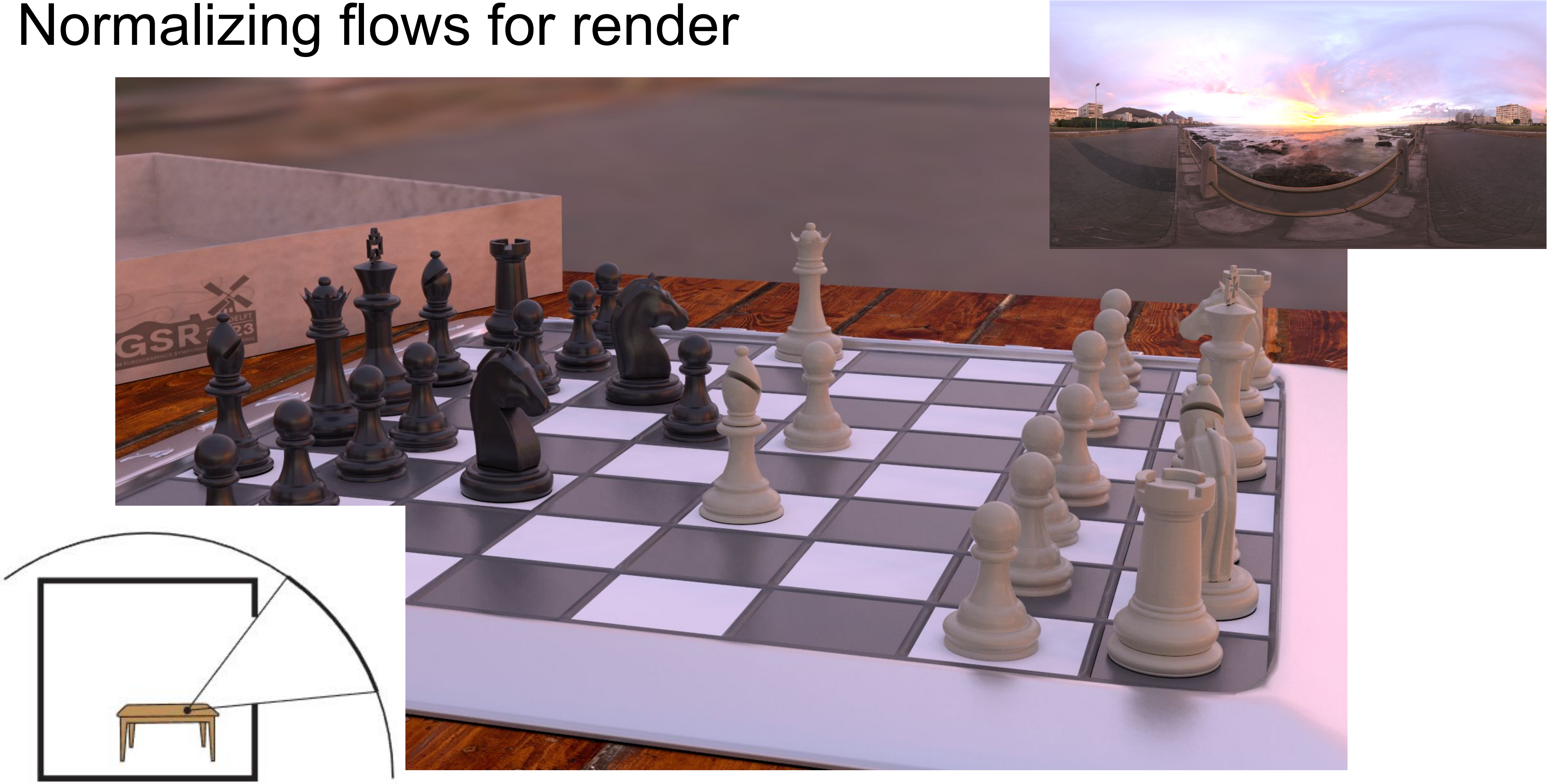
Normalizing flows for render



Normalizing flows for render

Normalizing flows for render

Normalizing flows for render



Normalizing flows for render

Eurographics Symposium on Rendering 2023
T. Ritschel and A. Weidlich
(Guest Editors)

NEnv: Neural Environment Maps for Global Illumination

Carlos Rodriguez-Pardo*[1,2] and Javier Fabre*[1,2] and Elena Garces[1,2] and Jorge Lopez-Moreno[1,2]

[1]SEDDI, Madrid, Spain
[2]Universidad Rey Juan Carlos, Madrid, Spain
* Denotes equal contribution

Figure 1: *We introduce NEnv, an invertible and fully differentiable neural method which achieves high-quality reconstructions for environment maps and their probability distributions. NEnv is up to two orders of magnitude faster to sample from than analytical alternatives, providing fast and accurate lighting representations for global illumination using Multiple Importance Sampling. Our models can accurately represent both indoor and outdoor illumination, achieving higher generality than previous work on environment map approximations.*

Abstract

Environment maps are commonly used to represent and compute far-field illumination in virtual scenes. However, they are expensive to evaluate and sample from, limiting their applicability to real-time rendering. Previous methods have focused on compression through spherical-domain approximations, or on learning priors for natural, day-light illumination. These hinder both accuracy and generality, and do not provide the probability information required for importance-sampling Monte Carlo integration. We propose NEnv, a deep-learning fully-differentiable method, capable of compressing and learning to sample from a single environment map. NEnv is composed of two different neural networks: A normalizing flow, able to map samples from uniform distributions to the probability density of the illumination, also providing their corresponding probabilities; and an implicit neural representation which compresses the environment map into an efficient differentiable function. The computation time of environment samples with NEnv is two orders of magnitude less than with traditional methods. NEnv makes no assumptions regarding the content (i.e. natural illumination), thus achieving higher generality than previous learning-based approaches. We share our implementation and a diverse dataset of trained neural environment maps, which can be easily integrated into existing rendering engines.

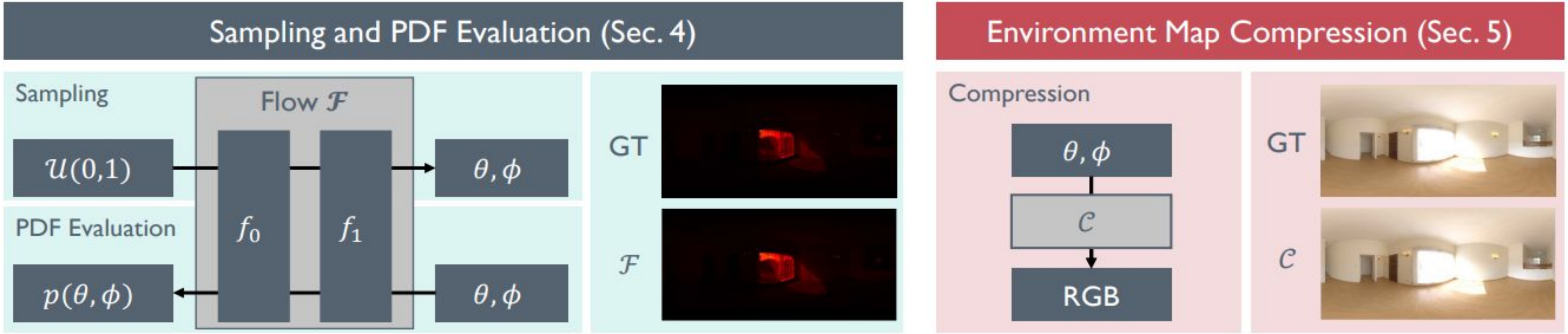
CCS Concepts
• ***Computing methodologies*** → ***Neural networks;*** *Image-based rendering; Image representations;*

NEnv: Neural Environment Maps for Global Illumination

Carlos Rodriguez-Pardo, Javier Fabre*, Elena Garces , Jorge Lopez-Moreno*

- Normalizing flows to sample environment maps
- Compression network to encode RGB
- Implemented in a production path–tracer

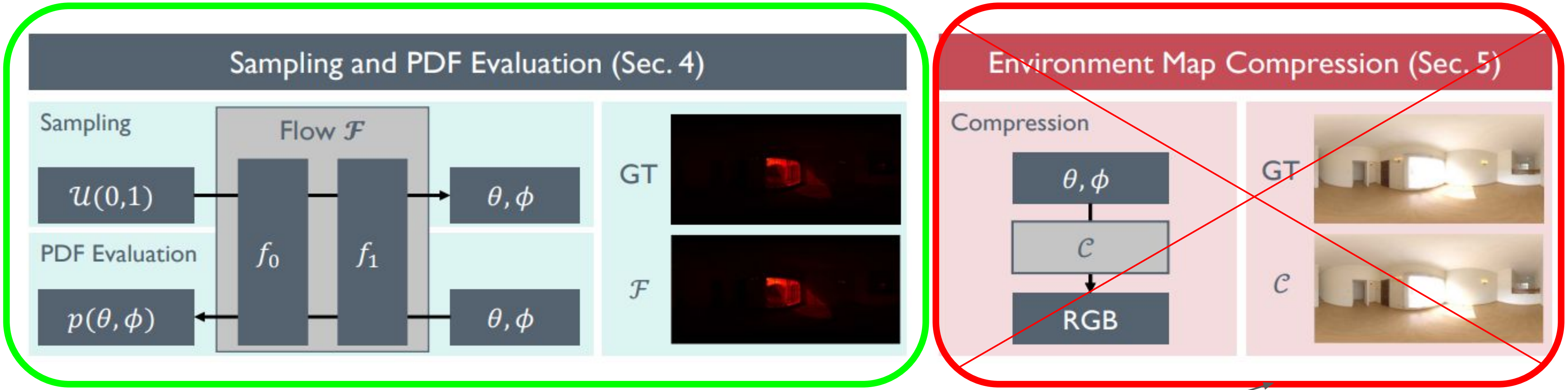
Normalizing flows for render



Normalizing flows for render

This is a normalizing flow

Normalizing flows for render



This is a normalizing flow

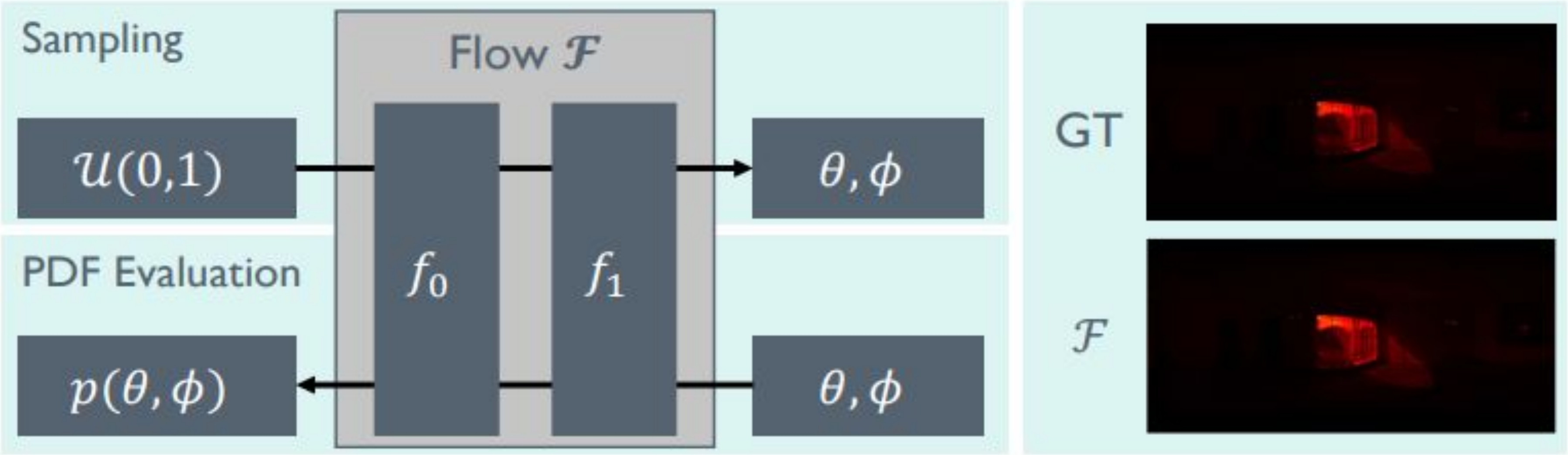
This is NOT a normalizing flow

Normalizing flows for render

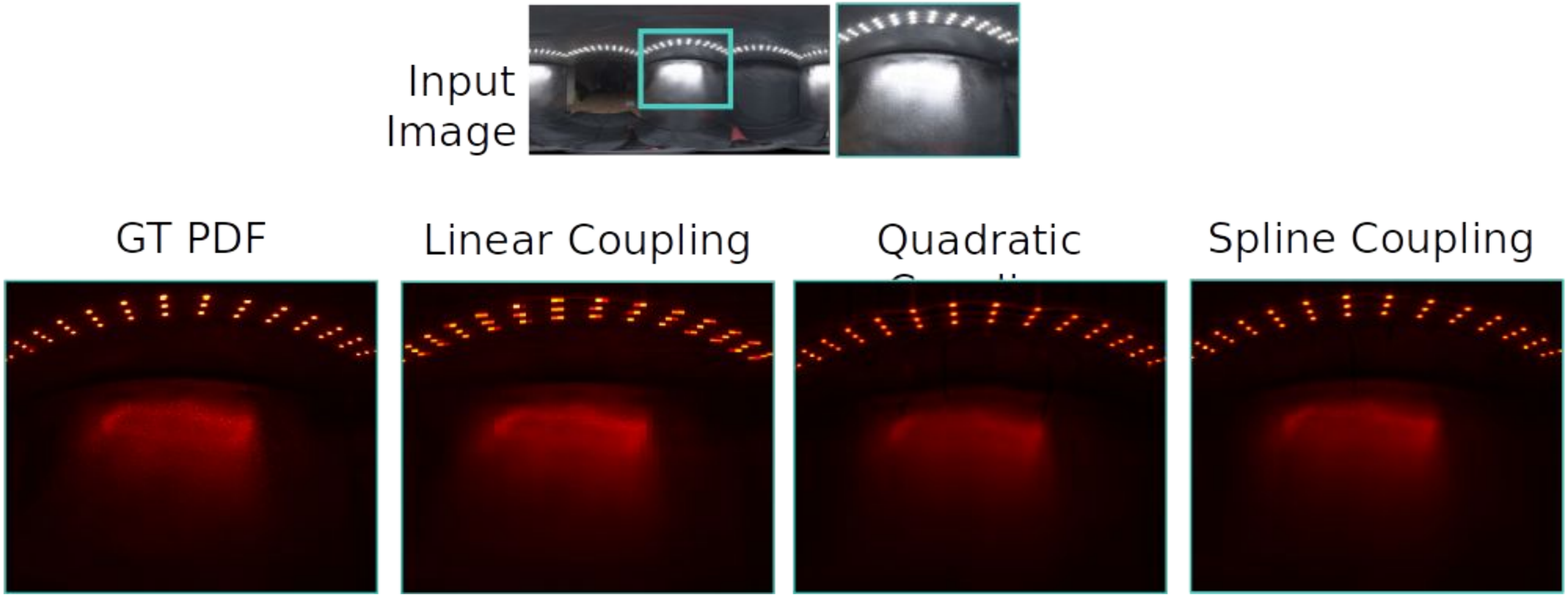
Input to NEnv: A single HDRi map

Training time: 2 Hours per image **(Nvidia RTX 3060 GPU)**

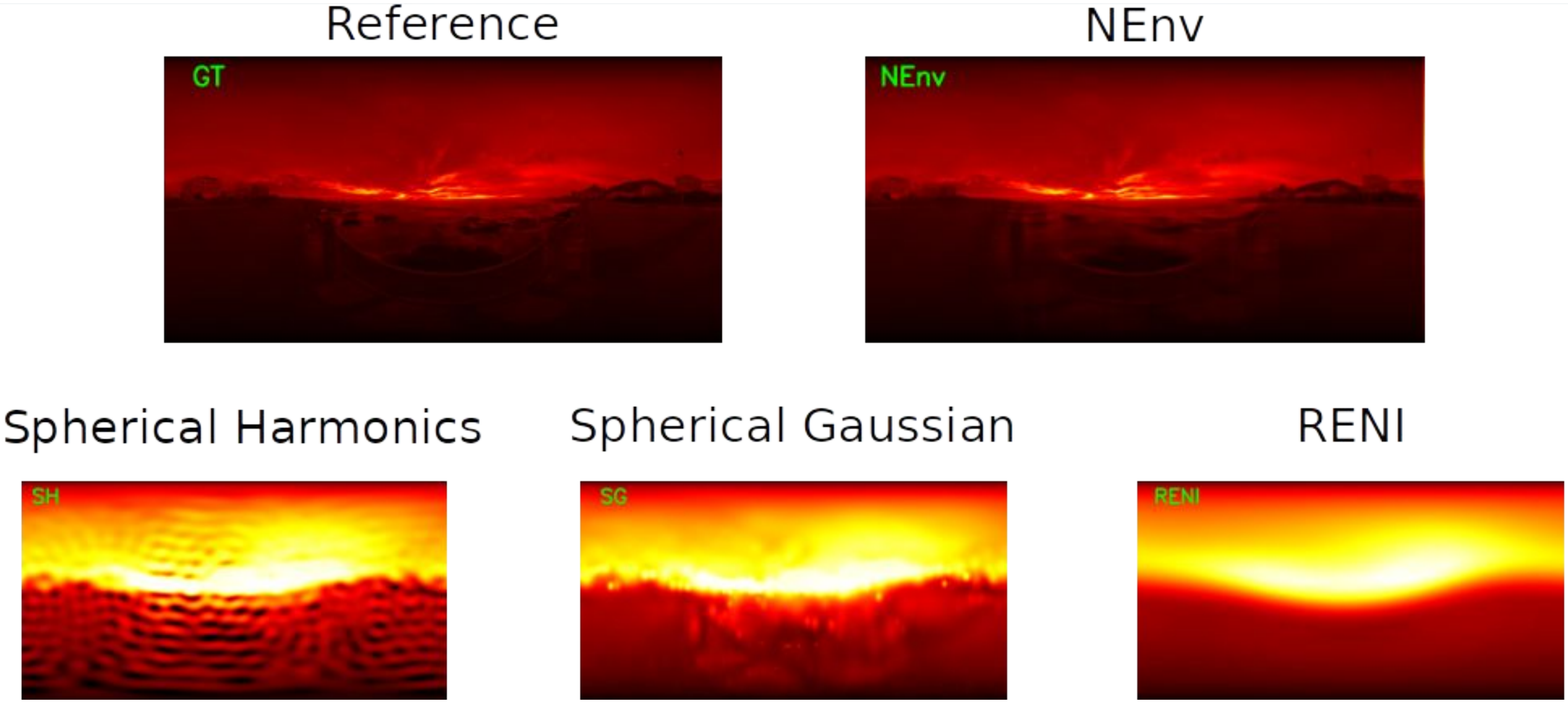
Normalizing flows for render



Normalizing flows for render

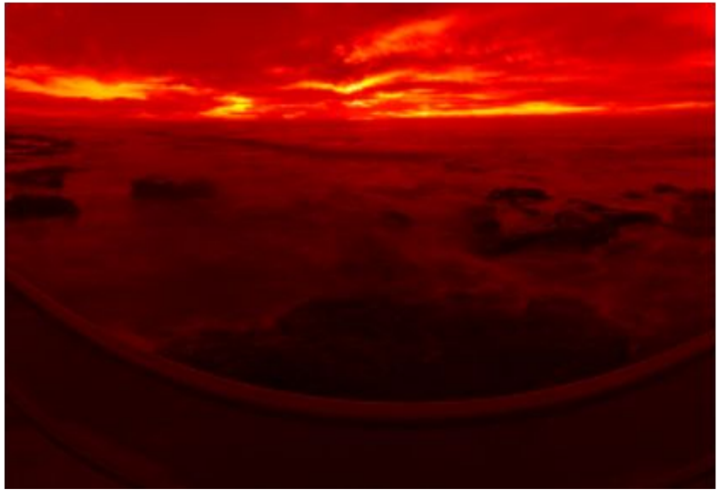


Normalizing flows for render

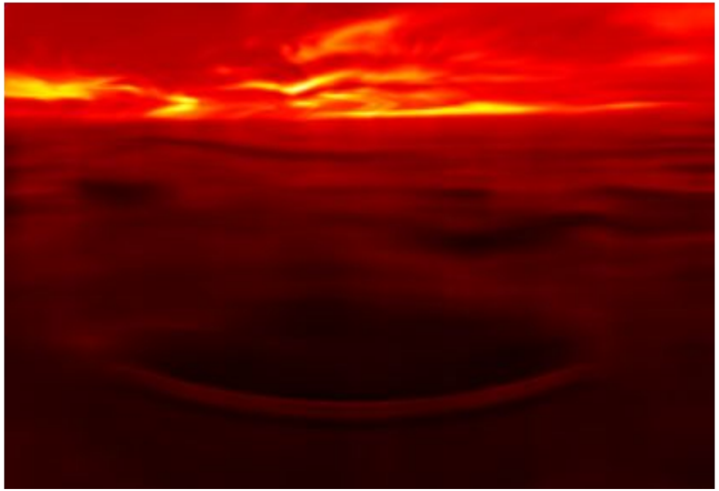


Normalizing flows for render

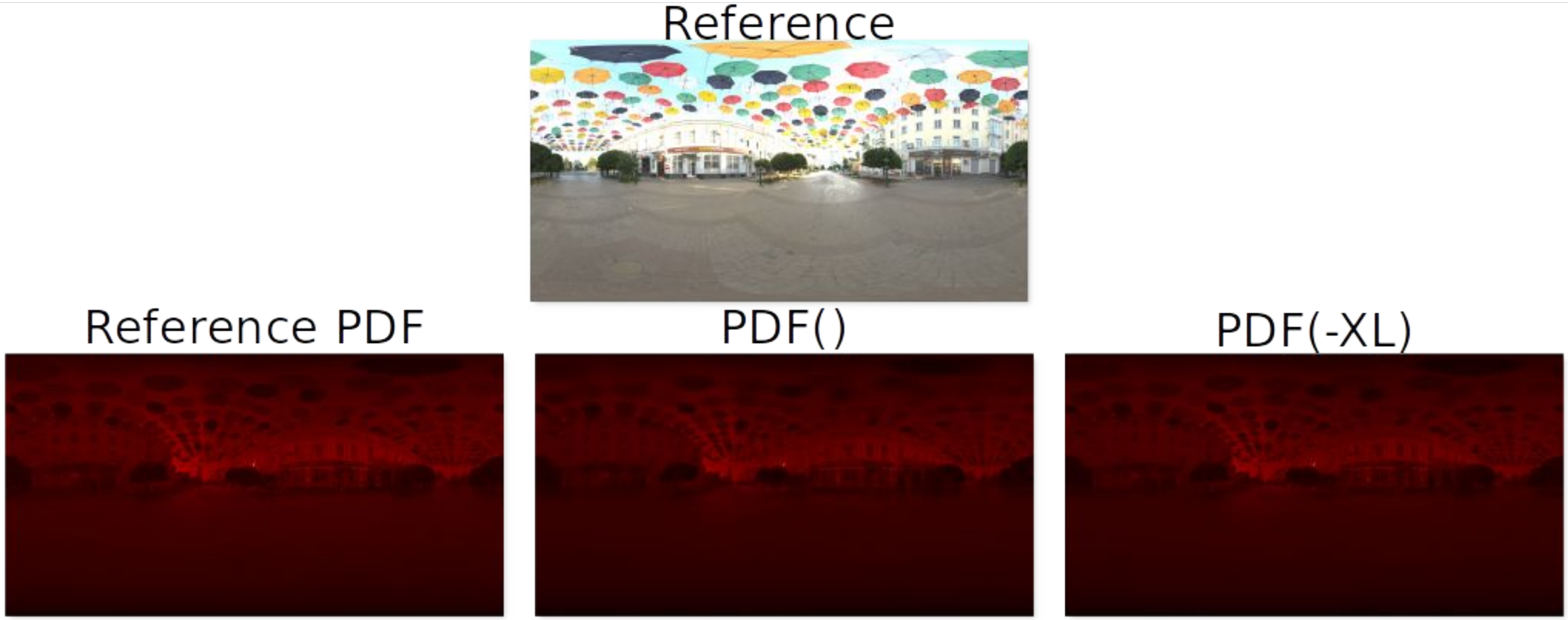
Reference



NEnv

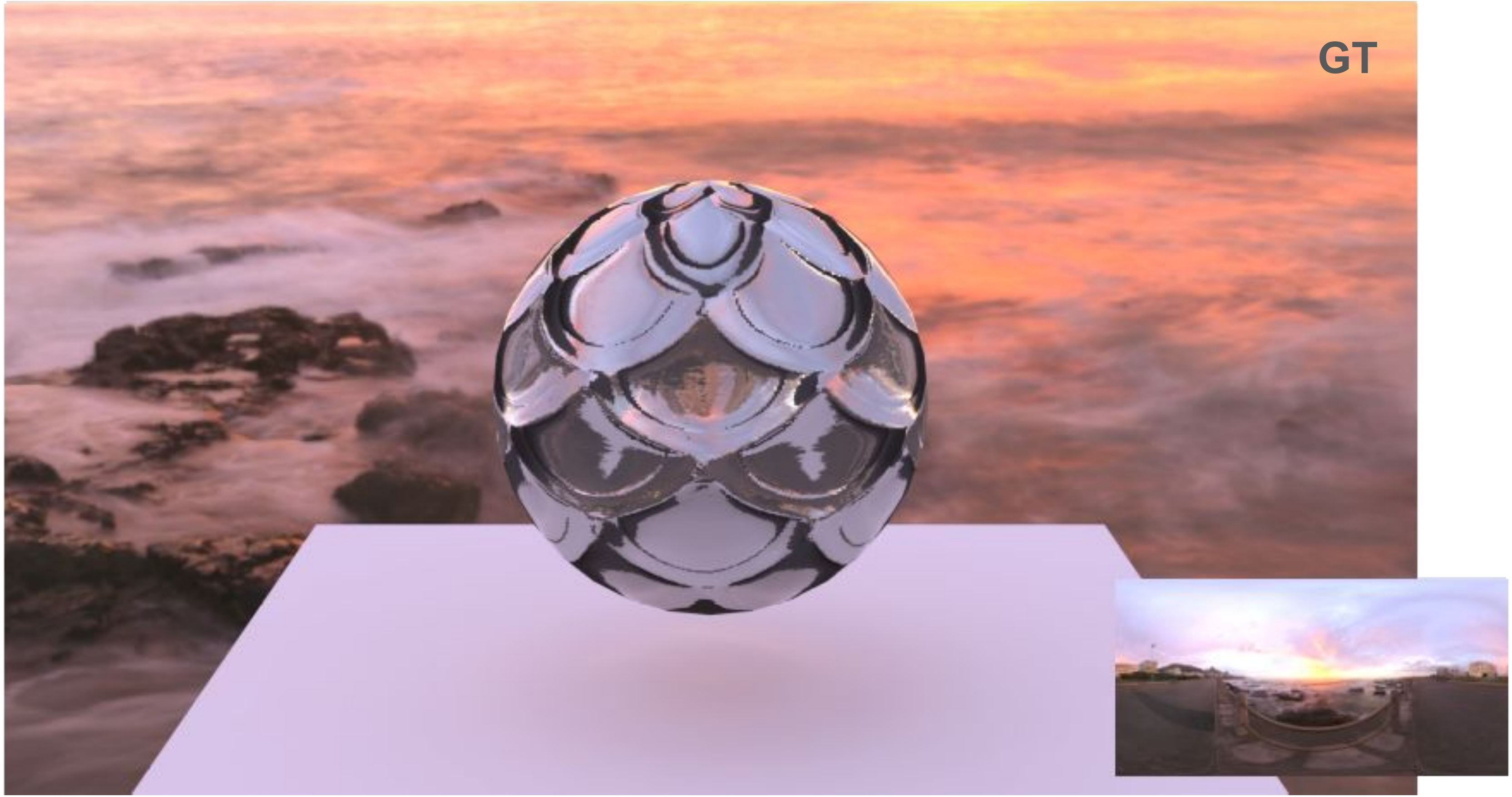


Normalizing flows for render

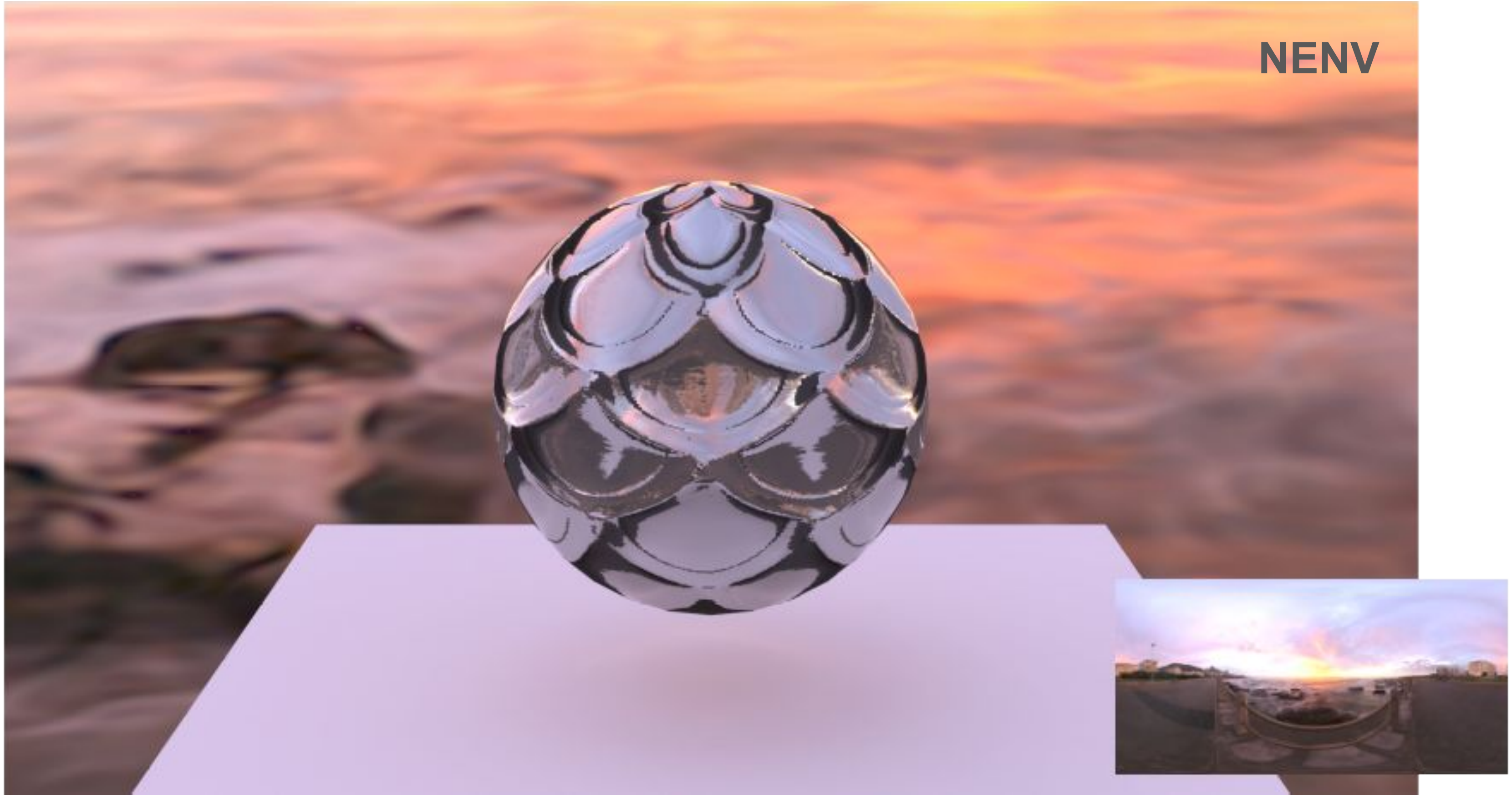


Normalizing flows for render

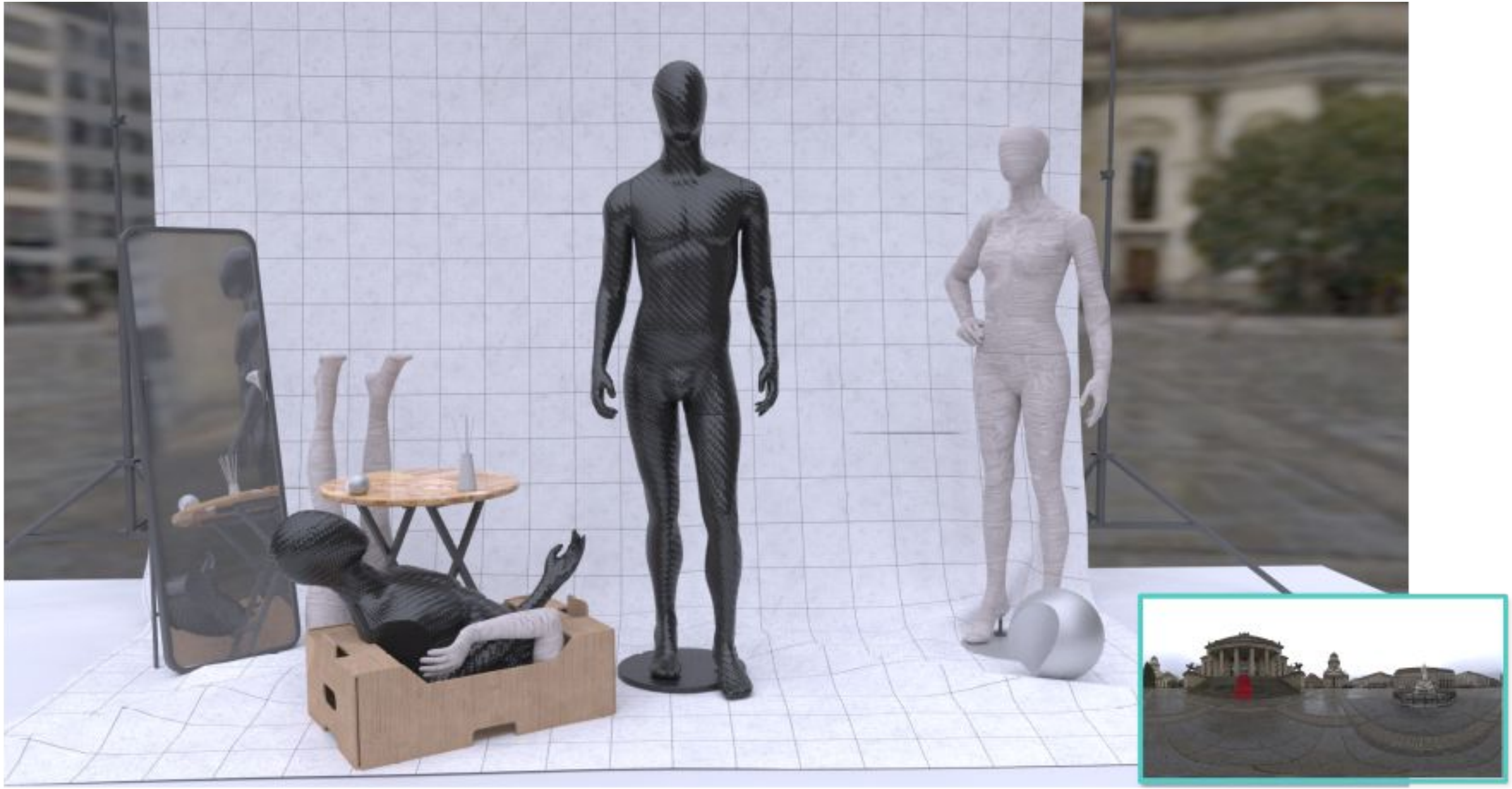
Normalizing flows for render



Normalizing flows for render



Normalizing flows for render



Normalizing flows for render

References

- Neural Spline Flows
- What are Normalizing Flows?
- CS480/680 Lecture 23: Normalizing flows (Priyank Jaini)
- Introduction to Normalizing Flows (ECCV2020 Tutorial)
- PBRT4: Infinite Area Lights
- Ray Tracing in One Weekend
- NEnv: Neural Environment Maps for Global Illumination