

PolySwyft

a sequential simulation-based nested sampler

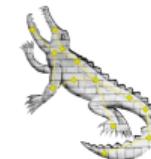
Will Handley
wh260@cam.ac.uk

Royal Society University Research Fellow
Astrophysics Group, Cavendish Laboratory, University of Cambridge
Kavli Institute for Cosmology, Cambridge
Gonville & Caius College
willhandley.co.uk/talks

16th May 2024



UNIVERSITY OF
CAMBRIDGE



Contents

1. Likelihood- vs Simulation-based inference (LBI vs SBI)
2. Neural Ratio estimation (NRE)
3. Nested sampling (NS)
4. NS+NRE
5. Future prospects

Stems from over a year of discussion, with the majority of the work done by Kilian Scheutwinkel (PhD student).

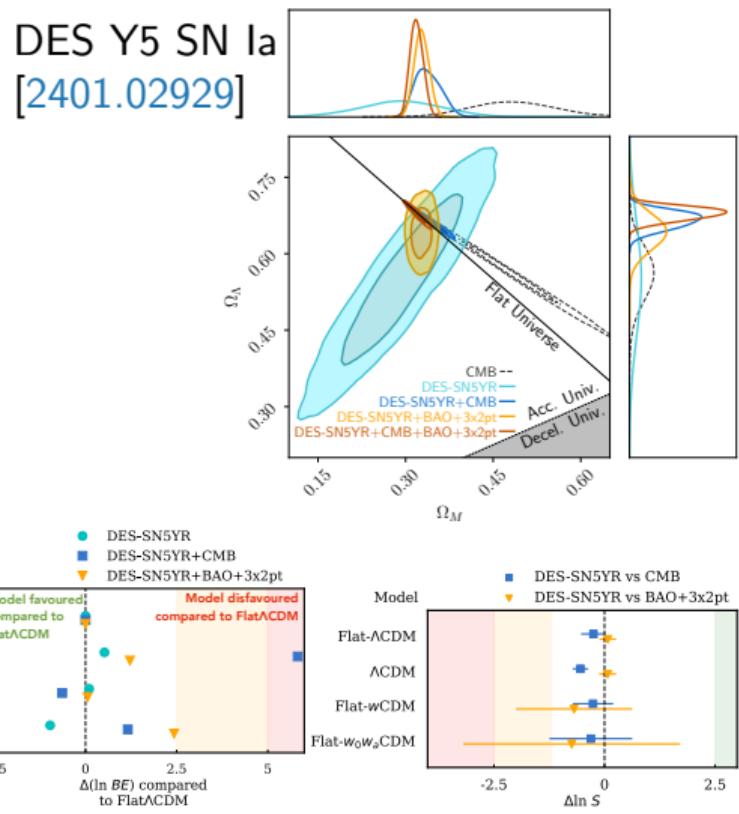


LBI: Likelihood-based inference

The standard approach if you are fortunate enough to have a likelihood function $P(D|\theta)$:

$$P(\theta|D) = \frac{P(D|\theta)P(\theta)}{P(D)}$$

1. Define prior $\pi(\theta)$
 - ▶ spend some time being philosophical
2. Sample posterior $P(\theta|D)$
 - ▶ use out-of-the-box MCMC tools such as emcee or MultiNest
 - ▶ make some triangle plots
3. Optionally compute evidence $\mathcal{Z}(D)$
 - ▶ e.g. nested sampling or parallel tempering
 - ▶ do some model comparison (i.e. science)
 - ▶ talk about tensions



LBI: Likelihood-based inference

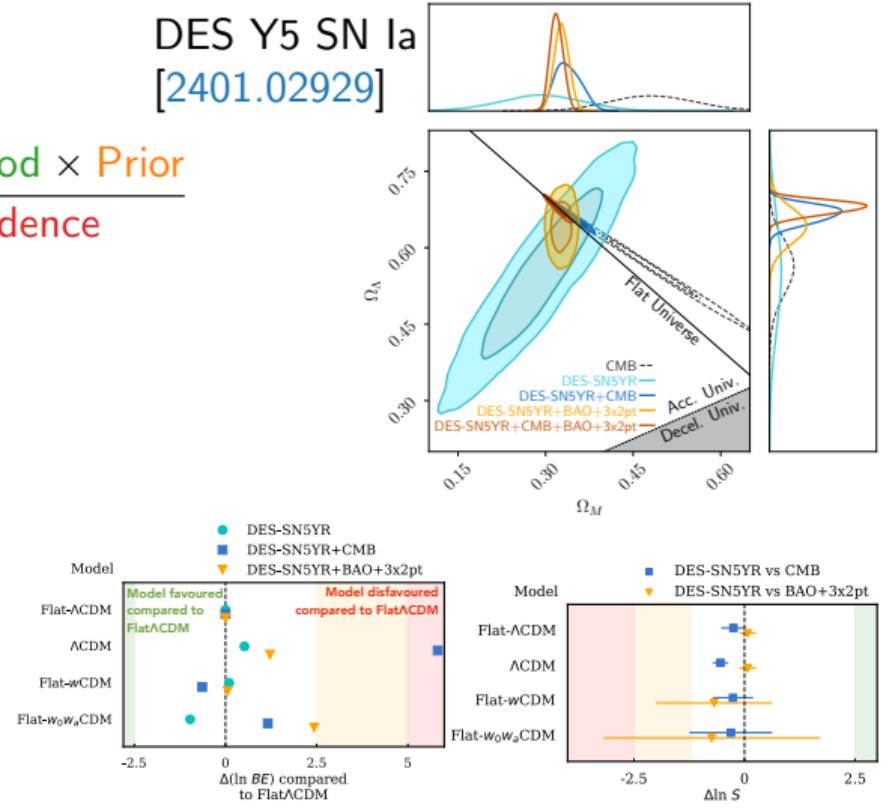
The standard approach if you are fortunate enough to have a likelihood function $P(D|\theta)$:

$$P(\theta|D) = \frac{P(D|\theta)P(\theta)}{P(D)}$$

$$\text{Posterior} = \frac{\text{Likelihood} \times \text{Prior}}{\text{Evidence}}$$

1. Define prior $\pi(\theta)$
 - ▶ spend some time being philosophical
2. Sample posterior $P(\theta|D)$
 - ▶ use out-of-the-box MCMC tools such as emcee or MultiNest
 - ▶ make some triangle plots
3. Optionally compute evidence $\mathcal{Z}(D)$
 - ▶ e.g. nested sampling or parallel tempering
 - ▶ do some model comparison (i.e. science)
 - ▶ talk about tensions

DES Y5 SN Ia
[2401.02929]



LBI: Likelihood-based inference

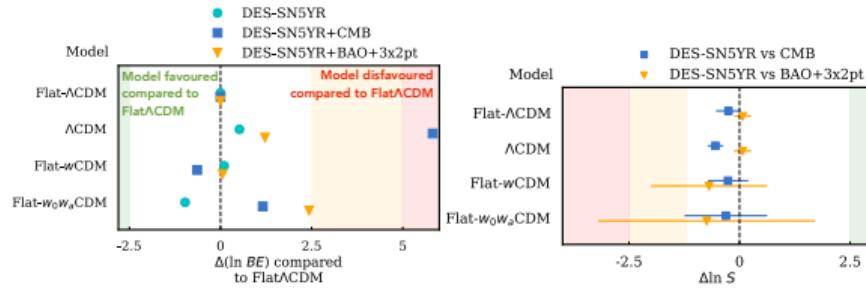
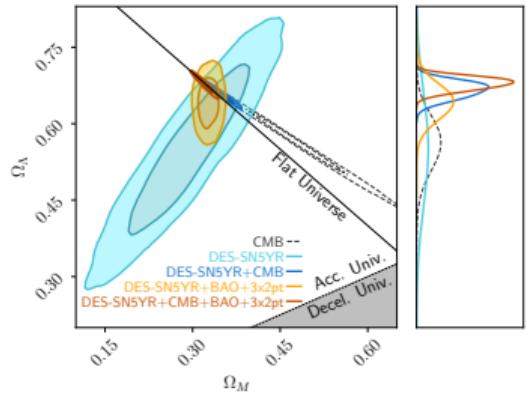
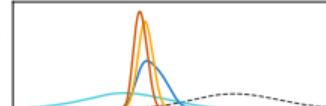
The standard approach if you are fortunate enough to have a likelihood function $\mathcal{L}(D|\theta)$:

$$\mathcal{P}(\theta|D) = \frac{\mathcal{L}(D|\theta)\pi(\theta)}{\mathcal{Z}(D)}$$

$$\text{Posterior} = \frac{\text{Likelihood} \times \text{Prior}}{\text{Evidence}}$$

1. Define prior $\pi(\theta)$
 - ▶ spend some time being philosophical
2. Sample posterior $\mathcal{P}(\theta|D)$
 - ▶ use out-of-the-box MCMC tools such as emcee or MultiNest
 - ▶ make some triangle plots
3. Optionally compute evidence $\mathcal{Z}(D)$
 - ▶ e.g. nested sampling or parallel tempering
 - ▶ do some model comparison (i.e. science)
 - ▶ talk about tensions

DES Y5 SN Ia
[2401.02929]

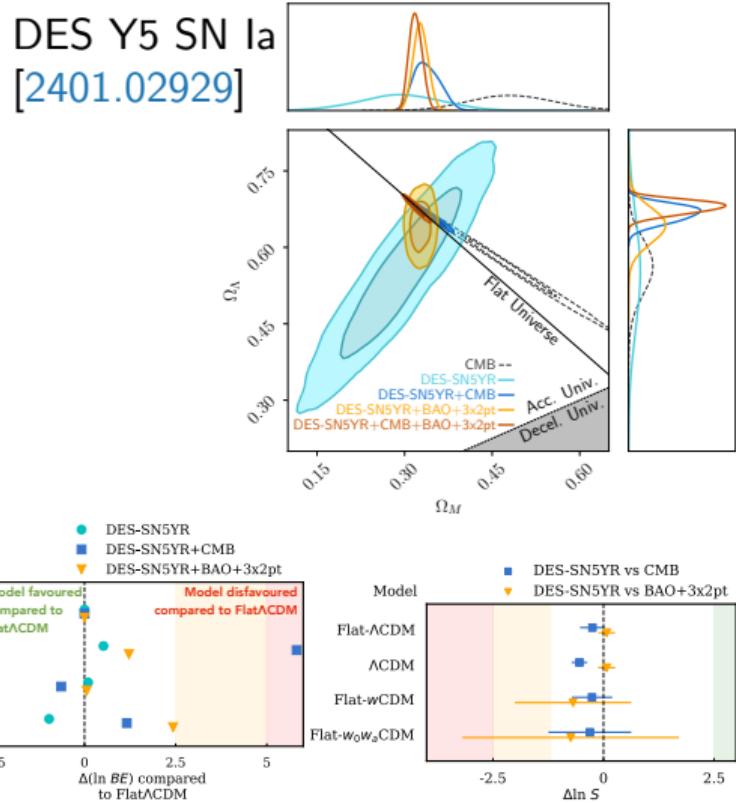


LBI: Likelihood-based inference

The standard approach if you are fortunate enough to have a likelihood function $\mathcal{L}(D|\theta)$:

$$P(\theta|D)P(D) = P(D, \theta) = P(D|\theta)P(\theta),$$

1. Define prior $\pi(\theta)$
 - ▶ spend some time being philosophical
2. Sample posterior $P(\theta|D)$
 - ▶ use out-of-the-box MCMC tools such as emcee or MultiNest
 - ▶ make some triangle plots
3. Optionally compute evidence $\mathcal{Z}(D)$
 - ▶ e.g. nested sampling or parallel tempering
 - ▶ do some model comparison (i.e. science)
 - ▶ talk about tensions

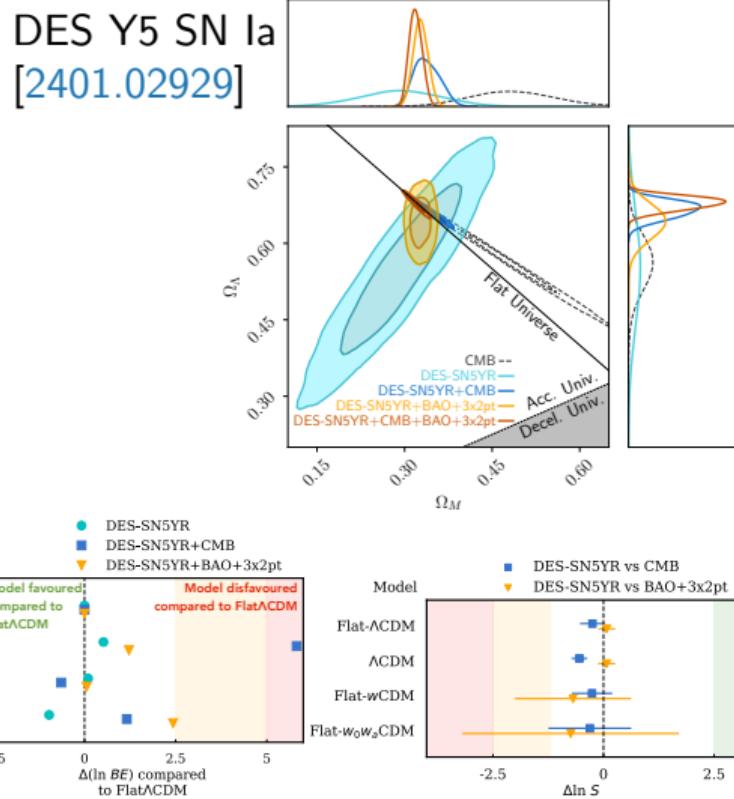


LBI: Likelihood-based inference

The standard approach if you are fortunate enough to have a likelihood function $\mathcal{L}(D|\theta)$:

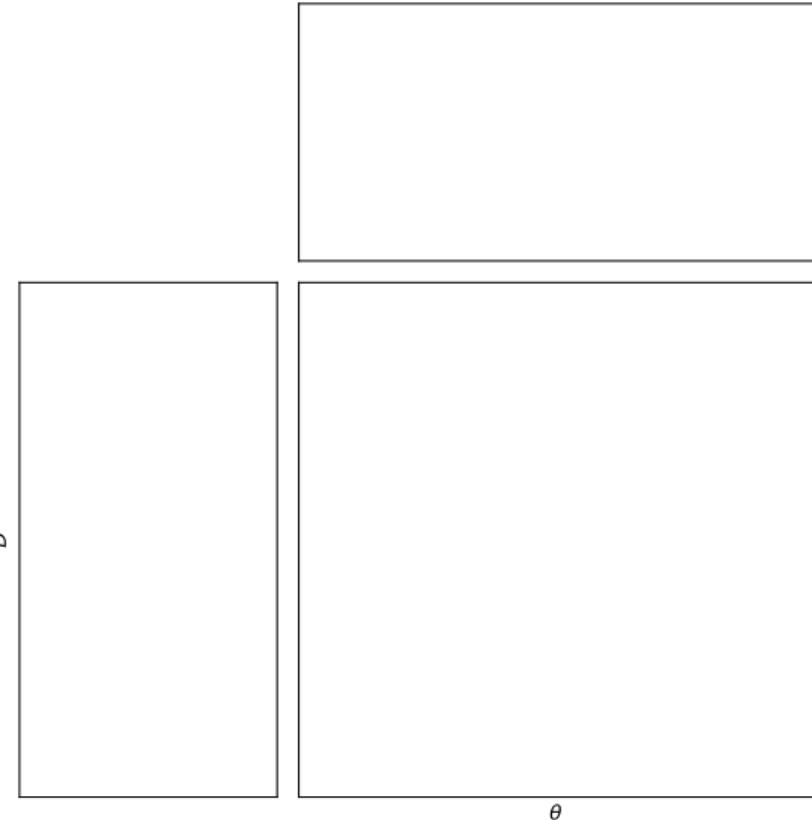
$$\mathcal{P} \times \mathcal{Z} = \mathcal{J} = \mathcal{L} \times \pi, \quad \text{Joint} = \mathcal{J} = P(D, \theta)$$

1. Define prior $\pi(\theta)$
 - ▶ spend some time being philosophical
2. Sample posterior $P(\theta|D)$
 - ▶ use out-of-the-box MCMC tools such as emcee or MultiNest
 - ▶ make some triangle plots
3. Optionally compute evidence $\mathcal{Z}(D)$
 - ▶ e.g. nested sampling or parallel tempering
 - ▶ do some model comparison (i.e. science)
 - ▶ talk about tensions



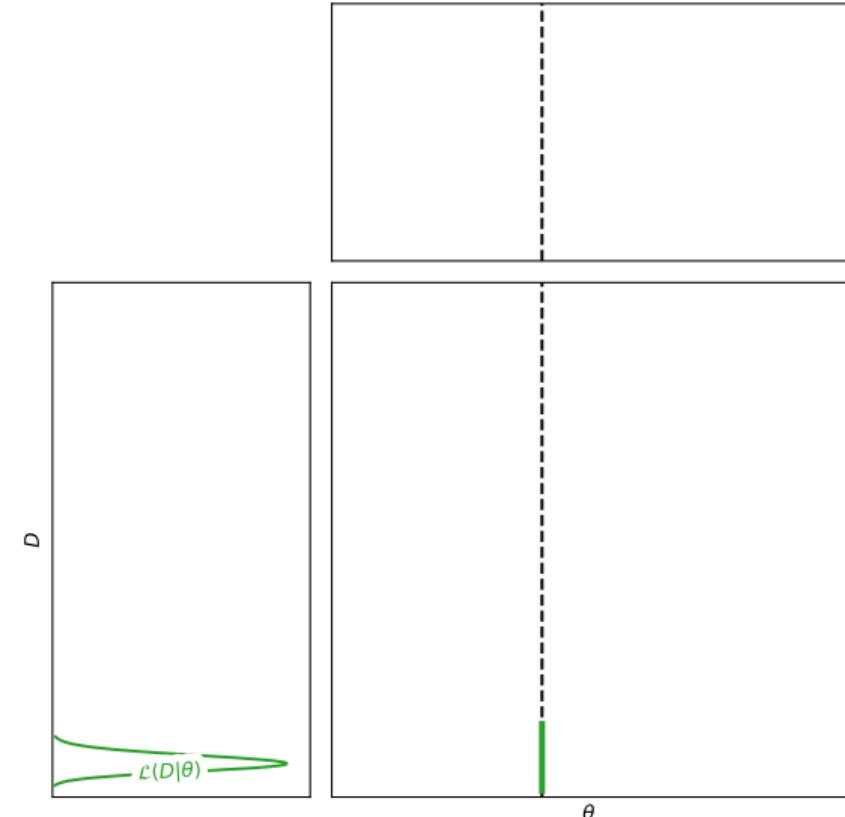
SBI: Simulation-based inference

- ▶ What do you do if you don't know $\mathcal{L}(D|\theta)$?
- ▶ If you have a simulator/forward model
 $\theta \rightarrow D$. defines an *implicit likelihood* \mathcal{L}
- ▶ Simulator generates samples from $\mathcal{L}(D|\theta)$.
- ▶ With a prior $\pi(\theta)$ can generate samples from
joint distribution $\mathcal{J}(\theta, D) = \mathcal{L}(D|\theta)\pi(\theta)$
the “probability of everything”.
- ▶ Task of SBI is take joint \mathcal{J} samples and
learn posterior $\mathcal{P}(\theta|D)$ and evidence $\mathcal{Z}(D)$
and possibly likelihood $\mathcal{L}(D|\theta)$.
- ▶ Present state of the art achieves this using
machine learning (neural networks).
 - ▶ My group's research tries to removes machine learning github.com/handley-lab/lsbi.



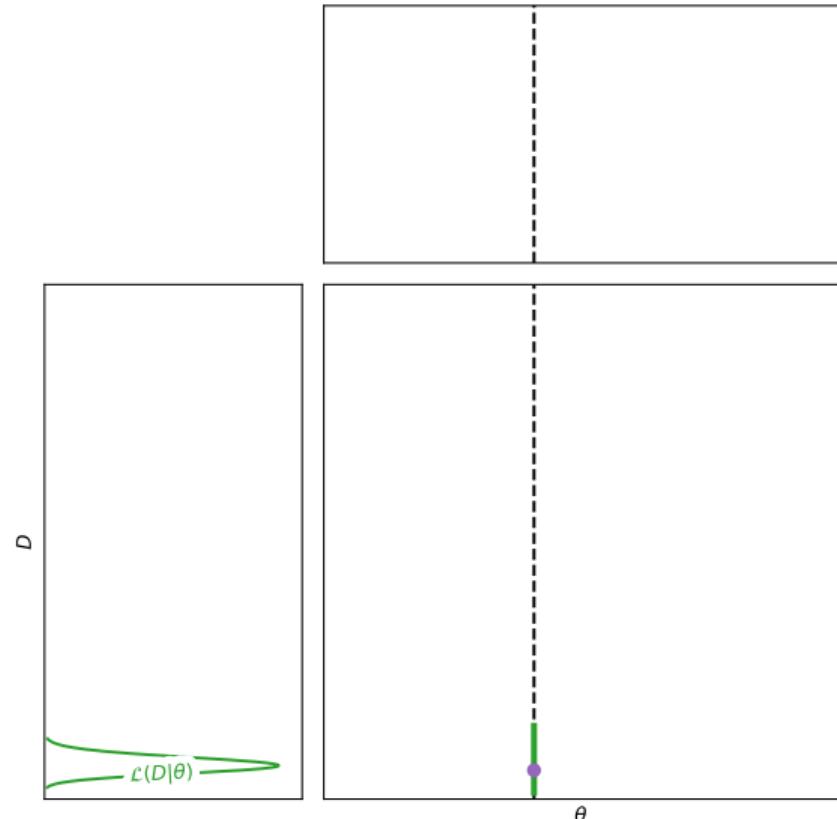
SBI: Simulation-based inference

- ▶ What do you do if you don't know $\mathcal{L}(D|\theta)$?
- ▶ If you have a simulator/forward model
 $\theta \rightarrow D$. defines an *implicit likelihood* \mathcal{L}
- ▶ Simulator generates samples from $\mathcal{L}(D|\theta)$.
- ▶ With a prior $\pi(\theta)$ can generate samples from
joint distribution $\mathcal{J}(\theta, D) = \mathcal{L}(D|\theta)\pi(\theta)$
the “probability of everything”.
- ▶ Task of SBI is take joint \mathcal{J} samples and
learn posterior $\mathcal{P}(\theta|D)$ and evidence $\mathcal{Z}(D)$
and possibly likelihood $\mathcal{L}(D|\theta)$.
- ▶ Present state of the art achieves this using
machine learning (neural networks).
 - ▶ My group's research tries to removes machine learning github.com/handley-lab/lsbi.



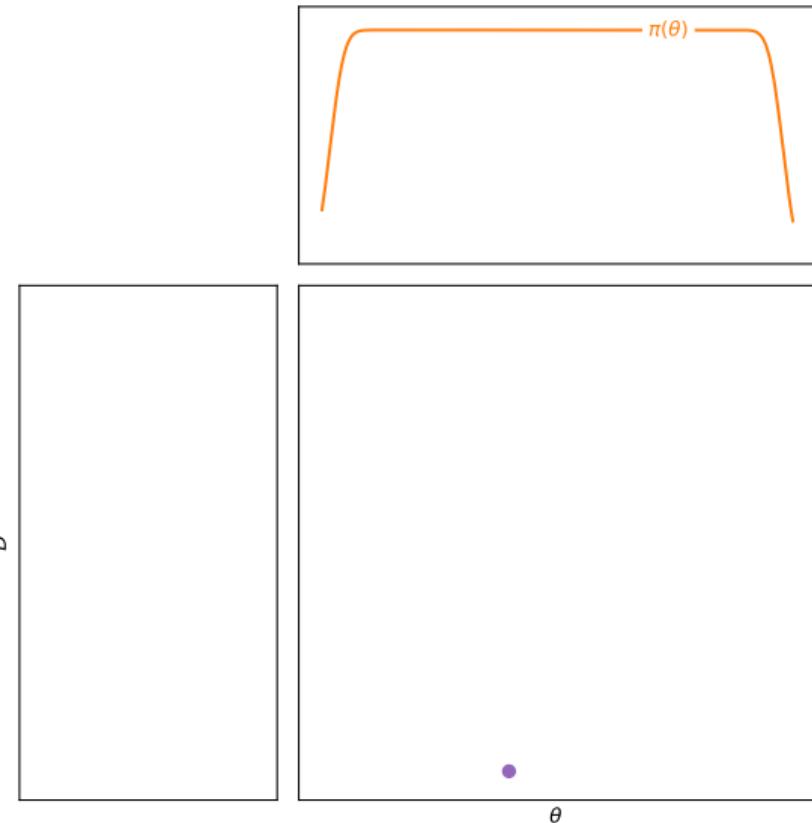
SBI: Simulation-based inference

- ▶ What do you do if you don't know $\mathcal{L}(D|\theta)$?
- ▶ If you have a simulator/forward model
 $\theta \rightarrow D$. defines an *implicit likelihood* \mathcal{L}
- ▶ Simulator generates samples from $\mathcal{L}(D|\theta)$.
- ▶ With a prior $\pi(\theta)$ can generate samples from joint distribution $\mathcal{J}(\theta, D) = \mathcal{L}(D|\theta)\pi(\theta)$
the “*probability of everything*”.
- ▶ Task of SBI is take joint \mathcal{J} samples and learn posterior $\mathcal{P}(\theta|D)$ and evidence $\mathcal{Z}(D)$
and possibly likelihood $\mathcal{L}(D|\theta)$.
- ▶ Present state of the art achieves this using *machine learning* (neural networks).
 - ▶ My group's research tries to removes machine learning github.com/handley-lab/lstbi.



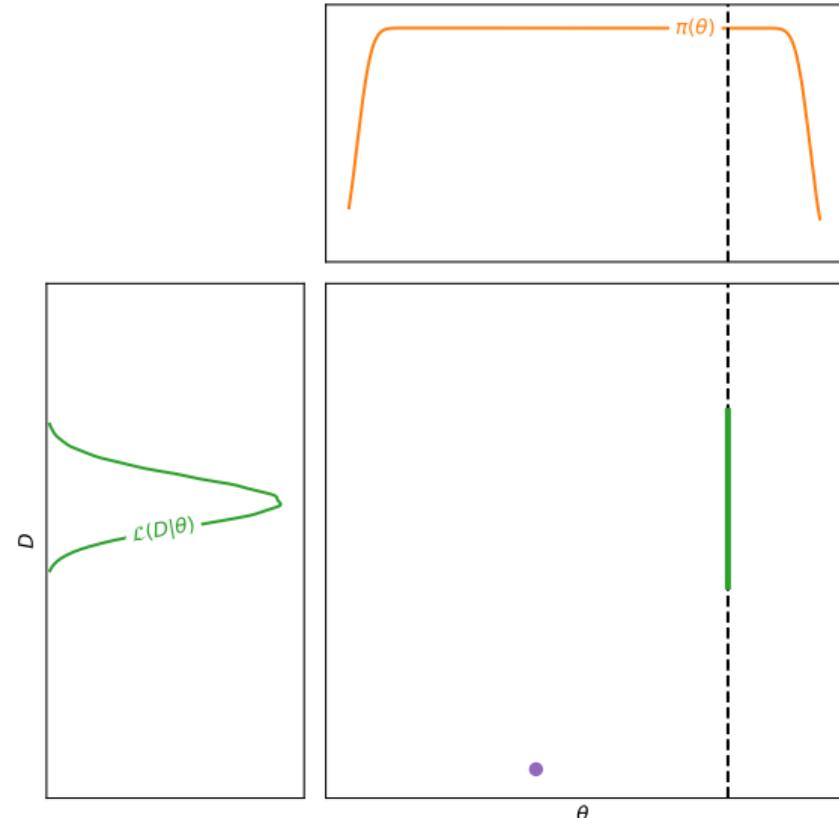
SBI: Simulation-based inference

- ▶ What do you do if you don't know $\mathcal{L}(D|\theta)$?
- ▶ If you have a simulator/forward model
 $\theta \rightarrow D$. defines an *implicit likelihood* \mathcal{L}
- ▶ Simulator generates samples from $\mathcal{L}(D|\theta)$.
- ▶ With a prior $\pi(\theta)$ can generate samples from
joint distribution $\mathcal{J}(\theta, D) = \mathcal{L}(D|\theta)\pi(\theta)$
the “*probability of everything*”.
- ▶ Task of SBI is take joint \mathcal{J} samples and
learn posterior $\mathcal{P}(\theta|D)$ and evidence $\mathcal{Z}(D)$
and possibly likelihood $\mathcal{L}(D|\theta)$.
- ▶ Present state of the art achieves this using
machine learning (neural networks).
 - ▶ My group's research tries to removes machine learning github.com/handley-lab/lsbi.



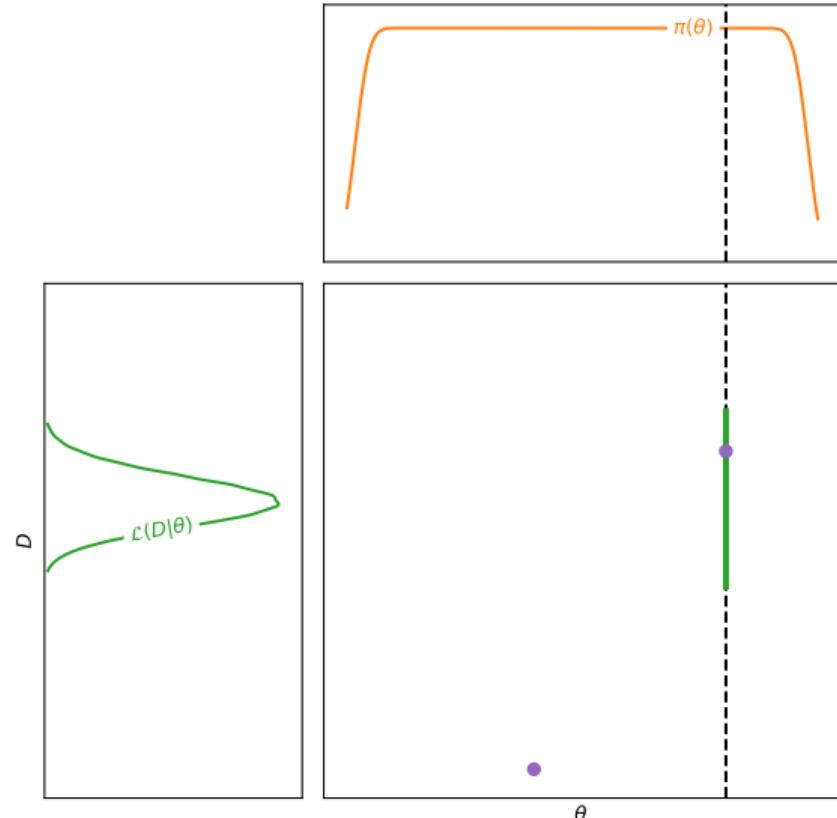
SBI: Simulation-based inference

- ▶ What do you do if you don't know $\mathcal{L}(D|\theta)$?
- ▶ If you have a simulator/forward model
 $\theta \rightarrow D$. defines an *implicit likelihood* \mathcal{L}
- ▶ Simulator generates samples from $\mathcal{L}(D|\theta)$.
- ▶ With a prior $\pi(\theta)$ can generate samples from joint distribution $\mathcal{J}(\theta, D) = \mathcal{L}(D|\theta)\pi(\theta)$
the “*probability of everything*”.
- ▶ Task of SBI is take joint \mathcal{J} samples and learn posterior $\mathcal{P}(\theta|D)$ and evidence $\mathcal{Z}(D)$
and possibly likelihood $\mathcal{L}(D|\theta)$.
- ▶ Present state of the art achieves this using *machine learning* (neural networks).
 - ▶ My group's research tries to removes machine learning github.com/handley-lab/lsbi.



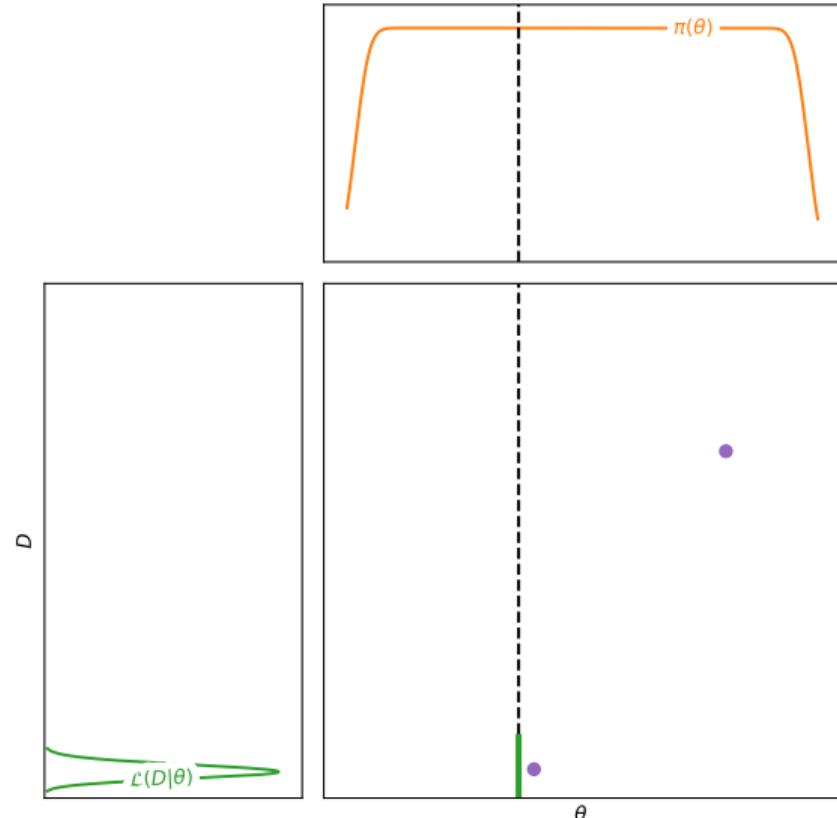
SBI: Simulation-based inference

- ▶ What do you do if you don't know $\mathcal{L}(D|\theta)$?
- ▶ If you have a simulator/forward model
 $\theta \rightarrow D$. defines an *implicit likelihood* \mathcal{L}
- ▶ Simulator generates samples from $\mathcal{L}(D|\theta)$.
- ▶ With a prior $\pi(\theta)$ can generate samples from joint distribution $\mathcal{J}(\theta, D) = \mathcal{L}(D|\theta)\pi(\theta)$
the “*probability of everything*”.
- ▶ Task of SBI is take joint \mathcal{J} samples and learn posterior $\mathcal{P}(\theta|D)$ and evidence $\mathcal{Z}(D)$
and possibly likelihood $\mathcal{L}(D|\theta)$.
- ▶ Present state of the art achieves this using *machine learning* (neural networks).
 - ▶ My group's research tries to removes machine learning github.com/handley-lab/lsbi.



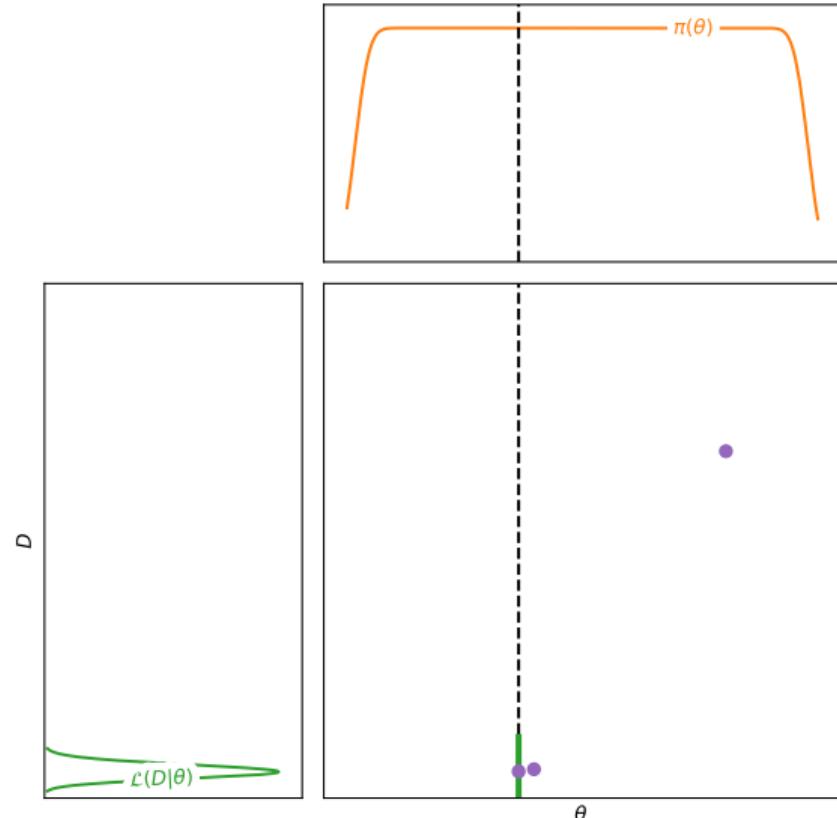
SBI: Simulation-based inference

- ▶ What do you do if you don't know $\mathcal{L}(D|\theta)$?
- ▶ If you have a simulator/forward model
 $\theta \rightarrow D$. defines an *implicit likelihood* \mathcal{L}
- ▶ Simulator generates samples from $\mathcal{L}(D|\theta)$.
- ▶ With a prior $\pi(\theta)$ can generate samples from joint distribution $\mathcal{J}(\theta, D) = \mathcal{L}(D|\theta)\pi(\theta)$
the “*probability of everything*”.
- ▶ Task of SBI is take joint \mathcal{J} samples and learn posterior $\mathcal{P}(\theta|D)$ and evidence $\mathcal{Z}(D)$
and possibly likelihood $\mathcal{L}(D|\theta)$.
- ▶ Present state of the art achieves this using *machine learning* (neural networks).
 - ▶ My group's research tries to removes machine learning github.com/handley-lab/lsbi.



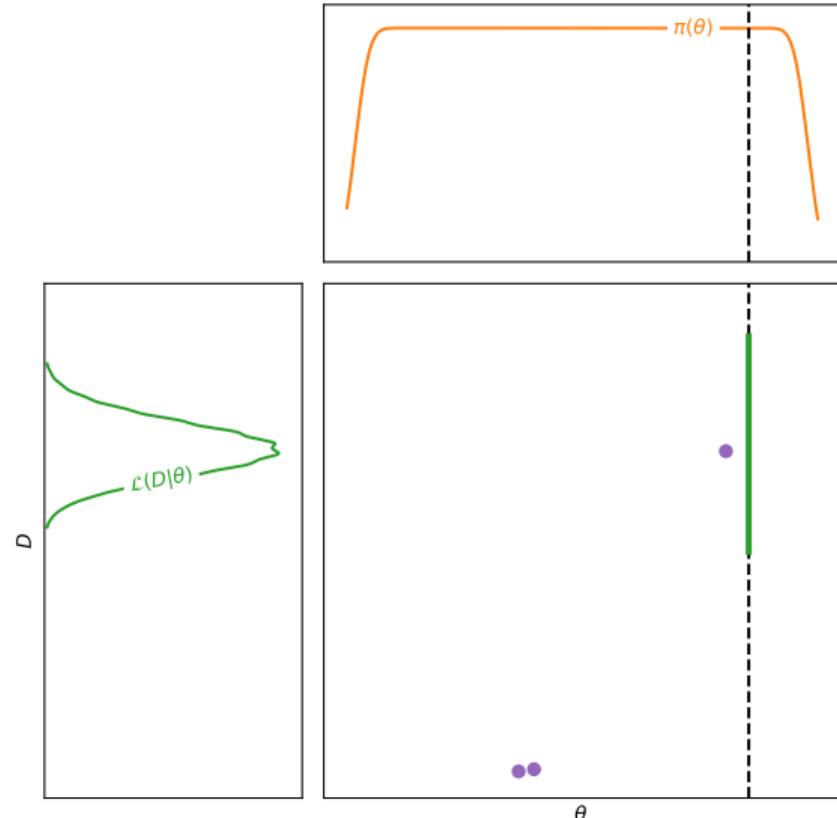
SBI: Simulation-based inference

- ▶ What do you do if you don't know $\mathcal{L}(D|\theta)$?
- ▶ If you have a simulator/forward model
 $\theta \rightarrow D$. defines an *implicit likelihood* \mathcal{L}
- ▶ Simulator generates samples from $\mathcal{L}(D|\theta)$.
- ▶ With a prior $\pi(\theta)$ can generate samples from joint distribution $\mathcal{J}(\theta, D) = \mathcal{L}(D|\theta)\pi(\theta)$
the “*probability of everything*”.
- ▶ Task of SBI is take joint \mathcal{J} samples and learn posterior $\mathcal{P}(\theta|D)$ and evidence $\mathcal{Z}(D)$
and possibly likelihood $\mathcal{L}(D|\theta)$.
- ▶ Present state of the art achieves this using *machine learning* (neural networks).
 - ▶ My group's research tries to removes machine learning github.com/handley-lab/lsbi.



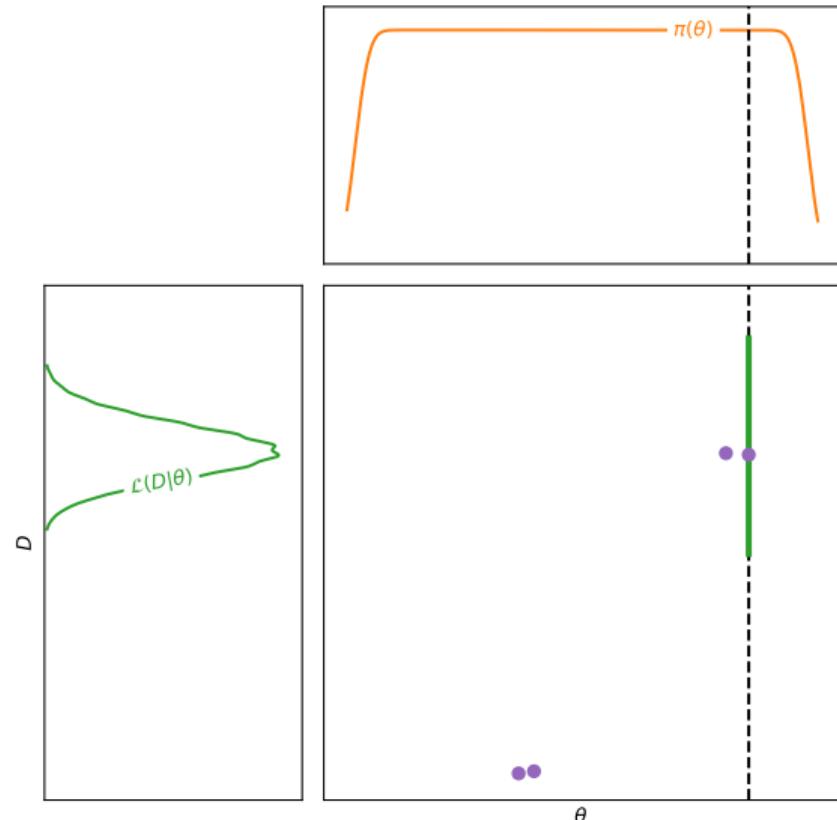
SBI: Simulation-based inference

- ▶ What do you do if you don't know $\mathcal{L}(D|\theta)$?
- ▶ If you have a simulator/forward model
 $\theta \rightarrow D$. defines an *implicit likelihood* \mathcal{L}
- ▶ Simulator generates samples from $\mathcal{L}(D|\theta)$.
- ▶ With a prior $\pi(\theta)$ can generate samples from joint distribution $\mathcal{J}(\theta, D) = \mathcal{L}(D|\theta)\pi(\theta)$
the “probability of everything”.
- ▶ Task of SBI is take joint \mathcal{J} samples and learn posterior $\mathcal{P}(\theta|D)$ and evidence $\mathcal{Z}(D)$
and possibly likelihood $\mathcal{L}(D|\theta)$.
- ▶ Present state of the art achieves this using *machine learning* (neural networks).
 - ▶ My group's research tries to removes machine learning github.com/handley-lab/lstbi.



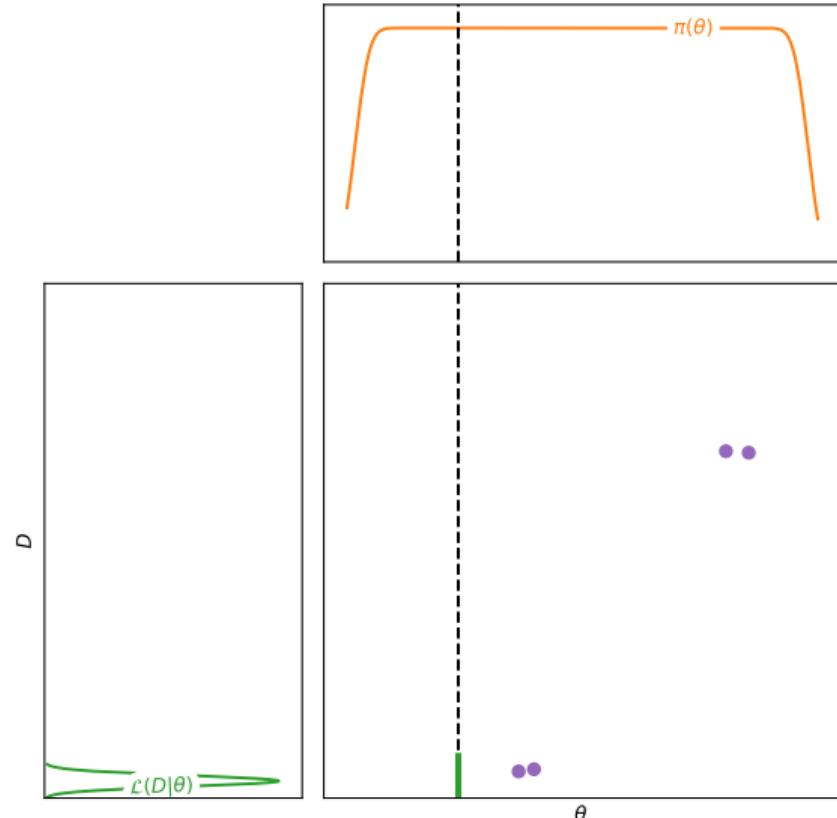
SBI: Simulation-based inference

- ▶ What do you do if you don't know $\mathcal{L}(D|\theta)$?
- ▶ If you have a simulator/forward model
 $\theta \rightarrow D$. defines an *implicit likelihood* \mathcal{L}
- ▶ Simulator generates samples from $\mathcal{L}(D|\theta)$.
- ▶ With a prior $\pi(\theta)$ can generate samples from joint distribution $\mathcal{J}(\theta, D) = \mathcal{L}(D|\theta)\pi(\theta)$
the “probability of everything”.
- ▶ Task of SBI is take joint \mathcal{J} samples and learn posterior $\mathcal{P}(\theta|D)$ and evidence $\mathcal{Z}(D)$
and possibly likelihood $\mathcal{L}(D|\theta)$.
- ▶ Present state of the art achieves this using *machine learning* (neural networks).
 - ▶ My group's research tries to removes machine learning github.com/handley-lab/lsbi.



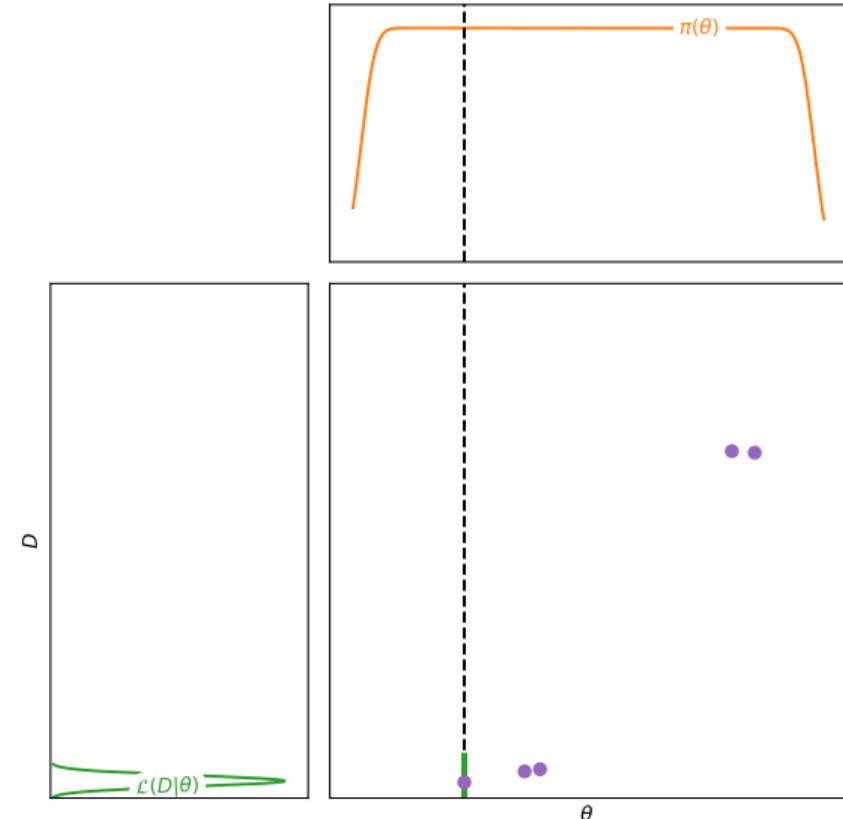
SBI: Simulation-based inference

- ▶ What do you do if you don't know $\mathcal{L}(D|\theta)$?
- ▶ If you have a simulator/forward model
 $\theta \rightarrow D$. defines an *implicit likelihood* \mathcal{L}
- ▶ Simulator generates samples from $\mathcal{L}(D|\theta)$.
- ▶ With a prior $\pi(\theta)$ can generate samples from joint distribution $\mathcal{J}(\theta, D) = \mathcal{L}(D|\theta)\pi(\theta)$
the “*probability of everything*”.
- ▶ Task of SBI is take joint \mathcal{J} samples and learn posterior $\mathcal{P}(\theta|D)$ and evidence $\mathcal{Z}(D)$
and possibly likelihood $\mathcal{L}(D|\theta)$.
- ▶ Present state of the art achieves this using *machine learning* (neural networks).
 - ▶ My group's research tries to removes machine learning github.com/handley-lab/lsbi.



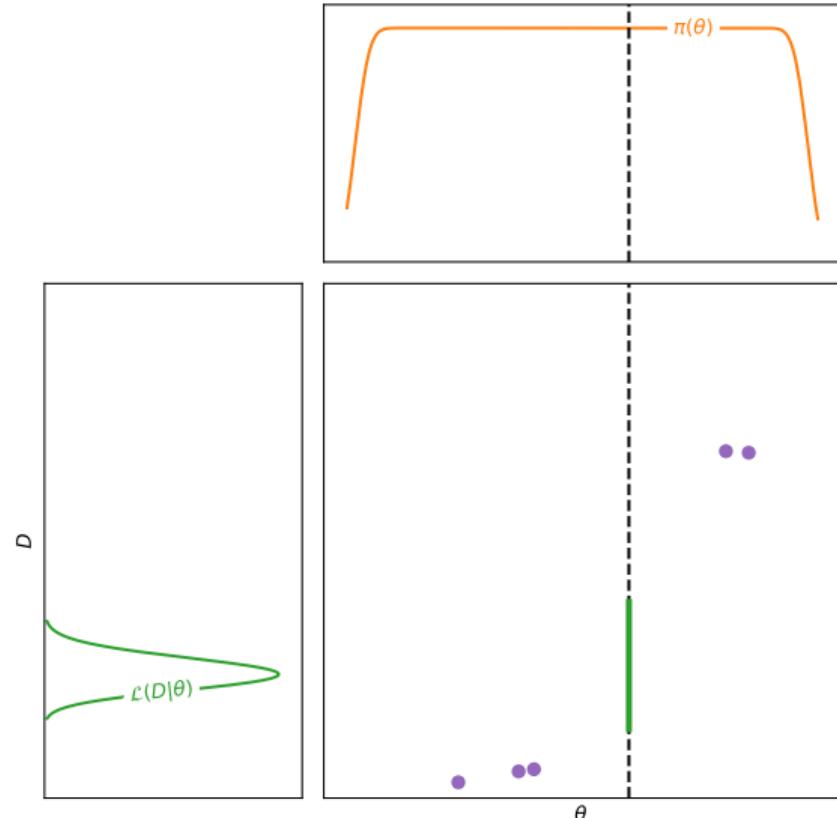
SBI: Simulation-based inference

- ▶ What do you do if you don't know $\mathcal{L}(D|\theta)$?
- ▶ If you have a simulator/forward model
 $\theta \rightarrow D$. defines an *implicit likelihood* \mathcal{L}
- ▶ Simulator generates samples from $\mathcal{L}(D|\theta)$.
- ▶ With a prior $\pi(\theta)$ can generate samples from joint distribution $\mathcal{J}(\theta, D) = \mathcal{L}(D|\theta)\pi(\theta)$
the “*probability of everything*”.
- ▶ Task of SBI is take joint \mathcal{J} samples and learn posterior $\mathcal{P}(\theta|D)$ and evidence $\mathcal{Z}(D)$
and possibly likelihood $\mathcal{L}(D|\theta)$.
- ▶ Present state of the art achieves this using *machine learning* (neural networks).
 - ▶ My group's research tries to removes machine learning github.com/handley-lab/lstbi.



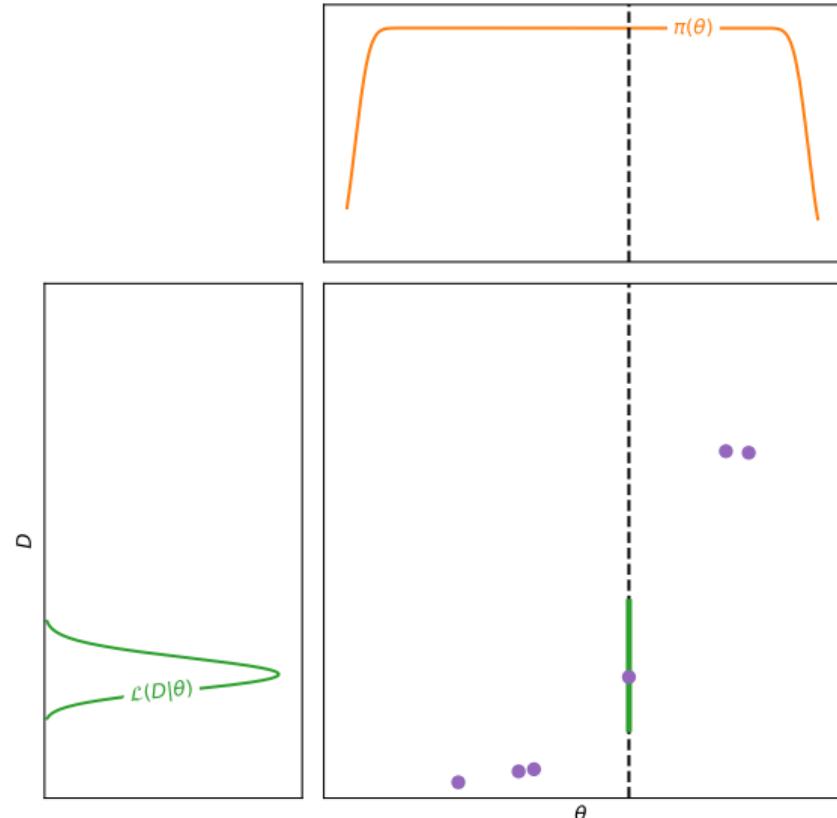
SBI: Simulation-based inference

- ▶ What do you do if you don't know $\mathcal{L}(D|\theta)$?
- ▶ If you have a simulator/forward model
 $\theta \rightarrow D$. defines an *implicit likelihood* \mathcal{L}
- ▶ Simulator generates samples from $\mathcal{L}(D|\theta)$.
- ▶ With a prior $\pi(\theta)$ can generate samples from joint distribution $\mathcal{J}(\theta, D) = \mathcal{L}(D|\theta)\pi(\theta)$
the “*probability of everything*”.
- ▶ Task of SBI is take joint \mathcal{J} samples and learn posterior $\mathcal{P}(\theta|D)$ and evidence $\mathcal{Z}(D)$
and possibly likelihood $\mathcal{L}(D|\theta)$.
- ▶ Present state of the art achieves this using *machine learning* (neural networks).
 - ▶ My group's research tries to removes machine learning github.com/handley-lab/lsbi.



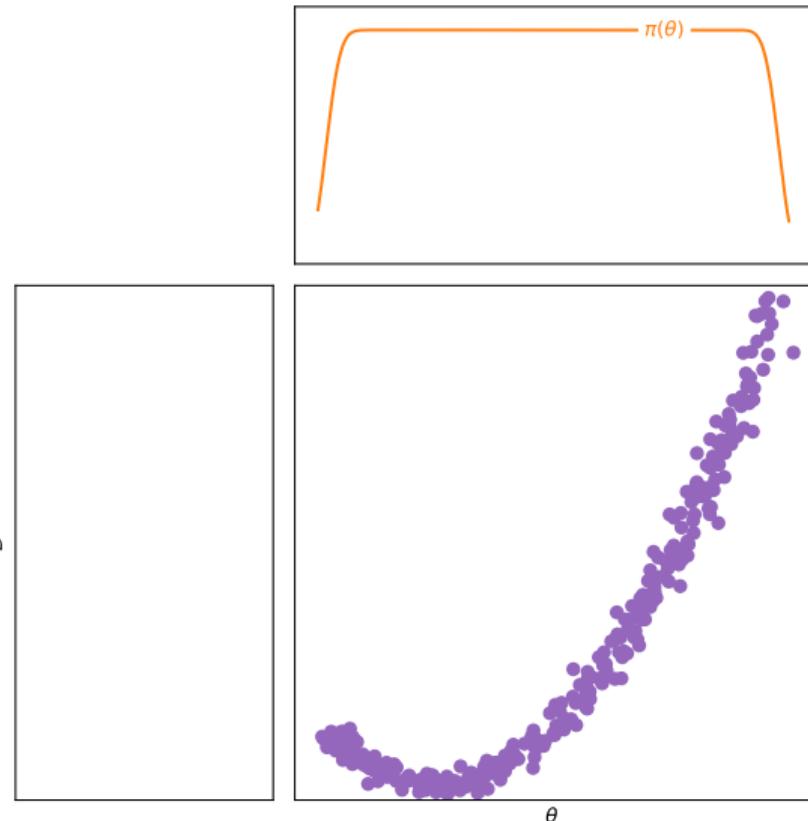
SBI: Simulation-based inference

- ▶ What do you do if you don't know $\mathcal{L}(D|\theta)$?
- ▶ If you have a simulator/forward model
 $\theta \rightarrow D$. defines an *implicit likelihood* \mathcal{L}
- ▶ Simulator generates samples from $\mathcal{L}(D|\theta)$.
- ▶ With a prior $\pi(\theta)$ can generate samples from joint distribution $\mathcal{J}(\theta, D) = \mathcal{L}(D|\theta)\pi(\theta)$
the “*probability of everything*”.
- ▶ Task of SBI is take joint \mathcal{J} samples and learn posterior $\mathcal{P}(\theta|D)$ and evidence $\mathcal{Z}(D)$
and possibly likelihood $\mathcal{L}(D|\theta)$.
- ▶ Present state of the art achieves this using *machine learning* (neural networks).
 - ▶ My group's research tries to removes machine learning github.com/handley-lab/lsbi.



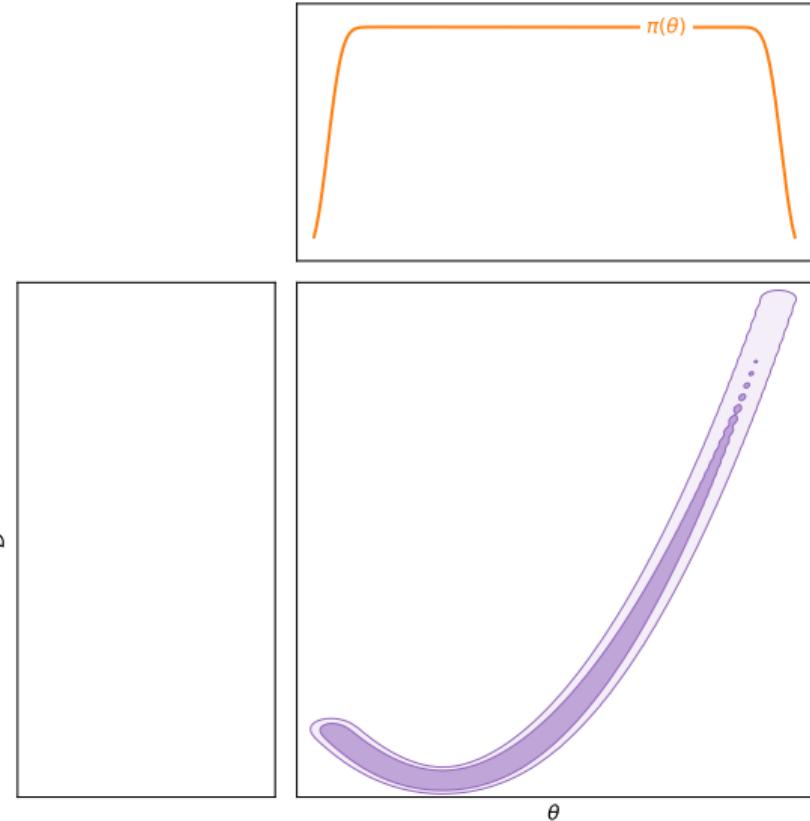
SBI: Simulation-based inference

- ▶ What do you do if you don't know $\mathcal{L}(D|\theta)$?
- ▶ If you have a simulator/forward model
 $\theta \rightarrow D$. defines an *implicit likelihood* \mathcal{L}
- ▶ Simulator generates samples from $\mathcal{L}(D|\theta)$.
- ▶ With a prior $\pi(\theta)$ can generate samples from joint distribution $\mathcal{J}(\theta, D) = \mathcal{L}(D|\theta)\pi(\theta)$
the “*probability of everything*”.
- ▶ Task of SBI is take joint \mathcal{J} samples and learn posterior $\mathcal{P}(\theta|D)$ and evidence $\mathcal{Z}(D)$
and possibly likelihood $\mathcal{L}(D|\theta)$.
- ▶ Present state of the art achieves this using *machine learning* (neural networks).
 - ▶ My group's research tries to removes machine learning github.com/handley-lab/lsbi.



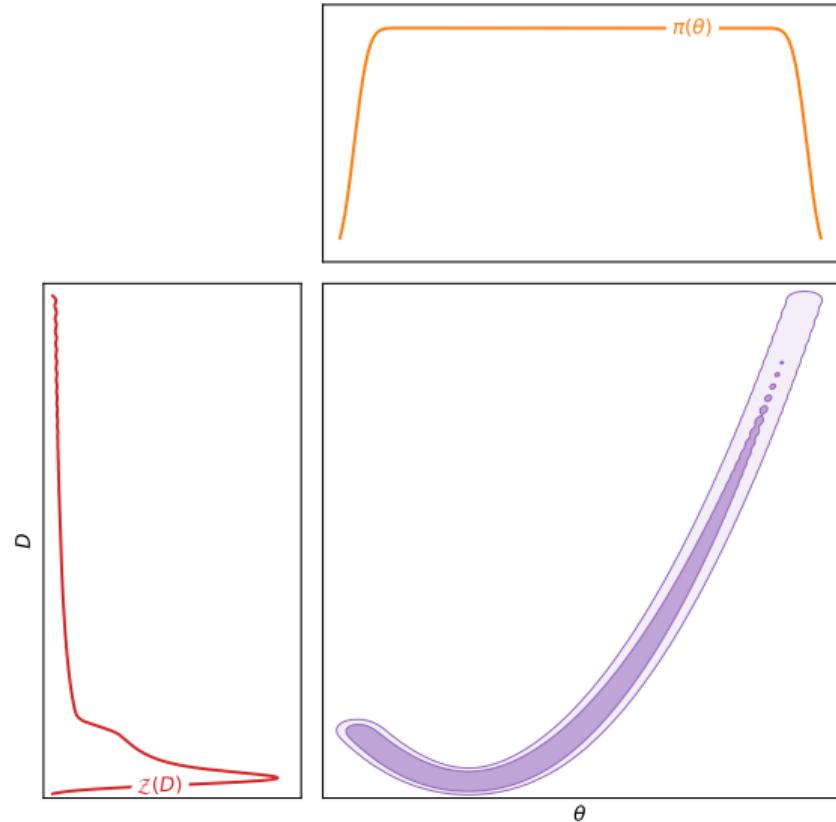
SBI: Simulation-based inference

- ▶ What do you do if you don't know $\mathcal{L}(D|\theta)$?
- ▶ If you have a simulator/forward model
 $\theta \rightarrow D$. defines an *implicit likelihood* \mathcal{L}
- ▶ Simulator generates samples from $\mathcal{L}(D|\theta)$.
- ▶ With a prior $\pi(\theta)$ can generate samples from joint distribution $\mathcal{J}(\theta, D) = \mathcal{L}(D|\theta)\pi(\theta)$
the “*probability of everything*”.
- ▶ Task of SBI is take joint \mathcal{J} samples and learn posterior $\mathcal{P}(\theta|D)$ and evidence $\mathcal{Z}(D)$
and possibly likelihood $\mathcal{L}(D|\theta)$.
- ▶ Present state of the art achieves this using *machine learning* (neural networks).
 - ▶ My group's research tries to removes machine learning github.com/handley-lab/lsbi.



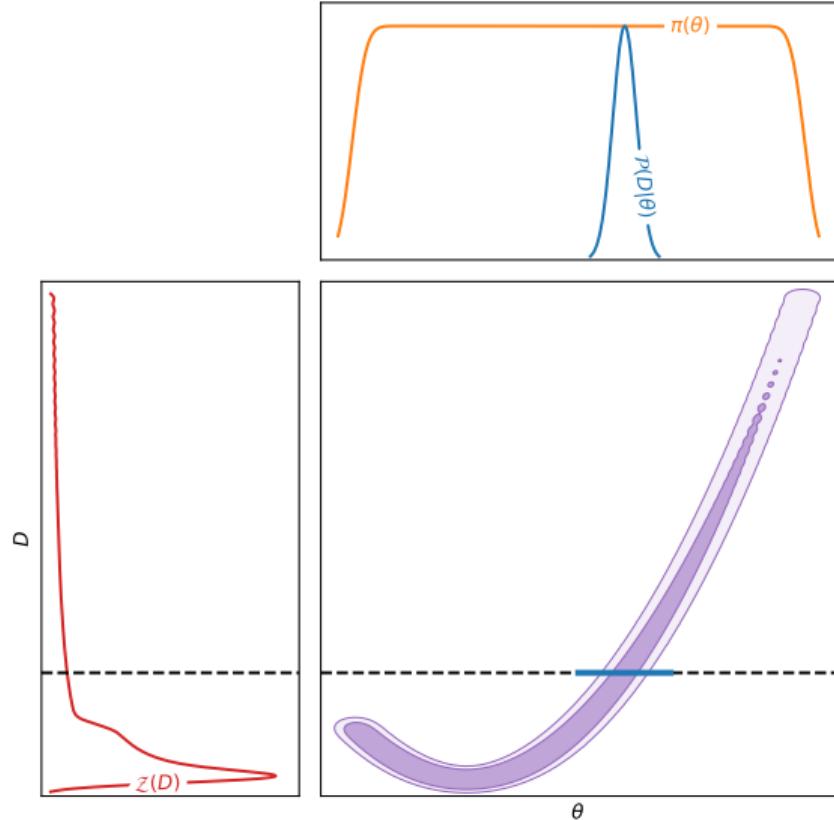
SBI: Simulation-based inference

- ▶ What do you do if you don't know $\mathcal{L}(D|\theta)$?
- ▶ If you have a simulator/forward model
 $\theta \rightarrow D$. defines an *implicit likelihood* \mathcal{L}
- ▶ Simulator generates samples from $\mathcal{L}(D|\theta)$.
- ▶ With a prior $\pi(\theta)$ can generate samples from joint distribution $\mathcal{J}(\theta, D) = \mathcal{L}(D|\theta)\pi(\theta)$
the “probability of everything”.
- ▶ Task of SBI is take joint \mathcal{J} samples and learn posterior $\mathcal{P}(\theta|D)$ and evidence $\mathcal{Z}(D)$
and possibly likelihood $\mathcal{L}(D|\theta)$.
- ▶ Present state of the art achieves this using *machine learning* (neural networks).
 - ▶ My group's research tries to removes machine learning github.com/handley-lab/lsbi.



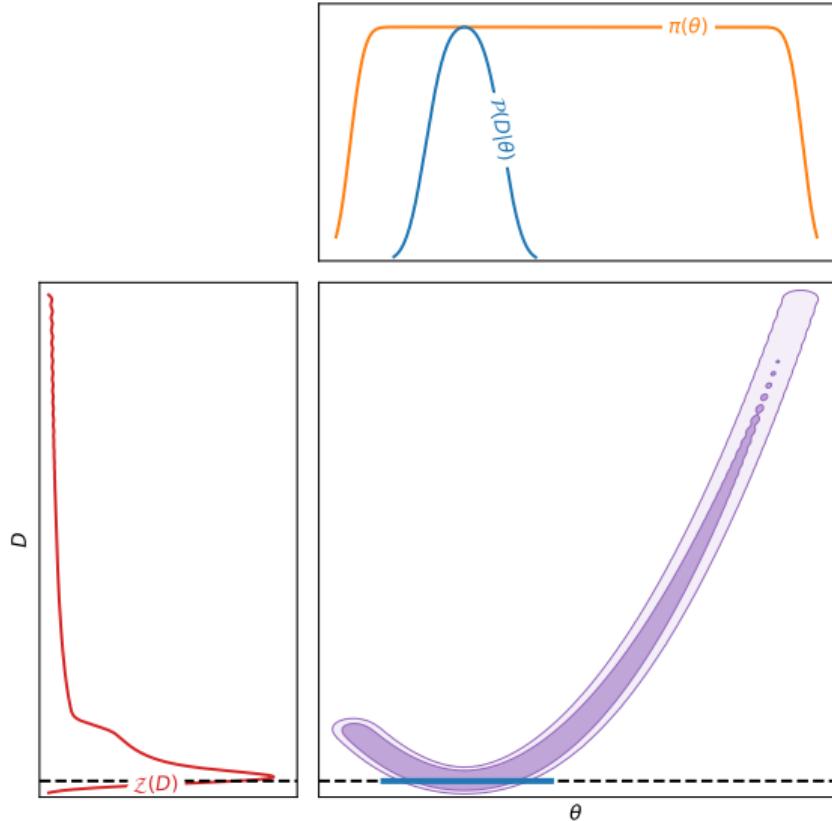
SBI: Simulation-based inference

- ▶ What do you do if you don't know $\mathcal{L}(D|\theta)$?
- ▶ If you have a simulator/forward model
 $\theta \rightarrow D$. defines an *implicit likelihood* \mathcal{L}
- ▶ Simulator generates samples from $\mathcal{L}(D|\theta)$.
- ▶ With a prior $\pi(\theta)$ can generate samples from joint distribution $\mathcal{J}(\theta, D) = \mathcal{L}(D|\theta)\pi(\theta)$
the “*probability of everything*”.
- ▶ Task of SBI is take joint \mathcal{J} samples and learn posterior $\mathcal{P}(\theta|D)$ and evidence $\mathcal{Z}(D)$
and possibly likelihood $\mathcal{L}(D|\theta)$.
- ▶ Present state of the art achieves this using *machine learning* (neural networks).
 - ▶ My group's research tries to removes machine learning github.com/handley-lab/lsbi.



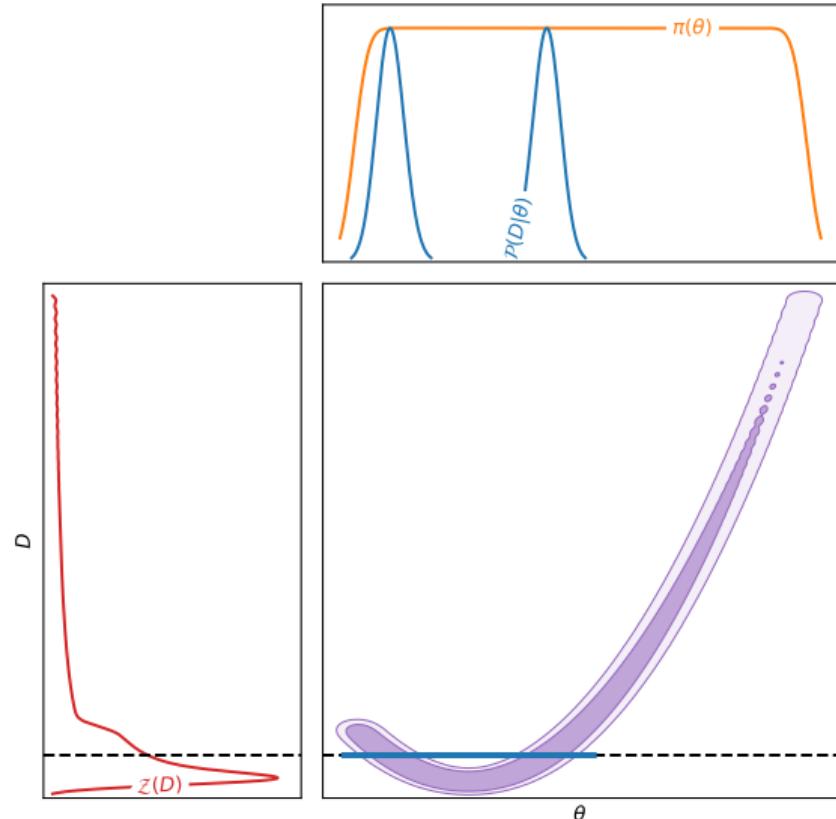
SBI: Simulation-based inference

- ▶ What do you do if you don't know $\mathcal{L}(D|\theta)$?
- ▶ If you have a simulator/forward model
 $\theta \rightarrow D$. defines an *implicit likelihood* \mathcal{L}
- ▶ Simulator generates samples from $\mathcal{L}(D|\theta)$.
- ▶ With a prior $\pi(\theta)$ can generate samples from joint distribution $\mathcal{J}(\theta, D) = \mathcal{L}(D|\theta)\pi(\theta)$
the “probability of everything”.
- ▶ Task of SBI is take joint \mathcal{J} samples and learn posterior $\mathcal{P}(\theta|D)$ and evidence $\mathcal{Z}(D)$
and possibly likelihood $\mathcal{L}(D|\theta)$.
- ▶ Present state of the art achieves this using *machine learning* (neural networks).
 - ▶ My group's research tries to remove machine learning github.com/handley-lab/lstbi.



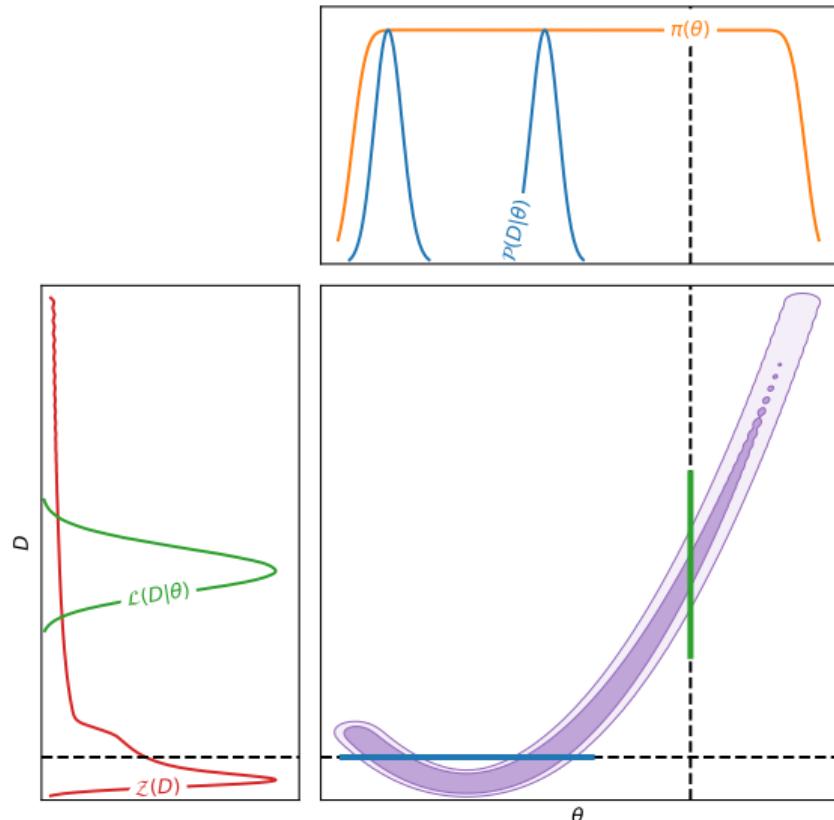
SBI: Simulation-based inference

- ▶ What do you do if you don't know $\mathcal{L}(D|\theta)$?
- ▶ If you have a simulator/forward model
 $\theta \rightarrow D$. defines an *implicit likelihood* \mathcal{L}
- ▶ Simulator generates samples from $\mathcal{L}(D|\theta)$.
- ▶ With a prior $\pi(\theta)$ can generate samples from joint distribution $\mathcal{J}(\theta, D) = \mathcal{L}(D|\theta)\pi(\theta)$
the “probability of everything”.
- ▶ Task of SBI is take joint \mathcal{J} samples and learn posterior $\mathcal{P}(\theta|D)$ and evidence $\mathcal{Z}(D)$ and possibly likelihood $\mathcal{L}(D|\theta)$.
- ▶ Present state of the art achieves this using *machine learning* (neural networks).
 - ▶ My group's research tries to removes machine learning github.com/handley-lab/lsbi.



SBI: Simulation-based inference

- ▶ What do you do if you don't know $\mathcal{L}(D|\theta)$?
- ▶ If you have a simulator/forward model
 $\theta \rightarrow D$. defines an *implicit likelihood* \mathcal{L}
- ▶ Simulator generates samples from $\mathcal{L}(D|\theta)$.
- ▶ With a prior $\pi(\theta)$ can generate samples from joint distribution $\mathcal{J}(\theta, D) = \mathcal{L}(D|\theta)\pi(\theta)$
the “*probability of everything*”.
- ▶ Task of SBI is take joint \mathcal{J} samples and learn posterior $\mathcal{P}(\theta|D)$ and evidence $\mathcal{Z}(D)$
and possibly likelihood $\mathcal{L}(D|\theta)$.
- ▶ Present state of the art achieves this using *machine learning* (neural networks).
 - ▶ My group's research tries to removes machine learning github.com/handley-lab/lsbi.



Why SBI?

SBI is useful because:

1. If you don't have a likelihood, you can still do inference
 - ▶ This is the usual case beyond CMB cosmology
2. Faster than LBI
 - ▶ emulation – also applies to LBI in principle
3. No need to pragmatically encode fiducial cosmologies
 - ▶ Covariance computation implicitly encoded in simulations
 - ▶ Highly relevant for disentangling tensions & systematics
4. Equips AI/ML with Bayesian interpretability
5. Lower barrier to entry than LBI
 - ▶ Much easier to forward model a systematic
 - ▶ Emerging set of plug-and-play packages
 - ▶ For this reason alone, it will come to dominate scientific inference

A screenshot of the GitHub repository page for 'sbi' (simulation-based inference). The page shows the README file, which contains Python code for a simulation-based inference example. The code includes imports, a function definition, and a call to a neural network for posterior sampling. The GitHub interface shows the repository's structure, including submodules like 'inference', 'likelihood', and 'likelihood_ratio'. The repository has over 100 stars and 1000 forks.

github.com/sbi-dev

A screenshot of the GitHub repository page for 'Swyft'. The page features a large logo and a brief description: 'Swyft is a system for scientific simulation-based inference at scale'. It includes sections for 'About', 'Installations', 'Examples', and 'Issues'. The repository has over 100 stars and 1000 forks.

github.com/undark-lab/swyft

A screenshot of the GitHub repository page for 'pyselfi'. The page shows the homepage with a navigation bar for 'HOME', 'SCIENTIFIC', 'ACADEMIC', 'BLOG', and 'ABOUT ME'. It features a 'Pyselfi' logo and a brief description: 'Pyselfi is a statistical inference package which implements the simulator-inference (SIBI) algorithm'. The repository has over 100 stars and 1000 forks.

github.com/florent-leclercq/pyselfi

A screenshot of the GitHub repository page for 'pydelfi'. The page shows the homepage with a navigation bar for 'HOME', 'ABOUT', 'Pydelfi', and 'Installation'. It features a 'Pydelfi' logo and a brief description: 'Density Estimation Likelihood-Free Inference with neural density estimators and adaptive acquisition of simulations. The implemented methods are described in detail in: [using Cholesky](#), [Huang et al. 2020](#) and [Huang et al. 2022](#)'. The repository has over 100 stars and 1000 forks.

github.com/justinalsing/pydelfi

Why aren't we currently using SBI in cosmology?

- ▶ Short answer: we are!
 - ▶ Mostly for weak lensing
 - ▶ 2024 has been the year it has started to be applied to real data.
- ▶ Longer answer: SBI requires mock data generation code
- ▶ Most data analysis codes were built before the generative paradigm.
- ▶ It's still a lot of work to upgrade cosmological likelihoods to be able to do this (e.g. plik & camspec).

Investigating the turbulent hot gas in X-COP galaxy clusters

S. Dupourqué¹, N. Clerc¹, E. Pointecouteau¹, D. Eckert², S. Ettori³, and F. Vazza^{4,5,6}

Dark Energy Survey Year 3 results: simulation-based cosmological inference with wavelet harmonics, scattering transforms, and moments of weak lensing mass maps II. Cosmological results

M. Gatti,^{1,*} G. Campailla,² N. Jeffrey,³ L. Whitney,³ A. Paredes,⁴ J. Prat,⁵ J. Williamson,³ M. Raveri,² B.

Neural Posterior Estimation with guaranteed exact coverage: the ringdown of GW150914

Marco Crisostomi^{1,2}, Kallol Dey³, Enrico Barausse^{1,2}, Roberto Trotta^{1,2,4,5}

Applying Simulation-Based Inference to Spectral and Spatial Information from the Galactic Center Gamma-Ray Excess

Katharena Christy,^a Eric J. Baxter,^b Jason Kumar^b

KIDS-1000 and DES-Y1 combined: Cosmology from peak count statistics

Joséchim Harnois-Déraps^{1*}, Sven Heydenreich², Benjamin Giblin³, Nicolas Martinet⁴,
Tilman Tröster⁵, Marika Asgari^{1,6,7}, Pierre Burger^{8,9,10}, Tiago Castro^{1,12,13,14},
Klaus Dolag¹⁵, Catherine Heymans^{3,16}, Hendrik Hildebrandt¹⁶, Benjamin Joachimi¹⁷ &
Angus H. Wright¹⁶

KiDS-SBI: Simulation-Based Inference Analysis of KiDS-1000 Cosmic Shear

Maximilian von Wietersheim-Kramata^{1,2,3}, Kiyam Lin⁴, Nicolas Tessore¹, Benjamin Joachimi¹, Arthur Lourenço^{4,5},
Robert Reischke^{6,7}, and Angus H. Wright¹

Simulation-based inference of deep fields: galaxy population model and redshift distributions

Beatrice Moser,^{a,1} Tomasz Kacprzak,^{a,b} Silvan Fischbacher,^a
Alexandre Refregier,^a Dominic Grimm,^a Luca Tortorelli^c

SmBiG: Cosmological Constraints using Simulation-Based Inference of Galaxy Clustering with Marked Power Spectra

ELENA MASSARA  ^{1,2,*}, CHANGHOON HAN  ², MICHAEL EICKENBERG ², SHERELY HO ³, JIAMIN HOU ²,
PABLO LEMOS ^{4,5}, CHIRAG MODI ^{4,6}, AZADEH MORADNEZHAD DEGHAT  ^{7,8,11}, LIAM PARKER ^{3,12} AND
BENOÎT RÉGALDO-SAINT BLANCARD 

Neural Ratio Estimation

- ▶ SBI flavours: github.com/sbi-dev/sbi

NPE Neural posterior estimation

NLE Neural likelihood estimation

NJE Neural joint estimation

NRE Neural ratio estimation

- ▶ NRE recap:

1. Generate joint samples $(\theta, D) \sim \mathcal{J}$

- ▶ *straightforward if you have a simulator:*
 $\theta \sim \pi(\cdot)$, $D \sim \mathcal{L}(\cdot | \theta)$

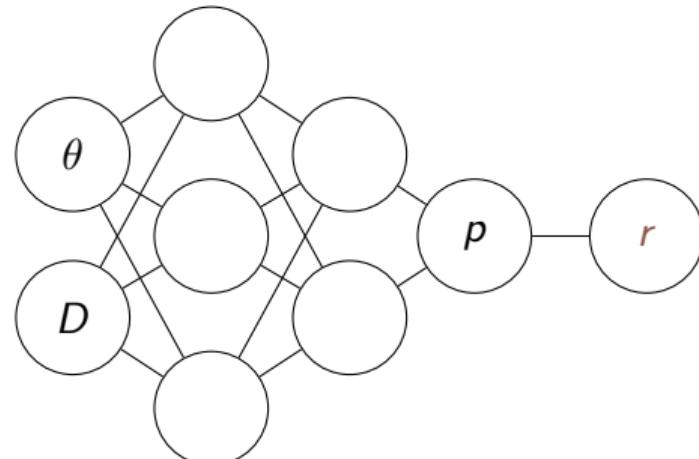
2. Generate separated samples $\theta \sim \pi$, $D \sim \mathcal{Z}$

- ▶ *aside: can shortcut step 2 by scrambling the (θ, D) pairings from step 1*

3. Train probabilistic classifier p to distinguish whether (θ, D) came from \mathcal{J} or $\pi \times \mathcal{Z}$.

4. $\frac{p}{1-p} = r = \frac{P(\theta, D)}{P(\theta)P(D)} = \frac{\mathcal{J}}{\pi \times \mathcal{Z}} = \frac{\mathcal{L}}{\mathcal{Z}} = \frac{\mathcal{P}}{\pi}$.

5. Use ratio r for parameter estimation $\mathcal{P} = r \times \pi$



Neural Ratio Estimation

- ▶ SBI flavours: github.com/sbi-dev/sbi

NPE Neural posterior estimation

NLE Neural likelihood estimation

NJE Neural joint estimation

NRE Neural ratio estimation

- ▶ NRE recap:

1. Generate joint samples $(\theta, D) \sim \mathcal{J}$

- ▶ *straightforward if you have a simulator:*
 $\theta \sim \pi(\cdot)$, $D \sim \mathcal{L}(\cdot | \theta)$

2. Generate separated samples $\theta \sim \pi$, $D \sim \mathcal{Z}$

- ▶ *aside: can shortcut step 2 by scrambling the (θ, D) pairings from step 1*

3. Train probabilistic classifier p to distinguish whether (θ, D) came from \mathcal{J} or $\pi \times \mathcal{Z}$.

4. $\frac{p}{1-p} = r = \frac{P(\theta, D)}{P(\theta)P(D)} = \frac{\mathcal{J}}{\pi \times \mathcal{Z}} = \frac{\mathcal{L}}{\mathcal{Z}} = \frac{\mathcal{P}}{\pi}$.

5. Use ratio r for parameter estimation $\mathcal{P} = r \times \pi$

Bayesian proof

- ▶ Let $M_{\mathcal{J}}$: $(\theta, D) \sim \mathcal{J}$, $M_{\pi \mathcal{Z}}$: $(\theta, D) \sim \pi \times \mathcal{Z}$

- ▶ Classifier gives

$$p(\theta, D) = P(M_{\mathcal{J}} | \theta, D) = 1 - P(M_{\pi \mathcal{Z}} | \theta, D)$$

- ▶ Bayes theorem then shows

$$\frac{p}{1-p} = \frac{P(M_{\mathcal{J}} | \theta, D)}{P(M_{\pi \mathcal{Z}} | \theta, D)} = \frac{P(\theta, D | M_{\mathcal{J}})P(M_{\mathcal{J}})}{P(\theta, D | M_{\pi \mathcal{Z}})P(M_{\pi \mathcal{Z}})} = \frac{\mathcal{J}}{\pi \mathcal{Z}},$$

where we have assumed

- ▶ $P(M_{\mathcal{J}}) = P(M_{\pi \mathcal{Z}})$,

and by definition

- ▶ $\mathcal{J}(D, \theta) = P(\theta, D | M_{\mathcal{J}})$

- ▶ $\pi(\theta)\mathcal{Z}(D) = P(\theta, D | M_{\pi \mathcal{Z}})$.

Neural Ratio Estimation

- SBI flavours: github.com/sbi-dev/sbi

NPE Neural posterior estimation

NLE Neural likelihood estimation

NJE Neural joint estimation

NRE Neural ratio estimation

- NRE recap:

1. Generate joint samples $(\theta, D) \sim \mathcal{J}$

- straightforward if you have a simulator:*

$$\theta \sim \pi(\cdot), D \sim \mathcal{L}(\cdot|\theta)$$

2. Generate separated samples $\theta \sim \pi, D \sim \mathcal{Z}$

- aside: can shortcut step 2 by scrambling the (θ, D) pairings from step 1*

3. Train probabilistic classifier p to distinguish whether (θ, D) came from \mathcal{J} or $\pi \times \mathcal{Z}$.

4. $\frac{p}{1-p} = r = \frac{P(\theta, D)}{P(\theta)P(D)} = \frac{\mathcal{J}}{\pi \times \mathcal{Z}} = \frac{\mathcal{L}}{\mathcal{Z}} = \frac{\mathcal{P}}{\pi}$.

5. Use ratio r for parameter estimation $\mathcal{P} = r \times \pi$

Why I like NRE

- The link between classification and inference is profound.
- Density estimation is hard – Dimensionless r divides out the hard-to-calculate parts.

Why I don't like NRE

- Practical implementations require marginalisation [[2107.01214](#)], or autoregression [[2308.08597](#)].
- Model comparison and parameter estimation are separate [[2305.11241](#)].

TMNRE: Truncated Marginal Neural Ratio Estimation

swyft: github.com/undark-lab/swyft

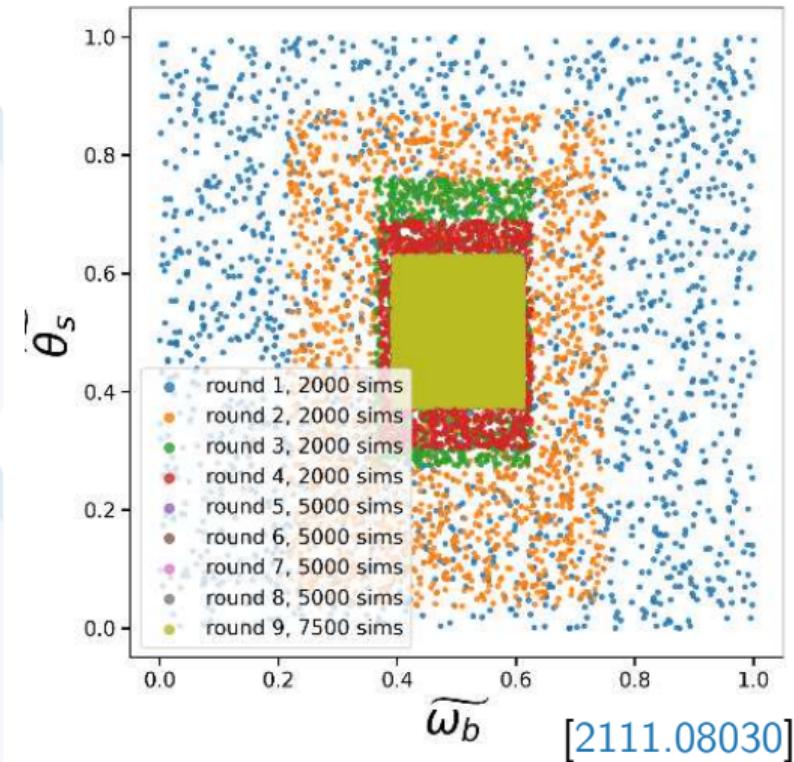
- ▶ Two tricks for practical NRE:

Marginalisation

- ▶ Only consider one or two parameters at a time.
- ▶ Fine if your goal is to produce triangle plots.
- ▶ Problematic if information is contained jointly in more than two parameters.

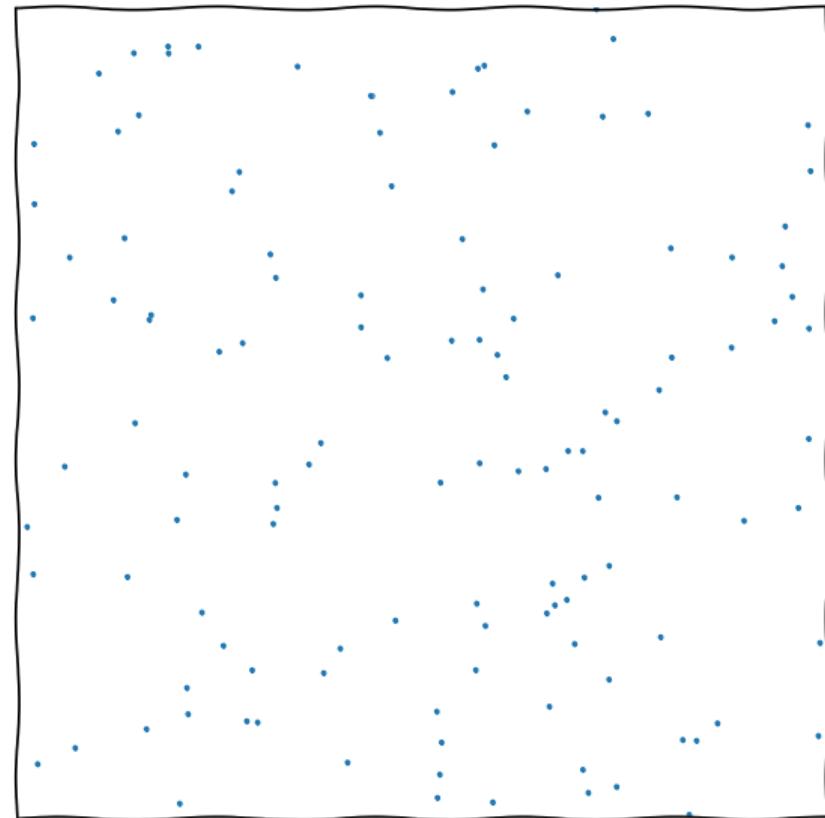
Truncation

- ▶ focus parameters θ on a subset of the prior which reproduces observed data D_{obs}
- ▶ region is somewhat arbitrary (usually a box)
- ▶ not amortised, sounds a bit like ABC



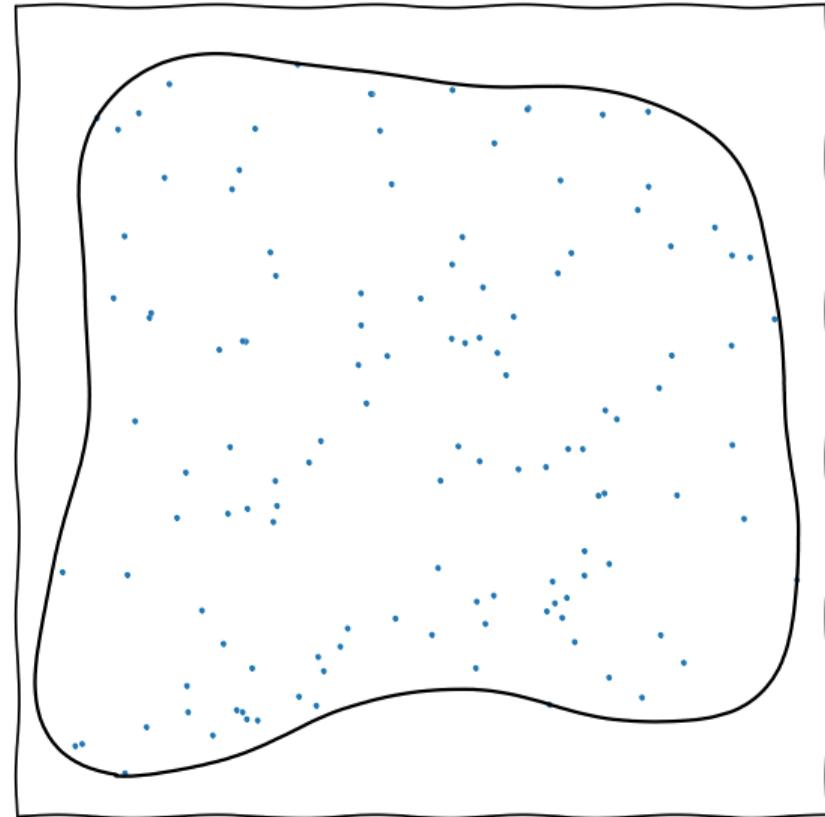
Nested sampling: numerical Lebesgue integration

0. Start with N random samples over the space.
 - i. Delete outermost sample, and replace with a new random one at higher integrand value.
- ▶ The “live points” steadily contract around the peak(s) of the function.
- ▶ Discarded “dead points” can be weighted to form posterior, prior, or anything in between.
- ▶ Estimates the **density of states** and calculates evidences & partition functions.
- ▶ The evolving ensemble of live points allows:
 - ▶ implementations to self-tune,
 - ▶ exploration of multimodal functions,
 - ▶ global and local optimisation.



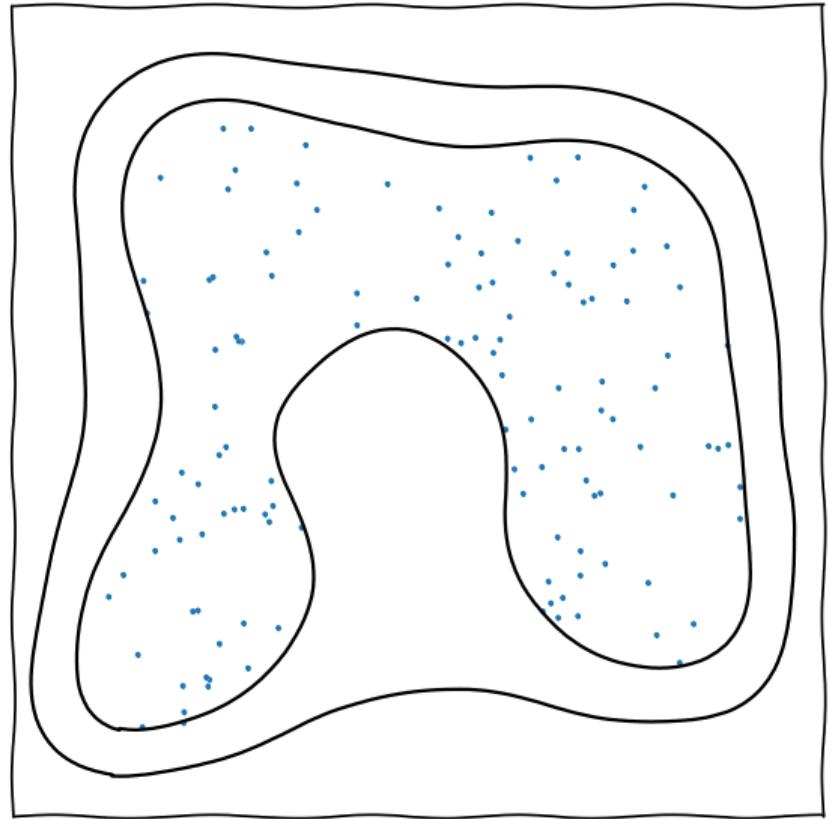
Nested sampling: numerical Lebesgue integration

0. Start with N random samples over the space.
 - i. Delete outermost sample, and replace with a new random one at higher integrand value.
- ▶ The “live points” steadily contract around the peak(s) of the function.
- ▶ Discarded “dead points” can be weighted to form posterior, prior, or anything in between.
- ▶ Estimates the **density of states** and calculates evidences & partition functions.
- ▶ The evolving ensemble of live points allows:
 - ▶ implementations to self-tune,
 - ▶ exploration of multimodal functions,
 - ▶ global and local optimisation.



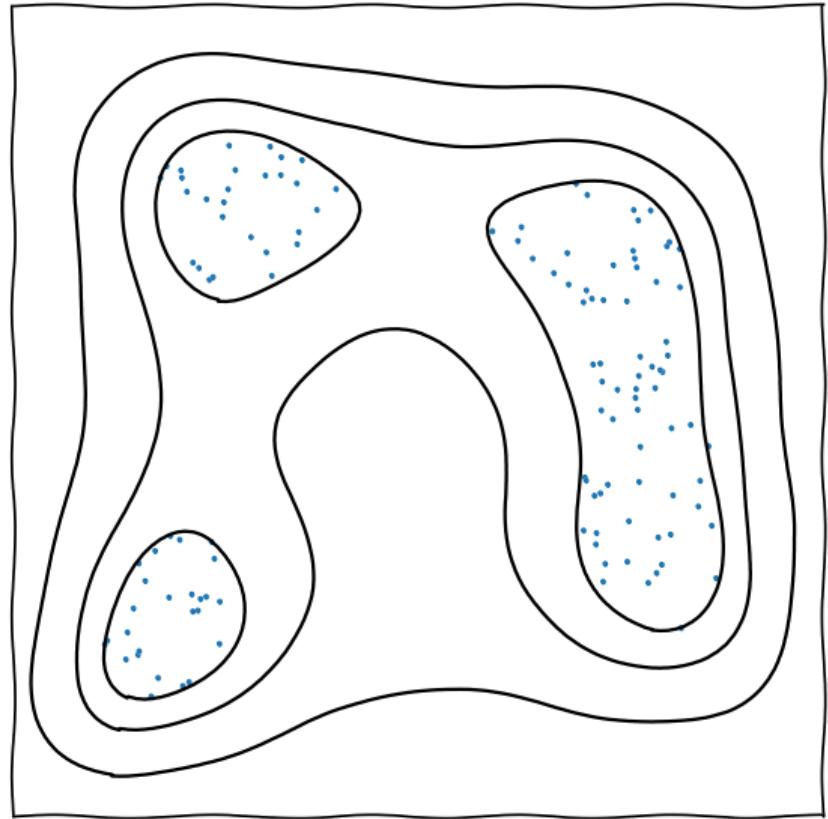
Nested sampling: numerical Lebesgue integration

0. Start with N random samples over the space.
 - i. Delete outermost sample, and replace with a new random one at higher integrand value.
- ▶ The “live points” steadily contract around the peak(s) of the function.
- ▶ Discarded “dead points” can be weighted to form posterior, prior, or anything in between.
- ▶ Estimates the **density of states** and calculates evidences & partition functions.
- ▶ The evolving ensemble of live points allows:
 - ▶ implementations to self-tune,
 - ▶ exploration of multimodal functions,
 - ▶ global and local optimisation.



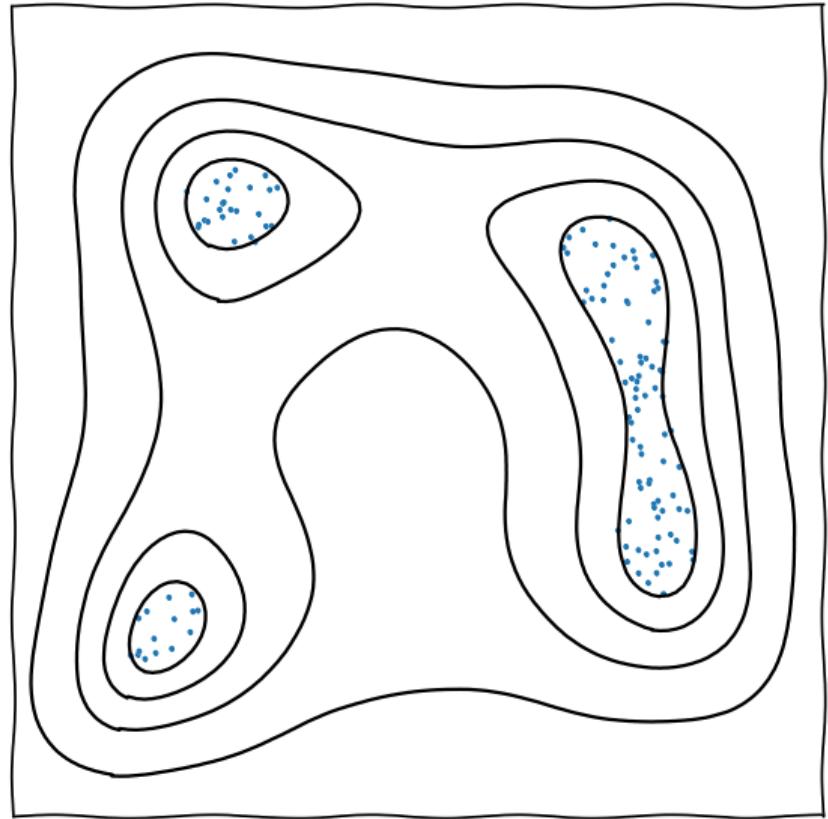
Nested sampling: numerical Lebesgue integration

0. Start with N random samples over the space.
 - i. Delete outermost sample, and replace with a new random one at higher integrand value.
- ▶ The “live points” steadily contract around the peak(s) of the function.
- ▶ Discarded “dead points” can be weighted to form posterior, prior, or anything in between.
- ▶ Estimates the **density of states** and calculates evidences & partition functions.
- ▶ The evolving ensemble of live points allows:
 - ▶ implementations to self-tune,
 - ▶ exploration of multimodal functions,
 - ▶ global and local optimisation.



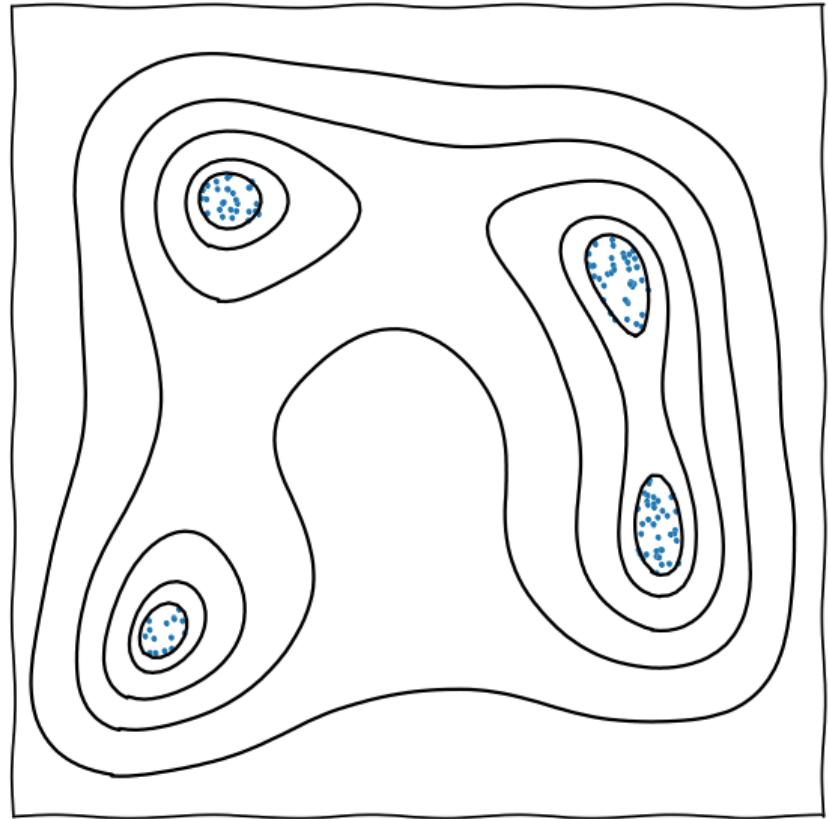
Nested sampling: numerical Lebesgue integration

0. Start with N random samples over the space.
 - i. Delete outermost sample, and replace with a new random one at higher integrand value.
- ▶ The “live points” steadily contract around the peak(s) of the function.
- ▶ Discarded “dead points” can be weighted to form posterior, prior, or anything in between.
- ▶ Estimates the **density of states** and calculates evidences & partition functions.
- ▶ The evolving ensemble of live points allows:
 - ▶ implementations to self-tune,
 - ▶ exploration of multimodal functions,
 - ▶ global and local optimisation.



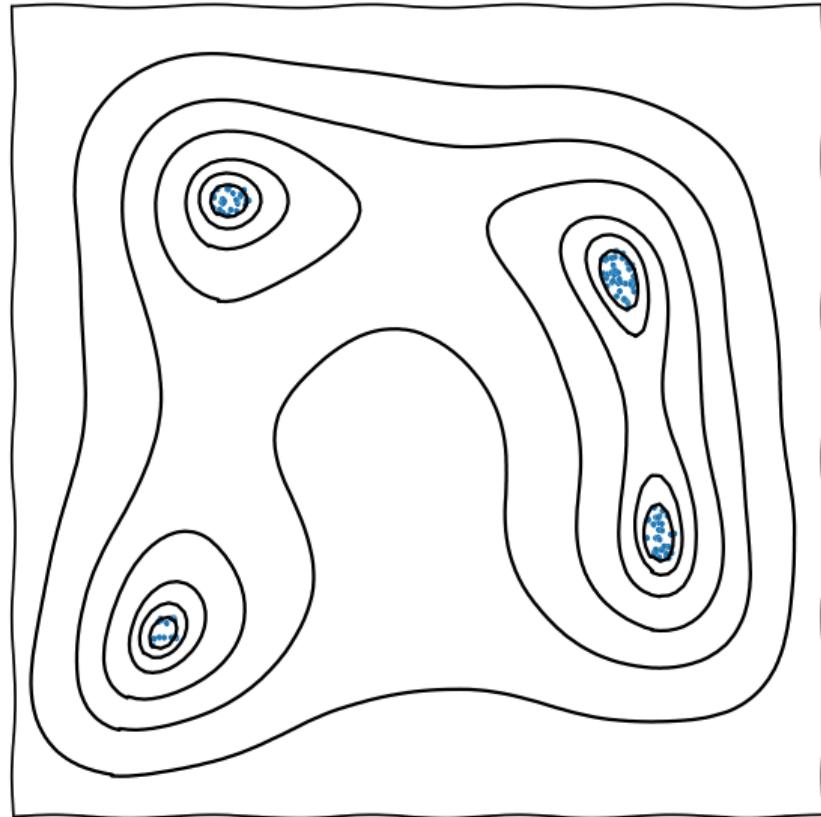
Nested sampling: numerical Lebesgue integration

0. Start with N random samples over the space.
 - i. Delete outermost sample, and replace with a new random one at higher integrand value.
- ▶ The “live points” steadily contract around the peak(s) of the function.
- ▶ Discarded “dead points” can be weighted to form posterior, prior, or anything in between.
- ▶ Estimates the **density of states** and calculates evidences & partition functions.
- ▶ The evolving ensemble of live points allows:
 - ▶ implementations to self-tune,
 - ▶ exploration of multimodal functions,
 - ▶ global and local optimisation.



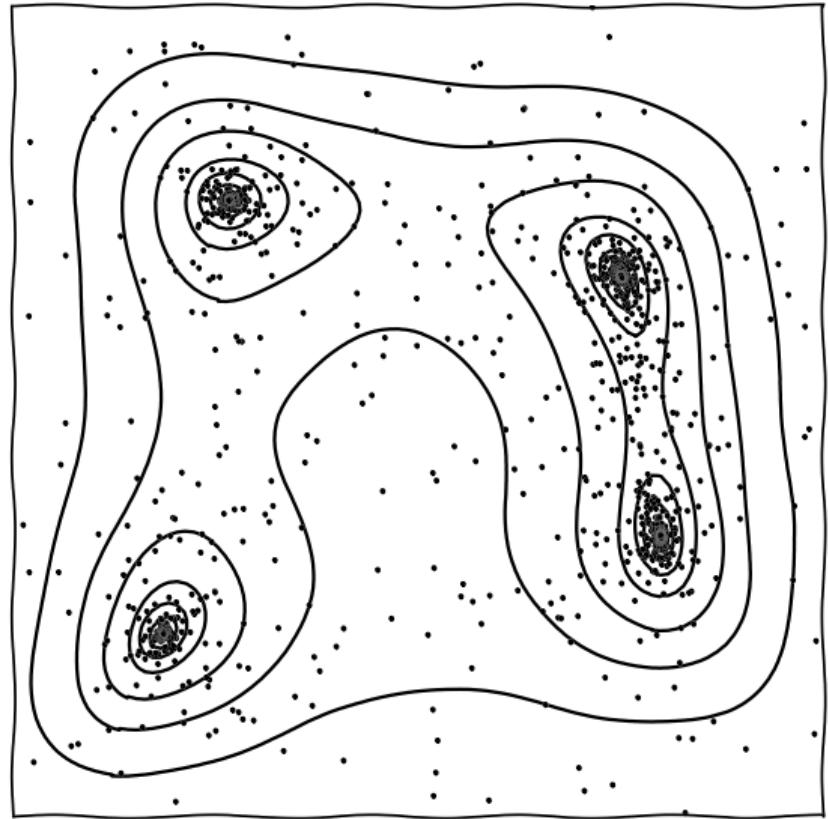
Nested sampling: numerical Lebesgue integration

0. Start with N random samples over the space.
 - i. Delete outermost sample, and replace with a new random one at higher integrand value.
- ▶ The “live points” steadily contract around the peak(s) of the function.
- ▶ Discarded “dead points” can be weighted to form posterior, prior, or anything in between.
- ▶ Estimates the **density of states** and calculates evidences & partition functions.
- ▶ The evolving ensemble of live points allows:
 - ▶ implementations to self-tune,
 - ▶ exploration of multimodal functions,
 - ▶ global and local optimisation.



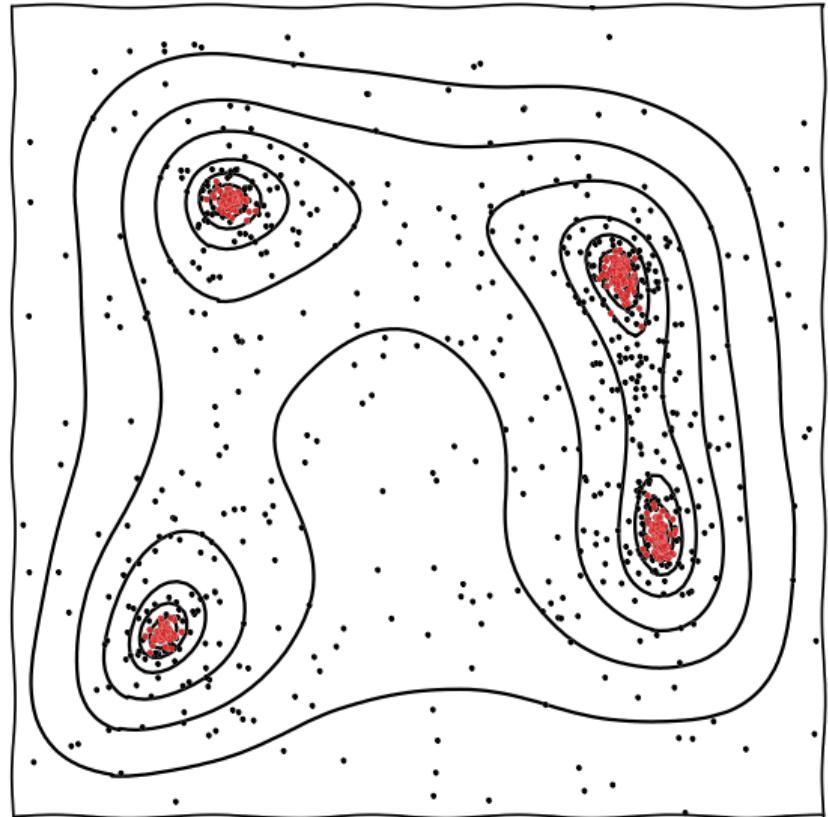
Nested sampling: numerical Lebesgue integration

0. Start with N random samples over the space.
 - i. Delete outermost sample, and replace with a new random one at higher integrand value.
- ▶ The “live points” steadily contract around the peak(s) of the function.
- ▶ Discarded “dead points” can be weighted to form posterior, prior, or anything in between.
- ▶ Estimates the **density of states** and calculates evidences & partition functions.
- ▶ The evolving ensemble of live points allows:
 - ▶ implementations to self-tune,
 - ▶ exploration of multimodal functions,
 - ▶ global and local optimisation.



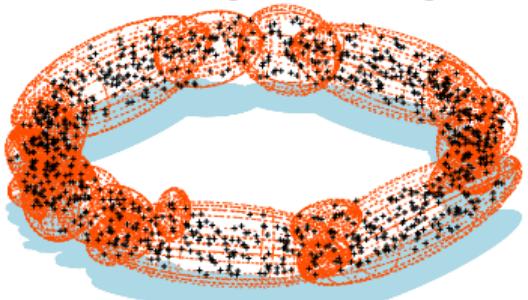
Nested sampling: numerical Lebesgue integration

0. Start with N random samples over the space.
 - i. Delete outermost sample, and replace with a new random one at higher integrand value.
- ▶ The “live points” steadily contract around the peak(s) of the function.
- ▶ Discarded “dead points” can be weighted to form posterior, prior, or anything in between.
- ▶ Estimates the **density of states** and calculates evidences & partition functions.
- ▶ The evolving ensemble of live points allows:
 - ▶ implementations to self-tune,
 - ▶ exploration of multimodal functions,
 - ▶ global and local optimisation.

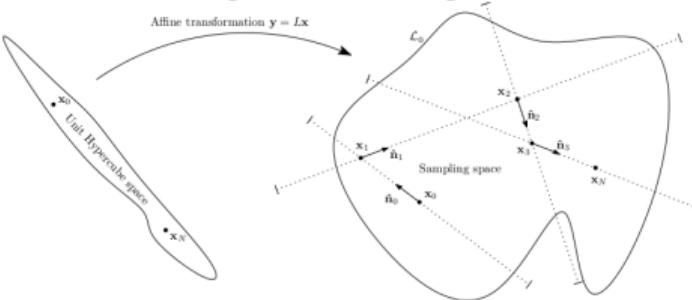


Implementations of Nested Sampling [2205.15570](NatReview)

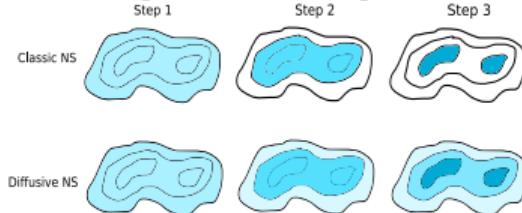
MultiNest [0809.3437]



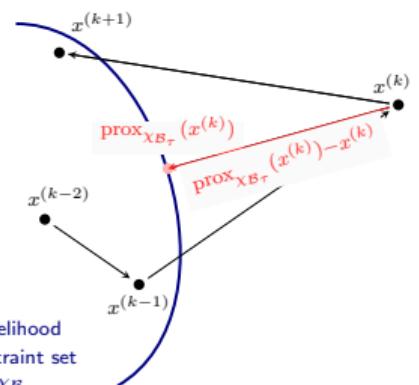
PolyChord [1506.00171]



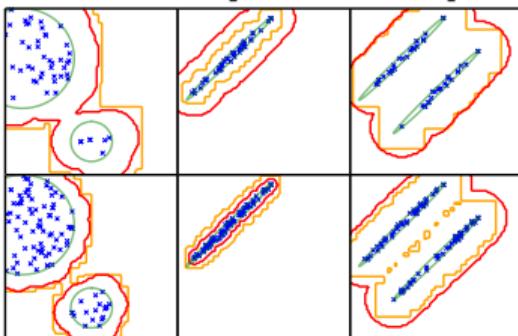
DNest [1606.03757]



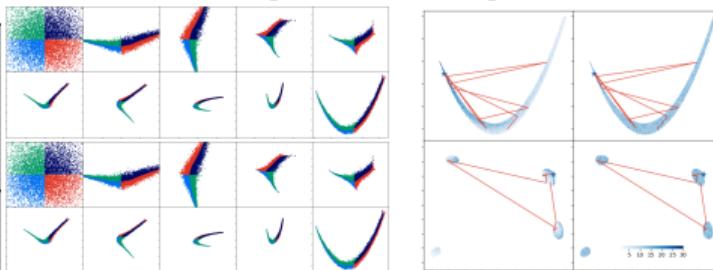
ProxNest [2106.03646]



UltraNest [2101.09604]



NeuralNest [1903.10860]



nessai [2102.11056]

nora [2305.19267]

jaxnest [2012.15286]

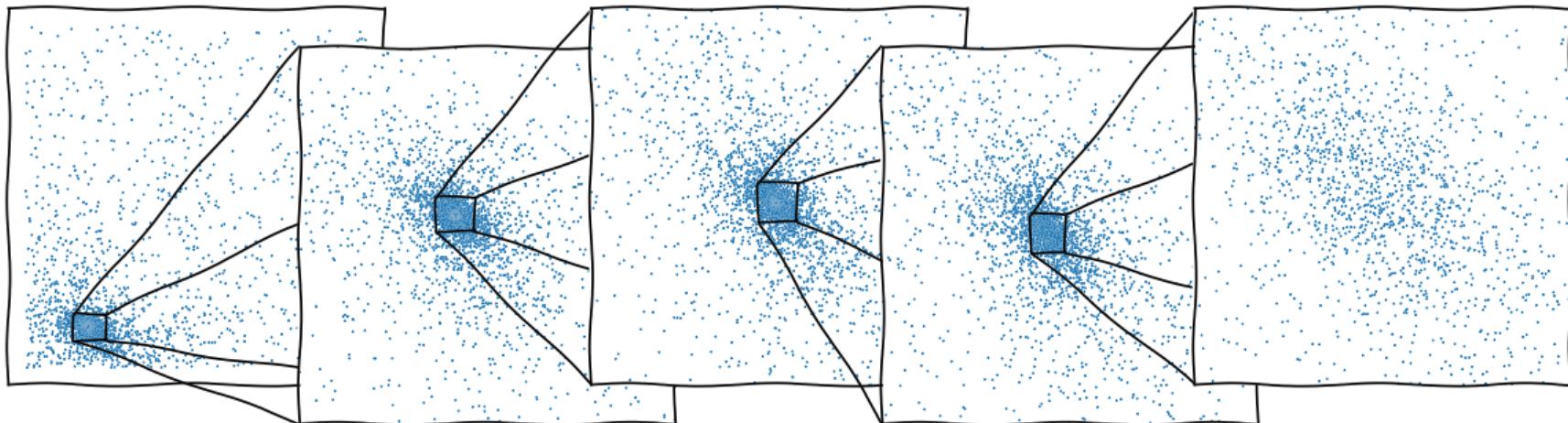
nautilus [2306.16923]

<wh260@cam.ac.uk>

willhandley.co.uk/talks

dynesty [1904.02180]

The nested sampling meta-algorithm: dead points

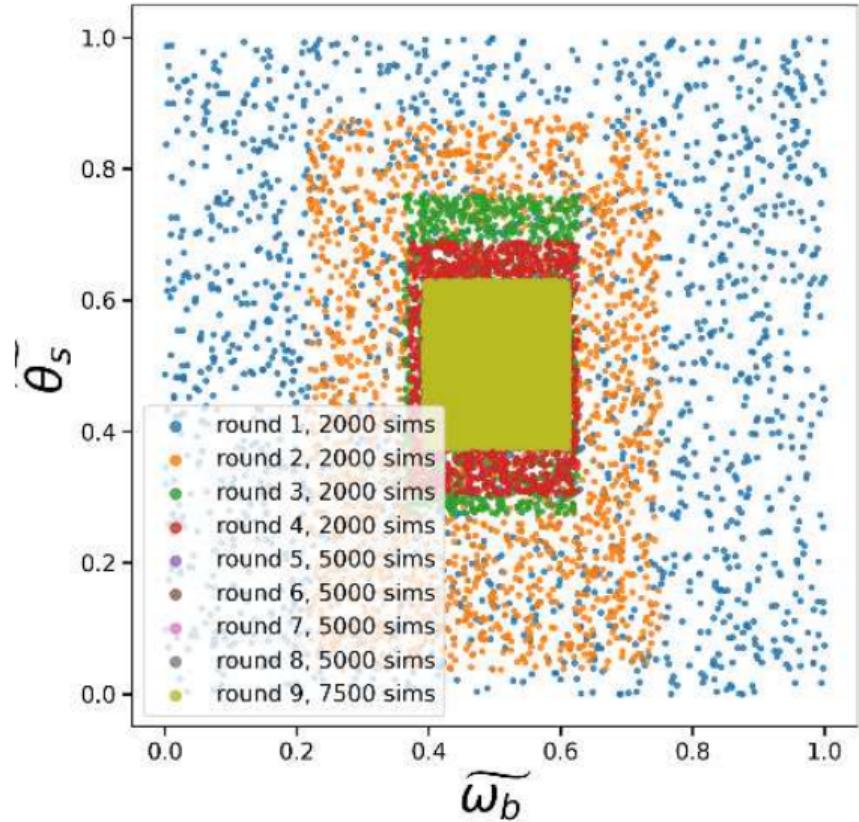
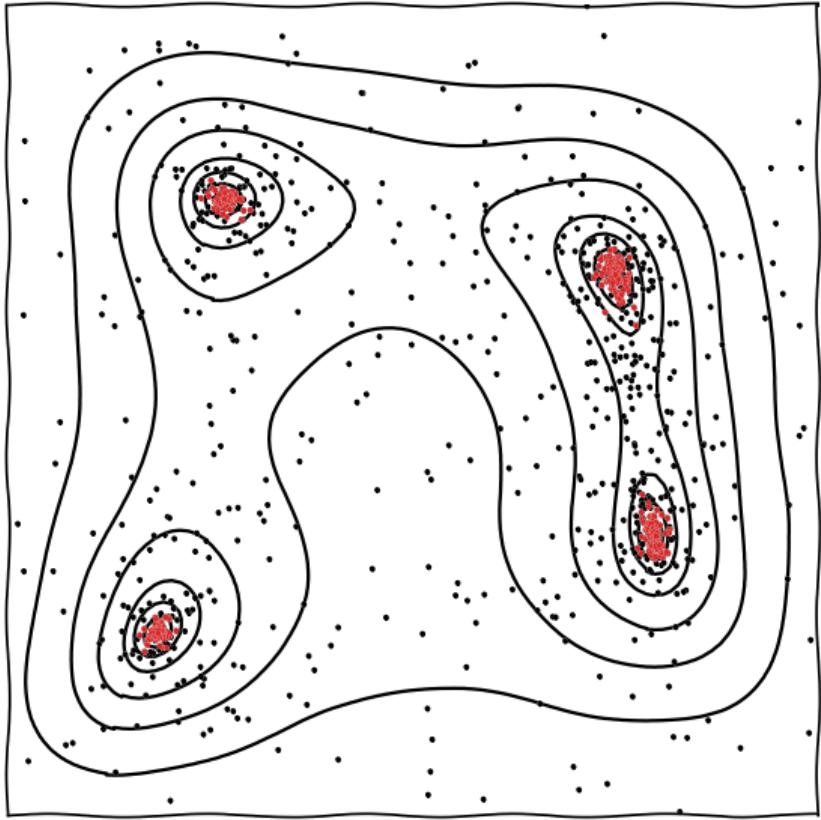


- ▶ At the end, one is left with a set of discarded “dead” points.
- ▶ Dead points have a unique scale-invariant distribution $\propto \frac{dV}{V}$.
- ▶ Uniform over original region, exponentially concentrating on region of interest (until termination volume).
- ▶ Good for training emulators (HERA [[2108.07282](#)]).

Applications

- ▶ training emulators.
- ▶ gridding simulations
- ▶ beta flows
- ▶ “dead measure”

Similarities



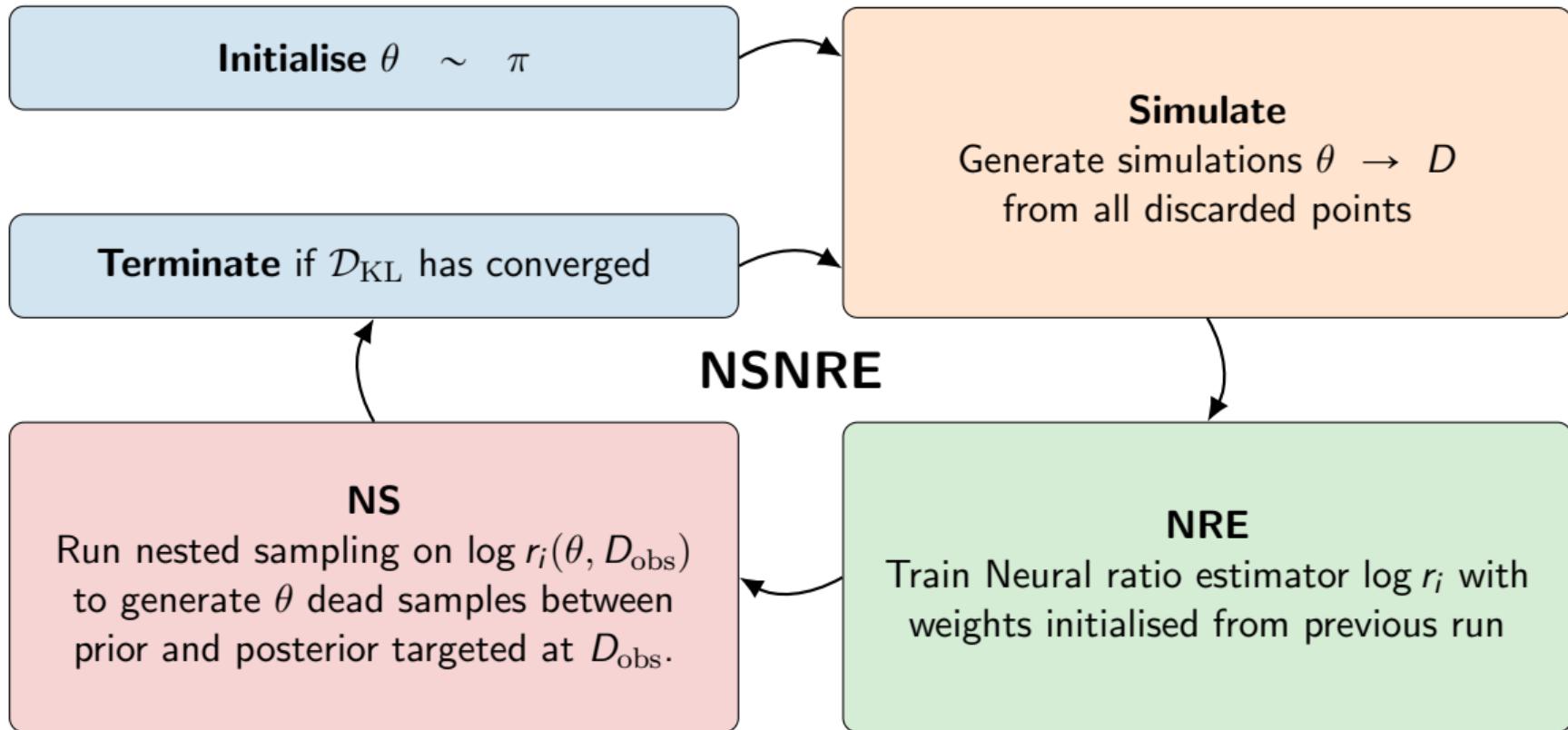
Why it's hard to do SBI with nested sampling

- ▶ At each iteration i , nested sampling requires you to be able to generate a new live point from the prior, subject to a hard likelihood constraint

$$\theta \sim \pi : \mathcal{L}(\theta) > \mathcal{L}_i$$

- ▶ This is hard if you don't have a likelihood!
- ▶ In addition, nested sampling does not do well if the likelihood is non-deterministic
- ▶ Previous attempts:
 - ▶ DNest paper [1606.03757](Section 10: Nested sampling for ABC)
 - ▶ ANRE [2308.08597] using non-box priors driven by current ratio estimate with slice sampling re-population.

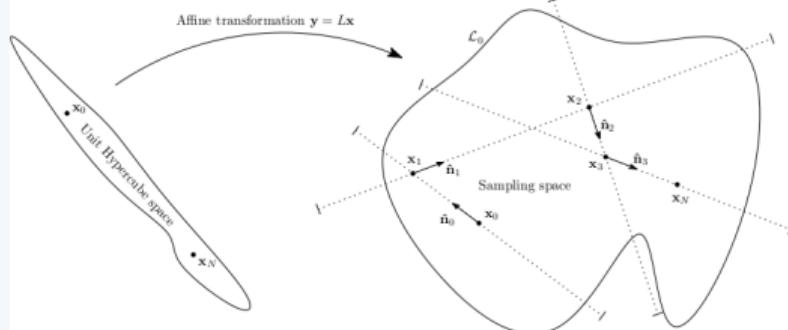
Sequential NRE with nested sampling



PolySwyft

PolyChord

github.com/PolyChord/PolyChordLite



- ▶ Widely used high-performance nested sampling tool (implementing slice sampling & clustering in MPI Fortran)

Swyft

github.com/undark-lab/swyft

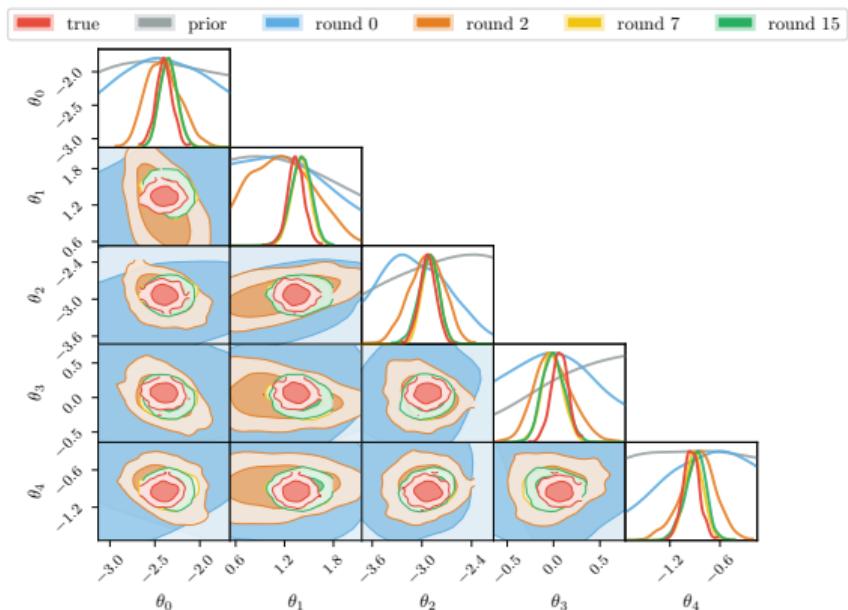


- ▶ Widely used TMNRE tool in cosmology/astrophysics.

However, NSNRE is general, and not specific to these choices.

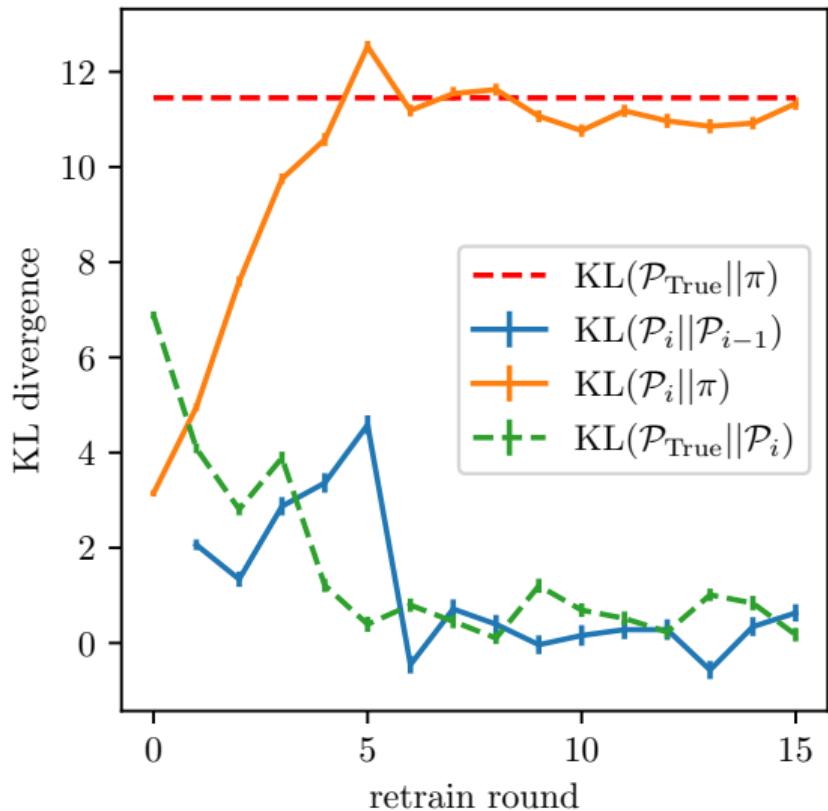
Convergence diagnostics

- ▶ Example for a $n = 5$ dimensional parameter space, with $d = 100$ data points, (lsbi gaussian mixture model).
- ▶ This is the regime for cosmological scale problems.
- ▶ To determine convergence we track:
 - ▶ The change in KL divergence between rounds (blue), and check when this goes to zero.
 - ▶ The total KL divergence between prior and posterior estimate (orange), and check when this levels off (ground truth in red).
 - ▶ Also shown is the KL divergence between the estimate and the ground truth (green).



Convergence diagnostics

- ▶ Example for a $n = 5$ dimensional parameter space, with $d = 100$ data points, (lsbi gaussian mixture model).
- ▶ This is the regime for cosmological scale problems.
- ▶ To determine convergence we track:
 - ▶ The change in KL divergence between rounds (blue), and check when this goes to zero.
 - ▶ The total KL divergence between prior and posterior estimate (orange), and check when this levels off (ground truth in red).
 - ▶ Also shown is the KL divergence between the estimate and the ground truth (green).



Conclusions

github.com/handley-lab



- ▶ PolySwyft can perform NRE on $n \sim 6$ parameter spaces and $d \sim 100$ data spaces.
- ▶ This makes it relevant for cosmological applications.
- ▶ Look out for imminent paper (post Kilian's thesis hand-in in $\sim \mathcal{O}(1\text{month})$)
- ▶ Examples produced using lsbi package: github.com/handley-lab/lsbi



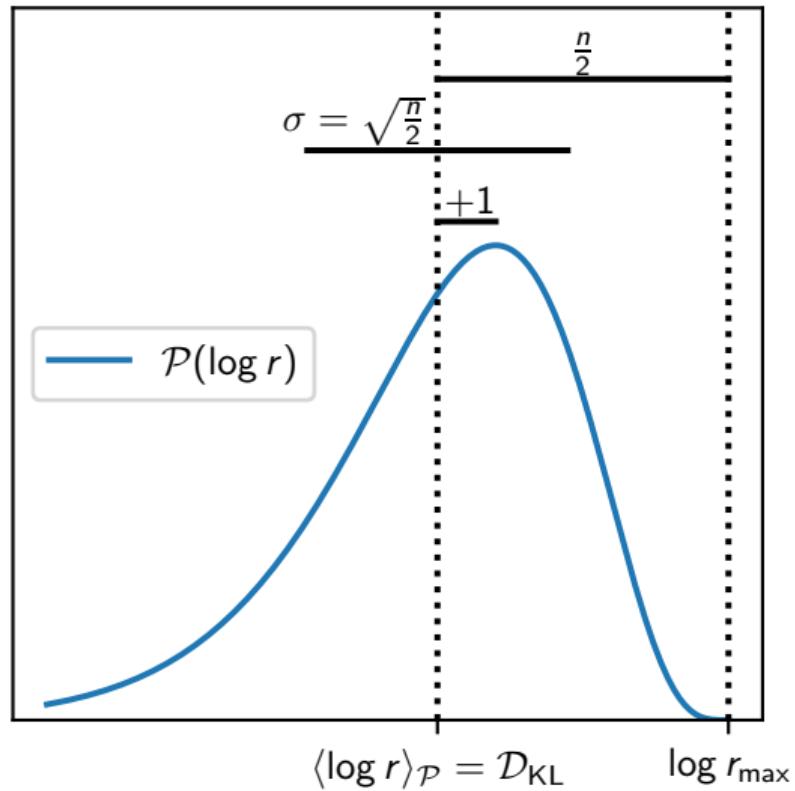
Considerations of ratio estimation

- ▶ Neural REs can in practice only estimate in a band of $\log r$ before the activation function saturates (typically $-5 < \log r < 5$).
- ▶ Consider a posterior \mathcal{P} well approximated by a Gaussian profile in an n -dimensional parameter space [2312.00294]
- ▶ If $\mathcal{D}_{\text{KL}} \gg 1$ between prior and posterior:

$$\log r = \frac{n}{2} + \mathcal{D}_{\text{KL}} + \chi_n^2$$

$$\langle \log r \rangle_{\mathcal{P}} = \mathcal{D}_{\text{KL}}, \quad \sigma(\log r)_{\mathcal{P}} = \sqrt{\frac{n}{2}}$$

- ▶ Truncation (**TMNRE**) reduces \mathcal{D}_{KL} , focusing the distribution into the $[-5, 5]$ band.
- ▶ Marginalisation (**TMNRE**) reduces n & σ .



Cosmological forecasting

Have you ever done a Fisher forecast, and then felt Bayesian guilt?

- ▶ Cosmologists are interested in forecasting what a Bayesian analysis of future data might produce.
- ▶ Useful for:
 - ▶ white papers/grants,
 - ▶ optimising existing instruments/strategies,
 - ▶ picking theory/observation to explore next.
- ▶ To do this properly:
 1. start from current knowledge $\pi(\theta)$, derived from current data
 2. Pick potential dataset D that might be collected from $P(D)$ ($= \mathcal{Z}$)
 3. Derive posterior $P(\theta|D)$
 4. Summarise science (e.g. constraint on θ , ability to perform model comparison)
- ▶ This procedure should be marginalised over:
 1. All possible parameters θ (consistent with prior knowledge)
 2. All possible data D
- ▶ i.e. marginalised over the joint $P(\theta, D) = P(D|\theta)P(\theta)$.
- ▶ Historically this has proven very challenging.
- ▶ Most analyses assume a fiducial cosmology θ_* , and/or a Gaussian likelihood/posterior (c.f. Fisher forecasting).
- ▶ This runs the risk of biasing forecasts by baking in a given theory/data realisation.

Fully Bayesian Forecasting [2309.06942]

Thomas Gessey-Jones



PhD

- ▶ Simulation based inference gives us the language to marginalise over parameters θ and possible future data D .
- ▶ Evidence networks give us the ability to do this at scale for forecasting [2305.11241].
- ▶ Demonstrated in 21cm global experiments, marginalising over:
 - ▶ theoretical uncertainty
 - ▶ foreground uncertainty
 - ▶ systematic uncertainty
- ▶ Able to say “at 67mK radiometer noise”, have a 50% chance of 5σ Bayes factor detection.
- ▶ Can use to optimise instrument design
- ▶ Re-usable package: prescience

