MAX PLANCK INSTITUTE
FOR INTELLIGENT SYSTEMS

Neural Posterior Estimation for Gravitational Waves

An Introduction

Annalena Kofler, 17.04.2024

Posterior Estimation

Why care about the Posterior?

Bayes Formula:

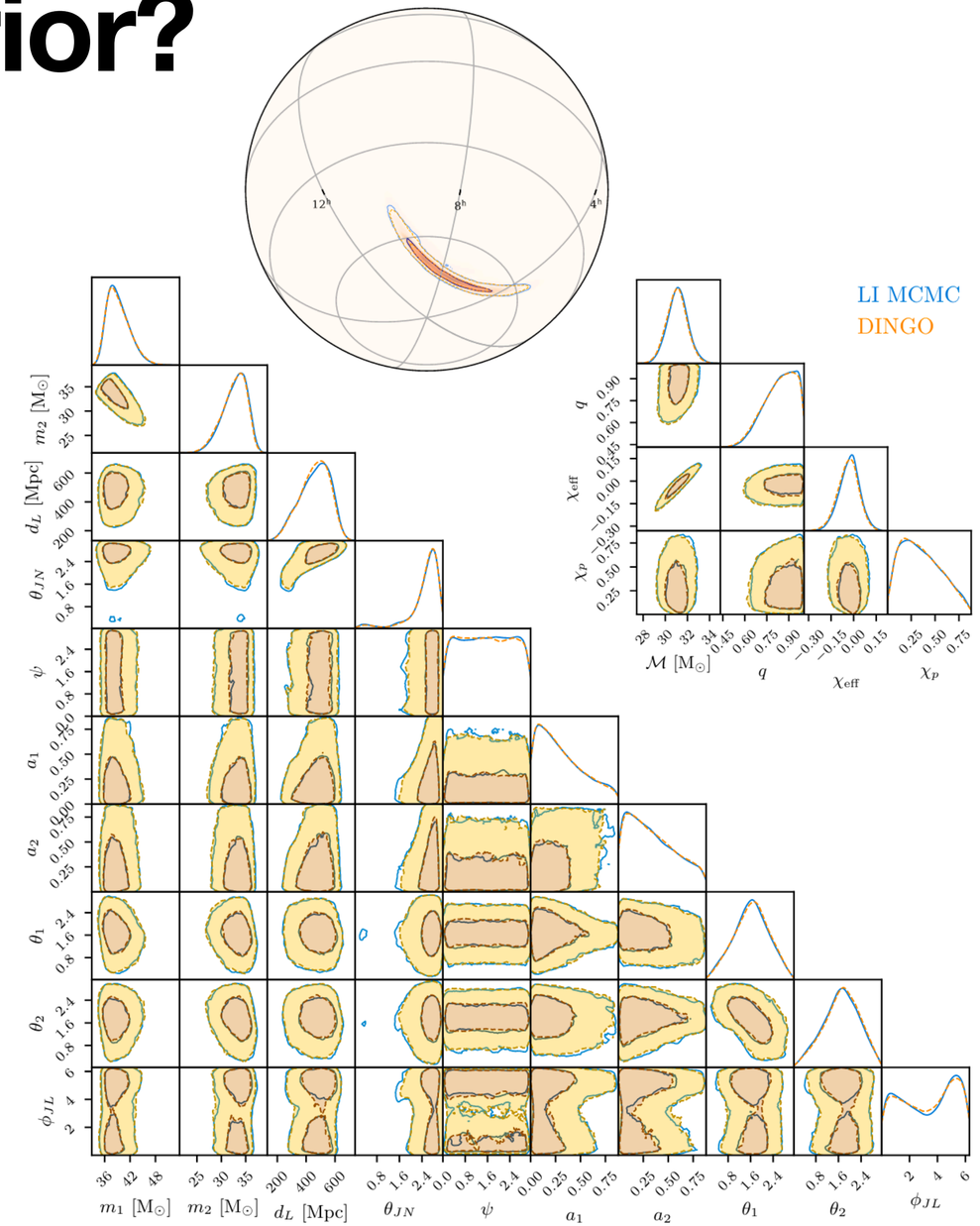
$$p(\theta \mid x) = \frac{p(x \mid \theta)}{p(x)}\, p(\theta)$$

Likelihood — Posterior — Prior — Evidence

x: Data

θ: Parameters

→ allow us to interpret gravitational wave signals



Dax, et al.,Real-time gravitational-wave science with neural posterior estimation, PRL2021

How does Posterior Estimation usually work?

Bayes Formula:

$$p(\theta \mid x) = \frac{p(x \mid \theta)}{p(x)}\, p(\theta)$$

Likelihood

Posterior

Prior

Evidence

x: Data

θ: Parameters

Steps:

1. Specify Prior, e.g. $d_l \in [100,\ 1000]$ Mpc
2. Run Monte Carlo Method, e.g. Nested Sampling
   - Simulate waveforms
   - Calculate likelihood
3. Obtain converged posterior

Why is this problematic in the future?

- Standard Bayesian methods: millons - billions of parameter estimations before converging
- Evaluating one event: Minutes - Weeks

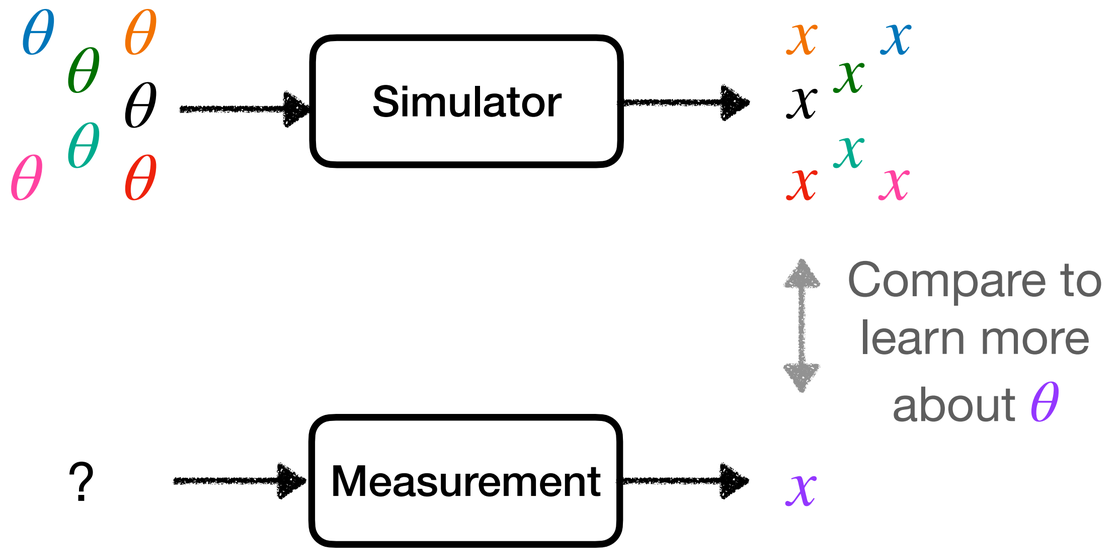
- In the future: detector upgrades → more signals to analyze

→ Solve problem by using alternative methods such as simulation-based inference

Simulation-Based Inference and Neural Posterior Estimation

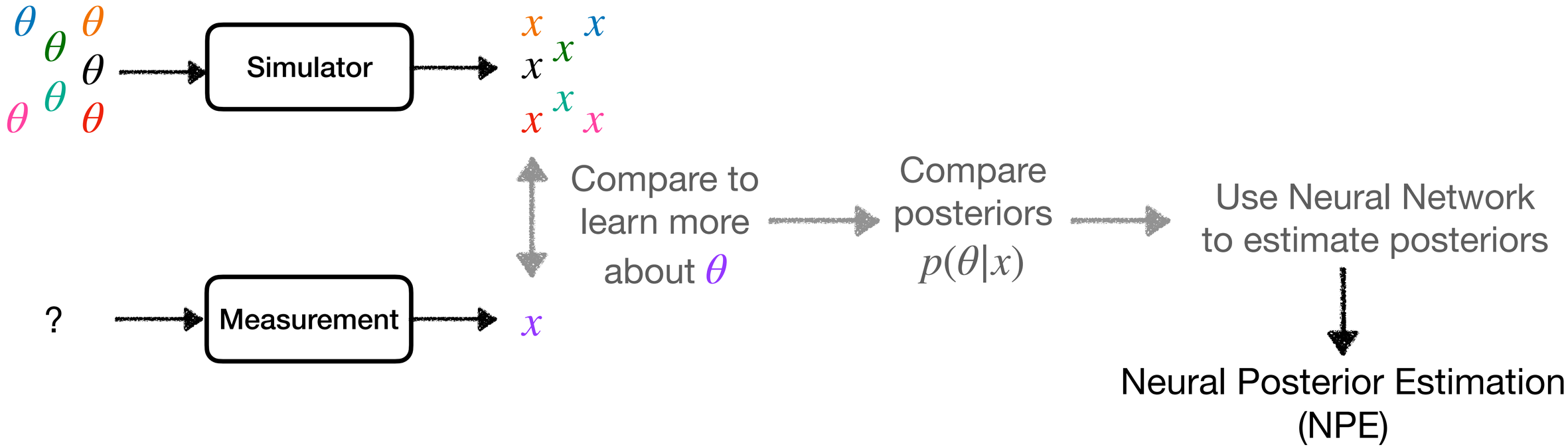
What is Simulation-Based Inference?

- Statistical Method to obtain information about a distribution based on simulated samples
- Requirement: Simulator



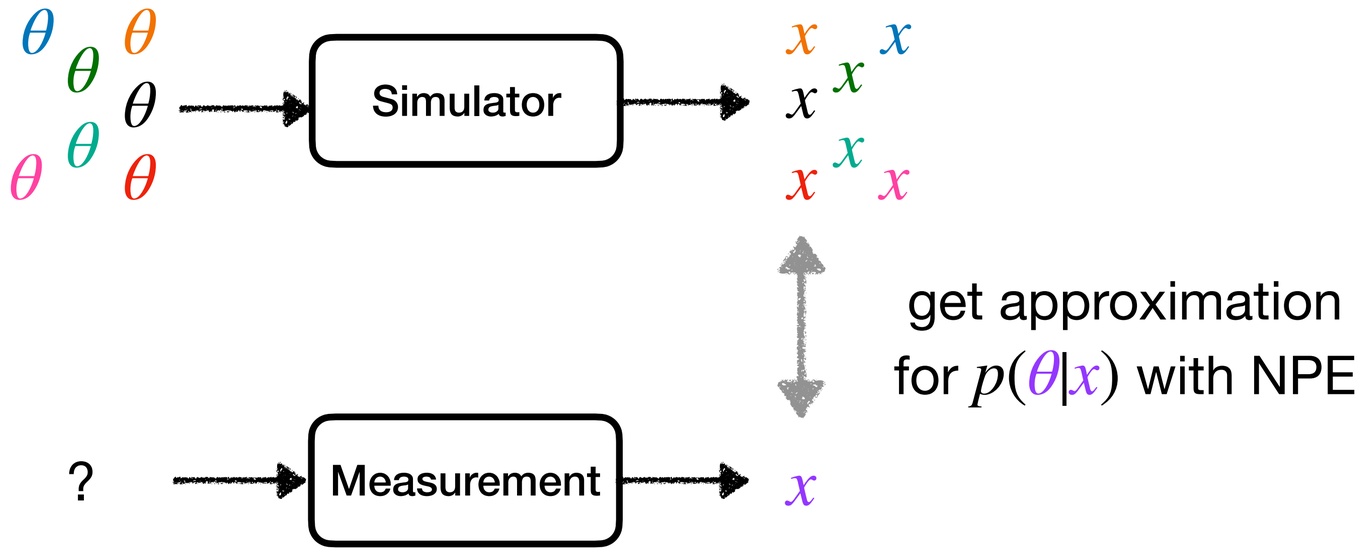
How can we do the comparison?

- Use posteriors



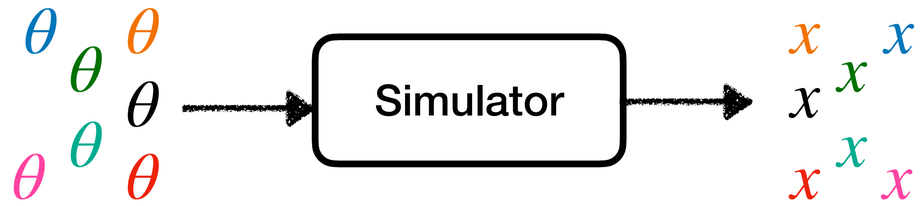
How can we do the comparison?

- Use **Neural Posterior Estimation (NPE)**



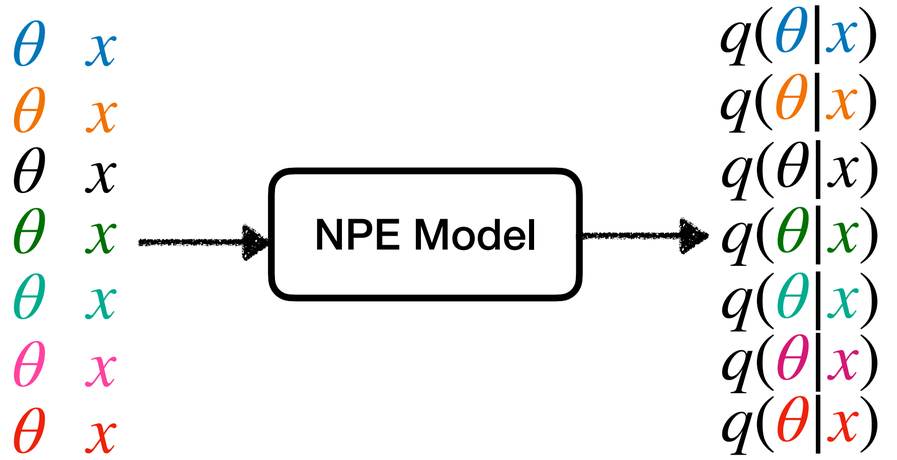
How does Neural Posterior Estimation work?

1. Generate large training data set with Simulator

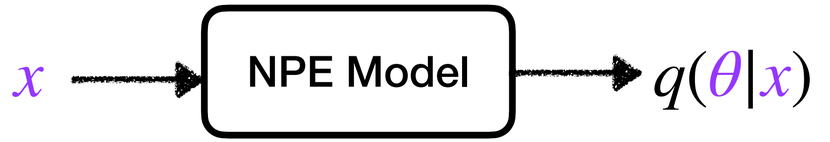


2. Train Machine Learning Model q to approximate posterior

$$p(\theta|x) = \frac{p(x|\theta)}{p(x)} p(\theta) \approx q(\theta|x)$$



3. Evaluate model on measurement to obtain $q(\theta|x)$



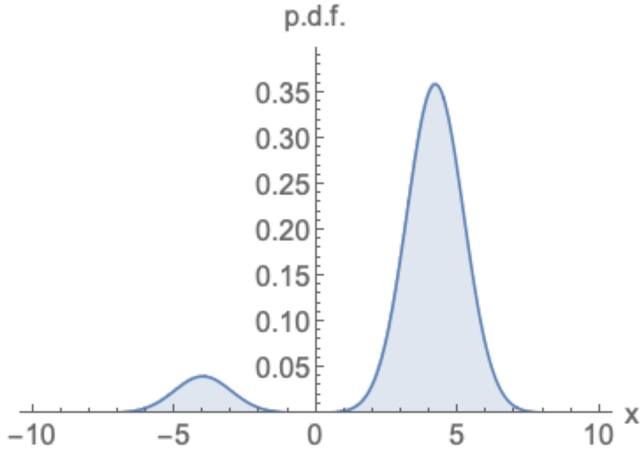
What is this mysterious ML Model q?

Can we use any Deep Neural Network to model a distribution?

→ No, it would not fulfill the properties of a probability density!

Properties:

1. Positive: $p(x) \geq 0 \forall x$
2. Normalized: $\int p(x)\ \mathrm{d}x = 1$

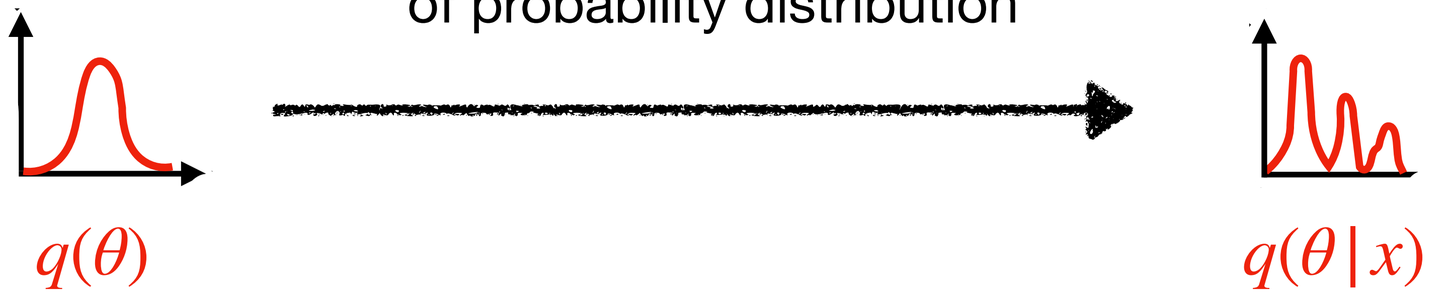


What is this mysterious ML Model q?

Model needs to fulfill these properties!

Idea:

Transformation that keeps properties of probability distribution



$q(\theta)$ $q(\theta \mid x)$

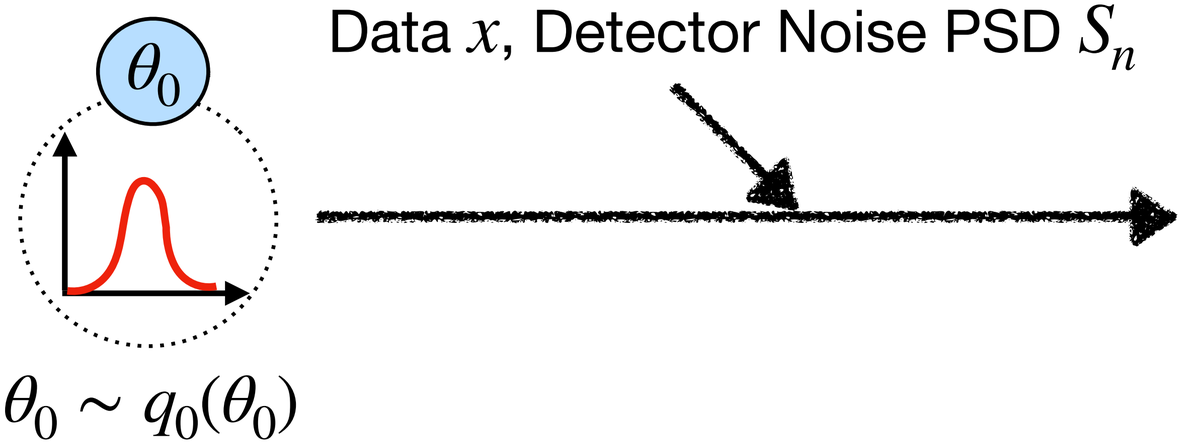
→ Normalizing Flow

Normalizing Flow

How to construct a Normalizing Flow?

Conditional Normalizing Flow

Prior
$q_0(\theta_0)$

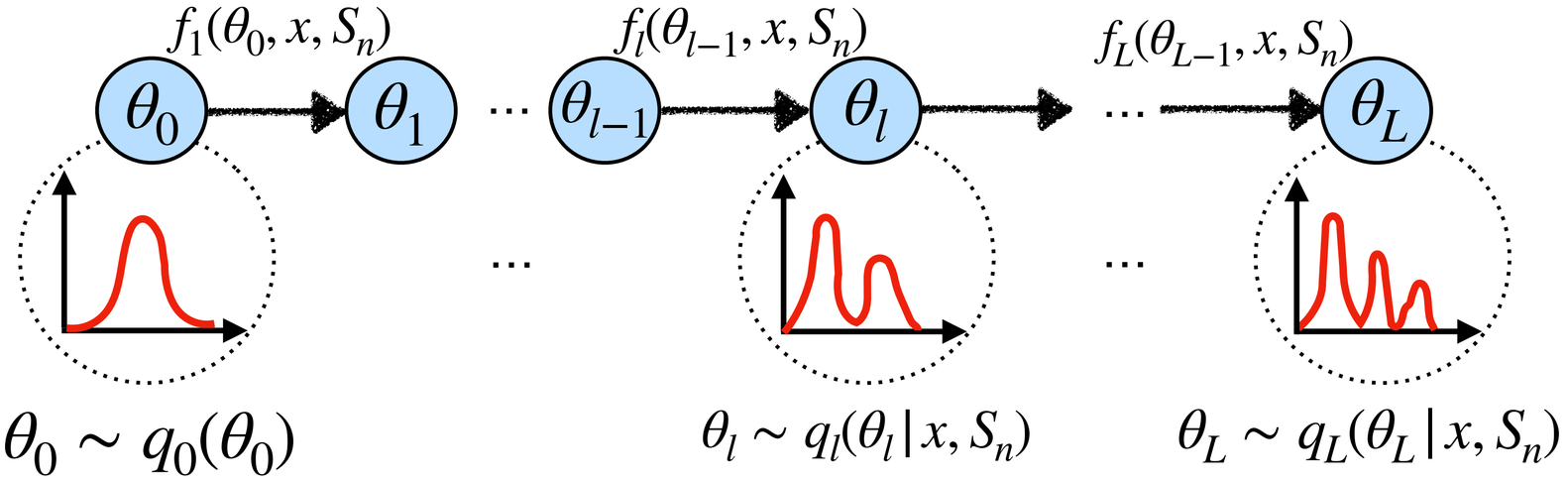


$\theta_0 \sim q_0(\theta_0)$

How to construct a Normalizing Flow?

Conditional Normalizing Flow = Sequence of invertible transformations with condition

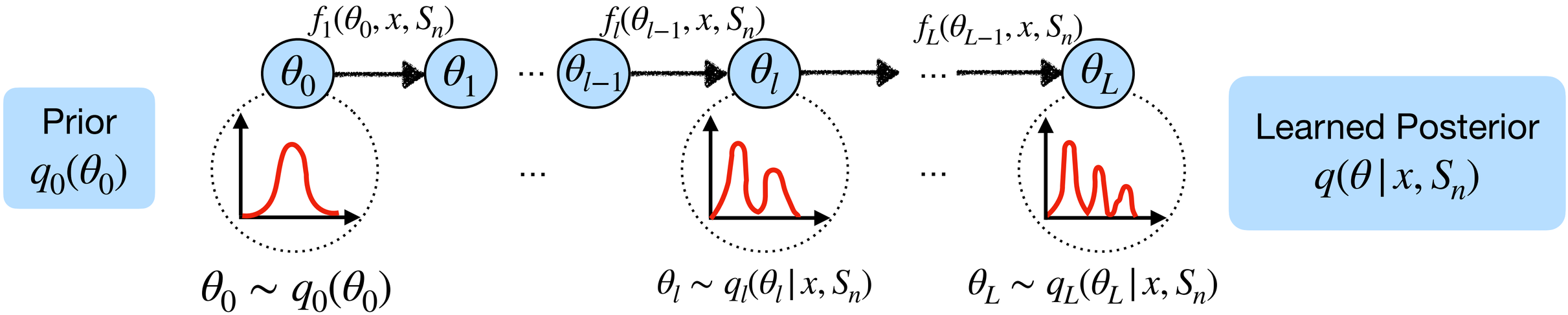
Prior
$q_0(\theta_0)$



Learned Posterior
$q(\theta \mid x, S_n)$

How to sample from a Normalizing Flow?

Conditional Normalizing Flow = Sequence of invertible transformations with condition



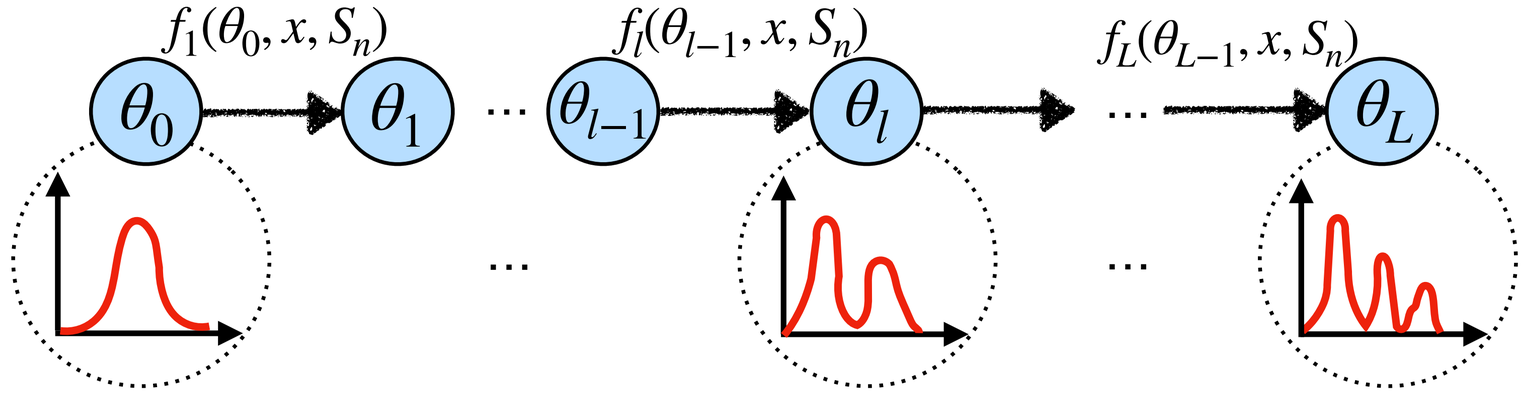
Data x, Detector Noise PSD S_n

Sampling: $\theta_0 \sim q_0(\theta_0)$ → $\theta_L = f_L \circ \ldots \circ f_1(\theta_0, x, S_n)$

How to evaluate the density?

Conditional Normalizing Flow = Sequence of invertible transformations with condition

Prior
$q_0(\theta_0)$



Learned Posterior
$q(\theta \mid x, S_n)$

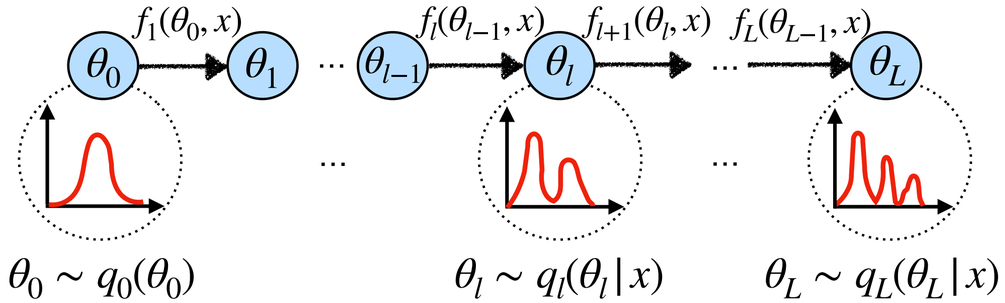
Evaluating the density:

$$\theta_0 = f_1^{-1} \circ \ldots \circ f_L^{-1}(\theta, x, S_n) \quad \xleftarrow{x, S_n} \quad \theta = f_L(\theta_{L-1}, x, S_n)$$

$$q(\theta \mid x, S_n) = q_0\left(f^{-1}(\theta, x, S_n)\right) \left| \det \left(\frac{\partial f^{-1}(\theta, x, S_n)}{\partial \theta} \right) \right| = q_0(f^{-1}(\theta, x, S_n)) \left| \det J_{f^{-1}} \right|$$

Example: Gaussian with two modes

Prior
$q_0(\theta_0)$

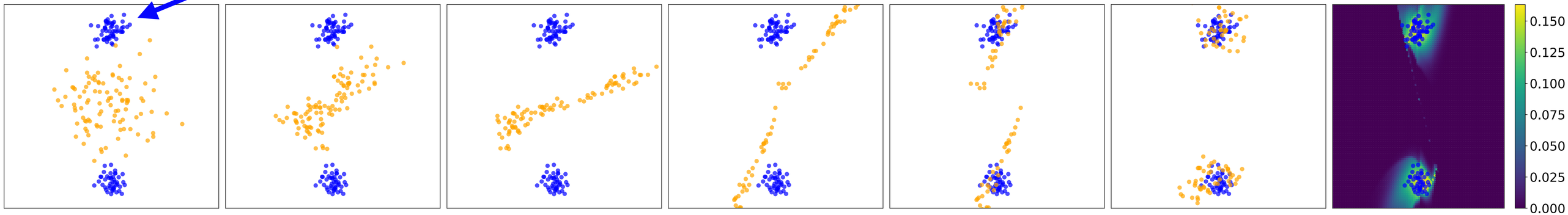


Learned Posterior
$q(\theta \mid x)$

Example:

Samples from target $p(\theta \mid x, S_n)$

$q(\theta \mid x, S_n)$
$= q_5(\theta_5 \mid x, S_n)$



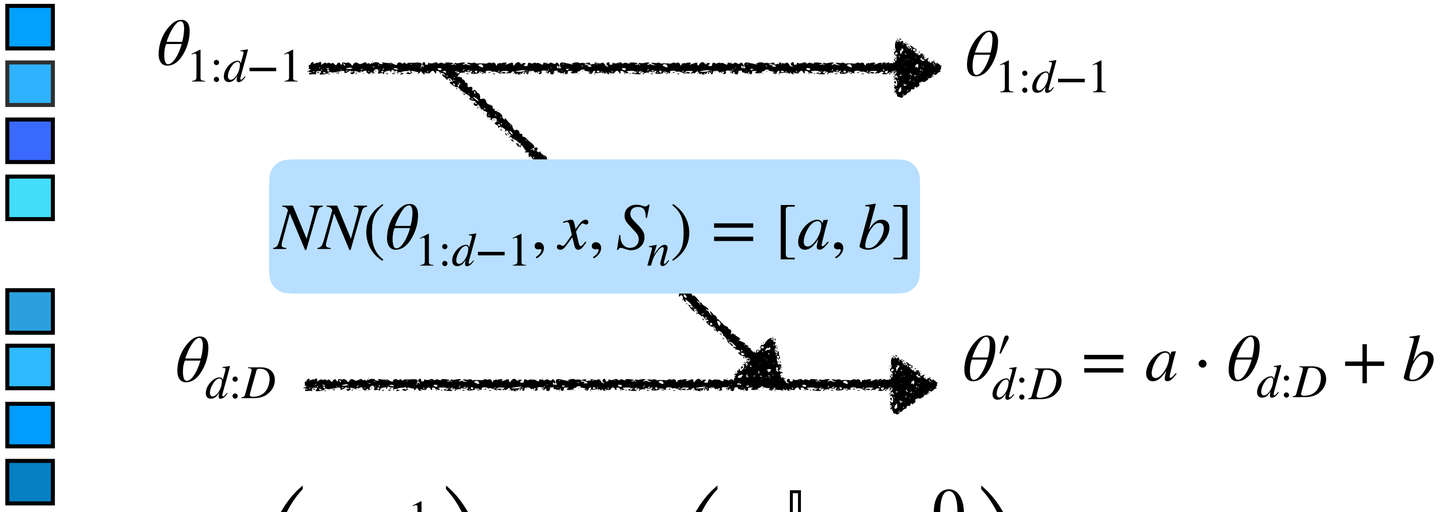
$\theta_0 \sim q_0(\theta_0)$

$\theta_5 \sim q_5(\theta_5 \mid x, S_n)$

How to construct the transformations?

How to construct **invertible** transformations with **tractable** Jacobian:

- Linear transformation: $\theta_i = a \cdot \theta_{i-1} + b \rightarrow$ learn a and b by NN
- Coupling layers: split $\vec{\theta} = (\theta_{1:d-1}, \theta_{d:D}) \in \mathbb{R}^D$

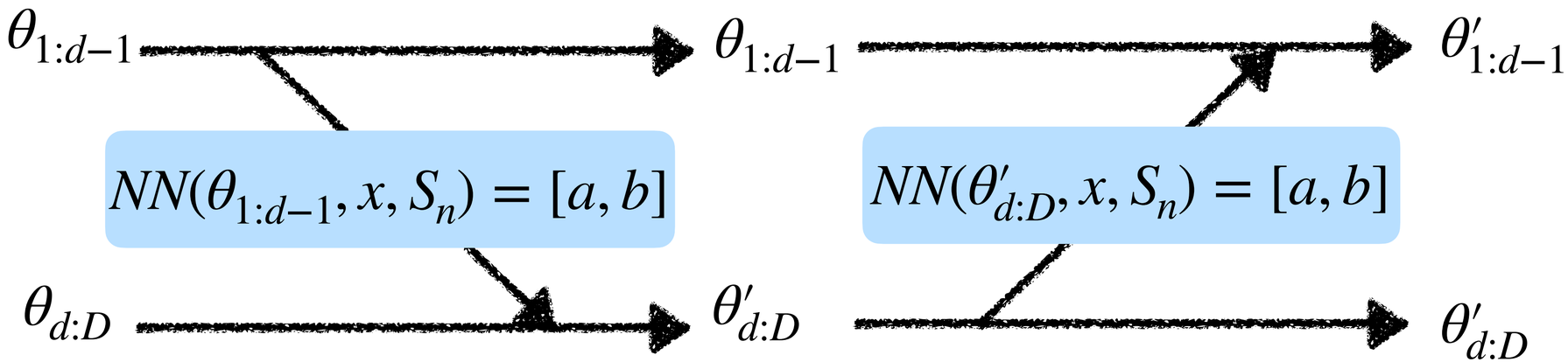


$$\det\left(\frac{\partial f^{-1}}{\partial \theta}\right) = \det\begin{pmatrix} \mathbb{I} & 0 \\ \frac{\partial \theta'_{d:D}}{\partial \theta_{1:d-1}} & \mathbb{I}a \end{pmatrix} = a^{D-d}$$

How to construct the transformations?

How to construct **invertible** transformations with **tractable** Jacobian:

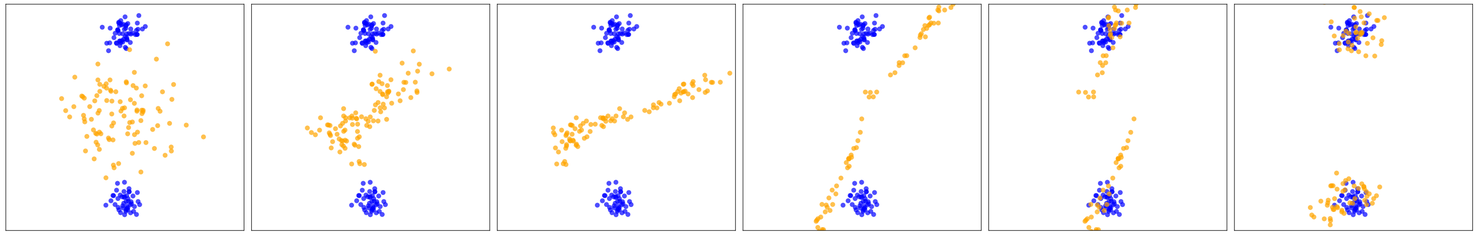
- Linear transformation: $\theta_i = a \cdot \theta_{i-1} + b \rightarrow$ learn a and b by NN
- Coupling layers: split $\vec{\theta} = (\theta_{1:d-1}, \theta_{d:D}) \in \mathbb{R}^D$

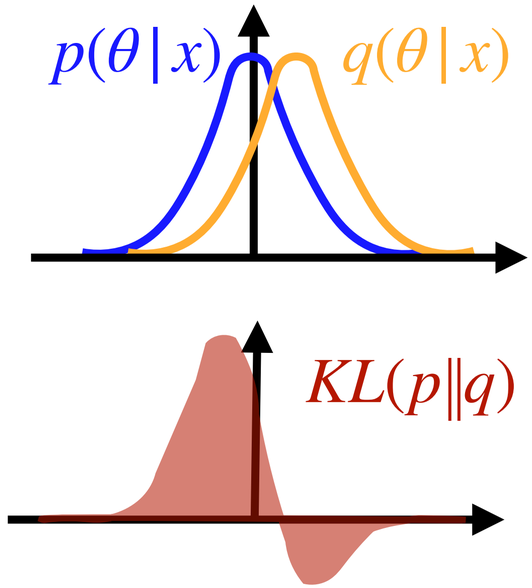


$$\det\left(\frac{\partial f^{-1}}{\partial \theta}\right) = \det\begin{pmatrix} \mathbb{I} & 0 \\ \frac{\partial \theta'_{d:D}}{\partial \theta_{1:d-1}} & \mathbb{I}a \end{pmatrix} = a^{D-d}$$

How to train a normalizing flow?

How can we compare two distributions?





Kullback-Leibler divergence

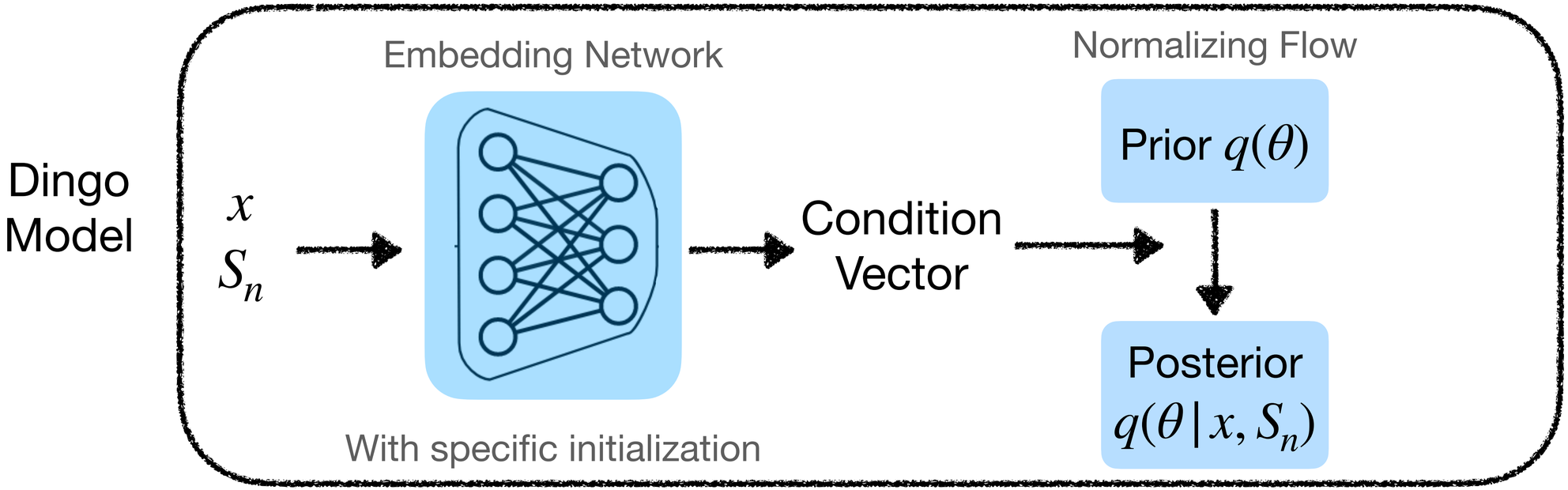
$$\mathscr{L} = \mathrm{KL}(p \,\|\, q_\theta) = \mathbb{E}_{x\sim p(x\mid\theta,S_n),S_n\sim p(S_n),\theta\sim p(\theta)}\left[\log \frac{p(\theta \mid x, S_n)}{q(\theta \mid x, S_n)}\right]$$

$$= -\sum_{i=1}^{N} \log q(\theta \mid x_i, S_{n,i}) + \text{const.}$$

→ minimize negative log-likelihood by evaluating the density of training samples

How is the condition included?

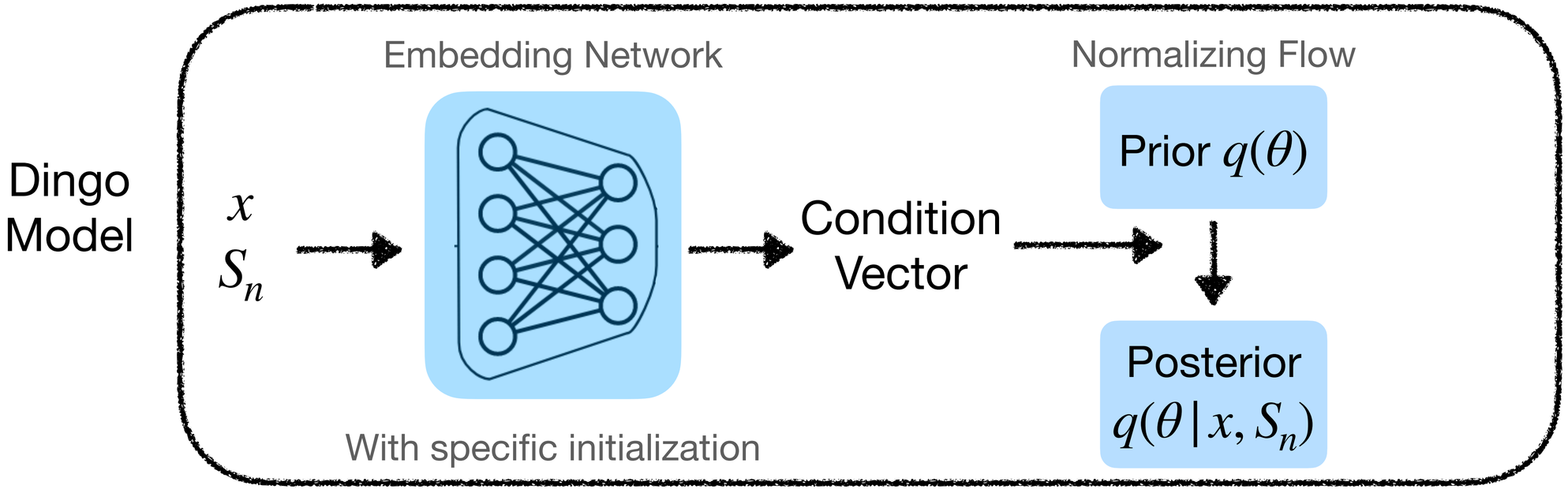
- Condition: Data x, Detector Noise PSD S_n (potentially of multiple detectors)
- Challenge: high-dimensional matrix, e.g. matrix of $[8000,\ 3,\ 3]$
  → too large as condition
  → compress to vector of length 128



DINGO

What is DINGO?

- Dingo = Deep INference for Gravitational wave Observations



Success of DINGO

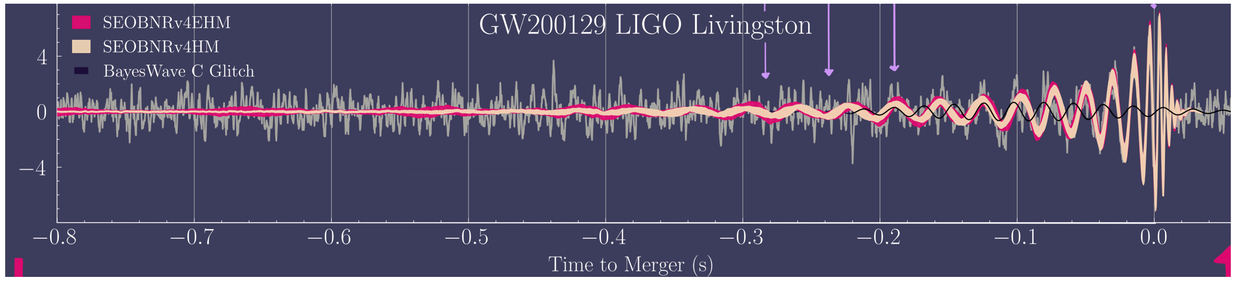
List of papers:

- Dax, et al., Real-Time Gravitational Wave Science with Neural Posterior Estimation. PRL 127, 2021
- Dax et al., Group Equivariant Neural Posterior Estimation, ICLR 2022
- Dax et al., Neural Importance Sampling for Rapid and Reliable Gravitational Wave Inference, PRL 130, 2023
- Wildberger et al., Adapting to noise distribution shifts in flow-based gravitational-wave inference, PRD 107, 2023
- Dax et al., Flow Matching for Scalable Simulation-Based Inference, NeurIPS 2023
- Gupte et al., In preparation, 2024

Success of DINGO

List of papers:

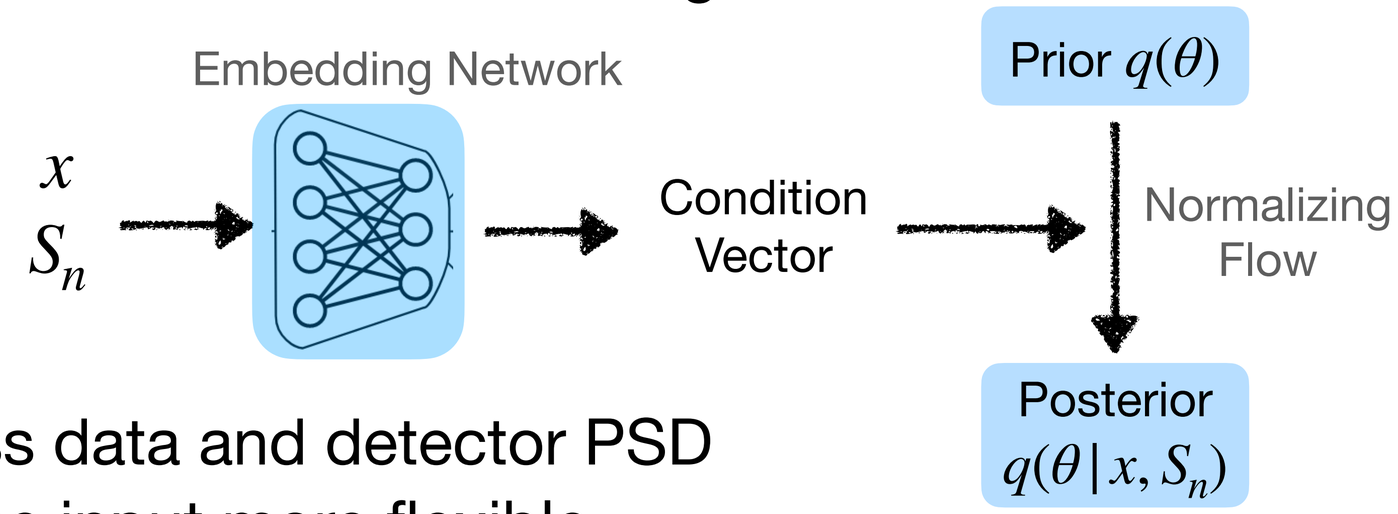
- Dax, et al., Real-Time Gravitational Wave Science with Neural Posterior Estimation. PRL 127, 2021
- Dax et al., Group Equivariant Neural Posterior Estimation, ICLR 2022
- Dax et al., Neural Importance Sampling for Rapid and Reliable Gravitational Wave Inference, PRL 130, 2023
- Wildberger et al., Adapting to noise distribution shifts in flow-based gravitational-wave inference, PRD 107, 2023
- Dax et al., Flow Matching for Scalable Simulation-Based Inference, NeurIPS 2023
- **Gupte et al., In preparation, 2024**



Gupte et al. In preparation, 2024

DINGO: 3 Take-Aways

1. DINGO = embedding network + conditional normalizing flow



2. Embedding network: Compress data and detector PSD
   → Goal of my work: Making the input more flexible

3. Normalizing Flow: Model posterior distribution

The Dingo Team

Max Dax

Stephen Green

Jonas Wildberger

Nihar Gupte

Michael Pürrer

Alex Roussopoulos

Samuel Clyne

Annalena Kofler

Jonathan Gair

Jakob Macke

Bernhard Schölkopf

Alessandra Buonanno

Thank you!

Do you have any questions?

Path to folders with plots

- PhenomPv2 plots:
  - Transformer Embedding:
    04_grid_search_resnet8/continuous_freq_enc/003
  - Standard Embedding with SVD init:
    04_grid_search_resnet8/fc_emb_net_svd_init
- PhenomXPHM plots:
  - Transformer Embedding:
    05_different_wfd_asd/discrete_freq_enc/003
  - Standard Embedding with SVD init:
    05_different_wfd_asd/fc_emb_net_svd_init