REACH: Nested sampling tools

Will Handley
<wh260@cam.ac.uk>

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

26th September 2023


What do we use nested sampling for?

Given a (scalar) function f with a vector of parameters x, nested sampling can be used for:

Optimisation

$$x_{\max} = \max_x f(x)$$

Experimental design

Exploration

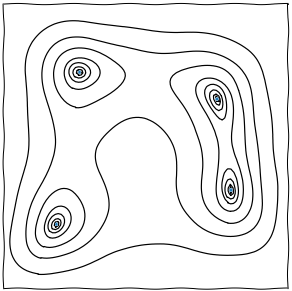
draw/sample $x \sim f$

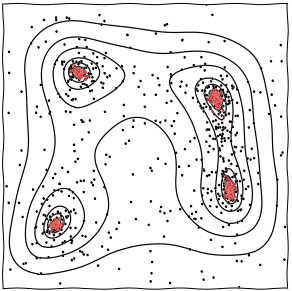
Parameter estimation

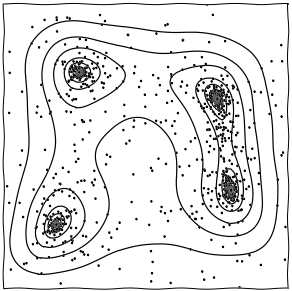
Integration

$$\int f(x) dV$$

Model comparison







The three pillars of Bayesian inference

Parameter estimation

What do the data tell us about the parameters of a model? *e.g. the central frequency ν_0 of Gaussian global signal*

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

$$\mathcal{P} = \frac{\mathcal{L} \times \pi}{\mathcal{Z}}$$

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

Model comparison

How much does the data support a particular model? *e.g. ΛCDM vs cosmic strings*

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

$$\frac{\mathcal{Z}_{\mathcal{M}}\Pi_{\mathcal{M}}}{\sum_m \mathcal{Z}_m \Pi_m}$$

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

Tension quantification

Do different datasets make consistent predictions from the same model? *e.g. SARAS vs EDGES*

$$\mathcal{R} = \frac{\mathcal{Z}_{AB}}{\mathcal{Z}_A \mathcal{Z}_B}$$

$$\log \mathcal{S} = \langle \log \mathcal{L}_{AB} \rangle_{\mathcal{P}_{AB}} - \langle \log \mathcal{L}_A \rangle_{\mathcal{P}_A} - \langle \log \mathcal{L}_B \rangle_{\mathcal{P}_B}$$

- Note, REACH has been a powerful ambassador for "model" involving theory, systematics and inference parameters [2204.04491]

Integration in Physics

- Integration is a fundamental concept in physics, statistics and data science:

Partition functions

$$Z(\beta) = \int e^{-\beta H(q,p)} dq dp$$

Path integrals

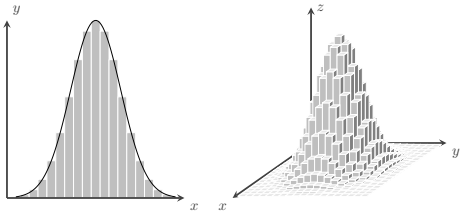
$$\Psi = \int e^{iS} \mathcal{D}x$$

Bayesian marginals

$$\mathcal{Z}(D) = \int \mathcal{L}(D|\theta)\pi(\theta) d\theta$$

- Need numerical tools if analytic solution unavailable.
- High-dimensional numerical integration is hard.
- Riemannian strategy estimates volumes geometrically:

$$\int f(x) d^n x \approx \sum_i f(x_i) \Delta V_i \sim \mathcal{O}(e^n)$$

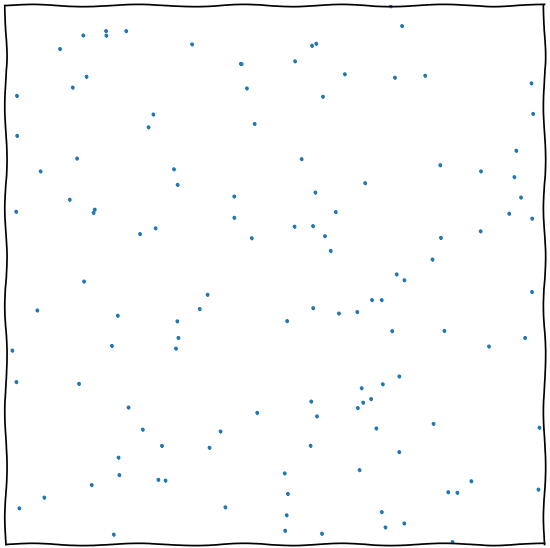


- Curse of dimensionality $\Rightarrow$ exponential scaling.

The nested sampling meta-algorithm: live points

- Start with n random samples over the space.
- 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.
- We can use this evolution to estimate volume *probabilistically*.
- At each iteration, the contours contract by $\sim \frac{1}{n}$ of their volume.
- This is an exponential contraction, so

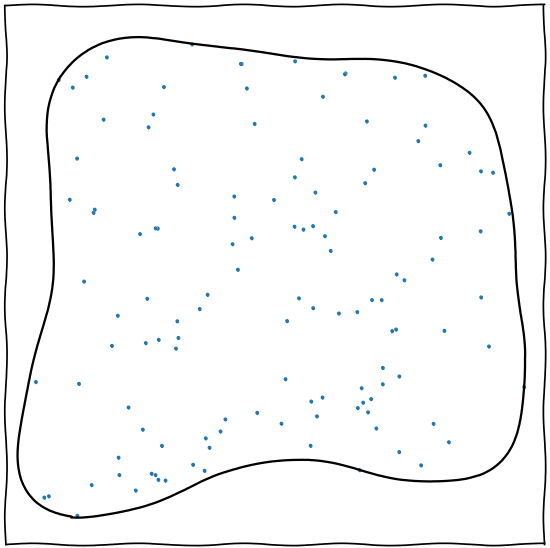
$$\int f(x)dV \approx \sum_i f(x_i)\Delta V_i, \quad V_i = V_0 e^{-i/n}$$



The nested sampling meta-algorithm: live points

- Start with n random samples over the space.
- 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.
- We can use this evolution to estimate volume *probabilistically*.
- At each iteration, the contours contract by $\sim \frac{1}{n}$ of their volume.
- This is an exponential contraction, so

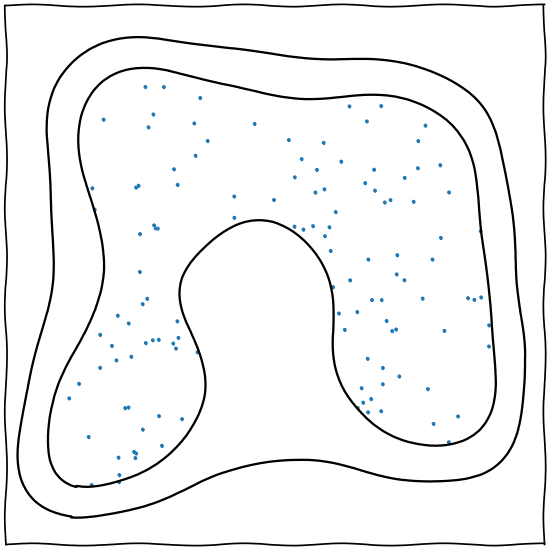
$$\int f(x)dV \approx \sum_i f(x_i)\Delta V_i, \quad V_i = V_0 e^{-i/n}$$



The nested sampling meta-algorithm: live points

- Start with n random samples over the space.
- 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.
- We can use this evolution to estimate volume *probabilistically*.
- At each iteration, the contours contract by $\sim \frac{1}{n}$ of their volume.
- This is an exponential contraction, so

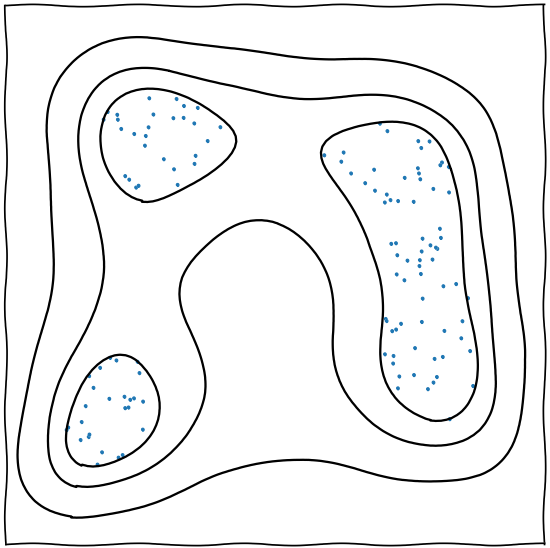
$$\int f(x)dV \approx \sum_i f(x_i)\Delta V_i, \quad V_i = V_0 e^{-i/n}$$



The nested sampling meta-algorithm: live points

- Start with n random samples over the space.
- 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.
- We can use this evolution to estimate volume *probabilistically*.
- At each iteration, the contours contract by $\sim \frac{1}{n}$ of their volume.
- This is an exponential contraction, so

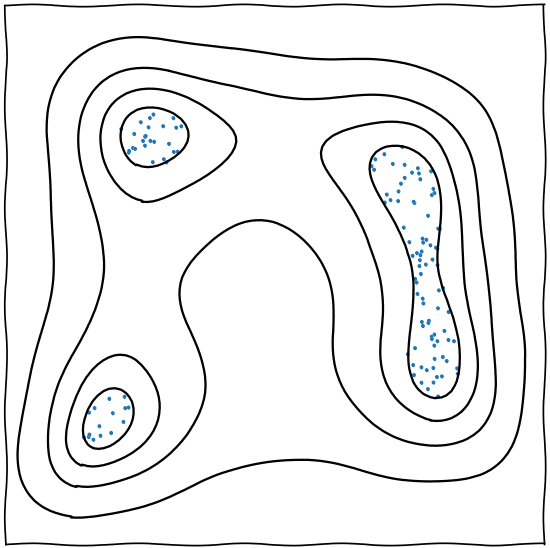
$$\int f(x)dV \approx \sum_i f(x_i)\Delta V_i, \quad V_i = V_0 e^{-i/n}$$



The nested sampling meta-algorithm: live points

- Start with n random samples over the space.
- 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.
- We can use this evolution to estimate volume *probabilistically*.
- At each iteration, the contours contract by $\sim \frac{1}{n}$ of their volume.
- This is an exponential contraction, so

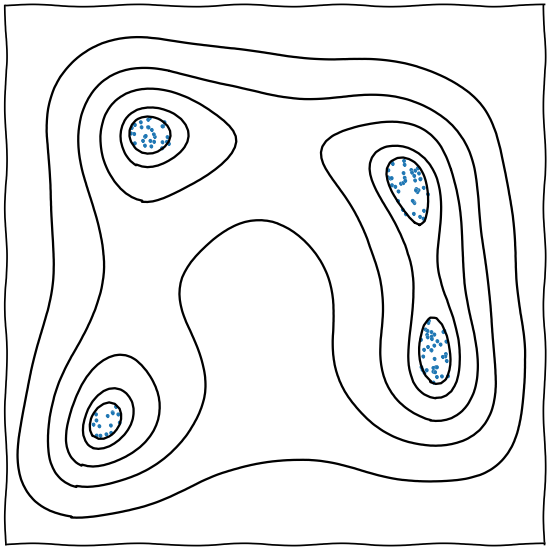
$$\int f(x)dV \approx \sum_i f(x_i)\Delta V_i, \quad V_i = V_0 e^{-i/n}$$



The nested sampling meta-algorithm: live points

- Start with n random samples over the space.
- 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.
- We can use this evolution to estimate volume *probabilistically*.
- At each iteration, the contours contract by $\sim \frac{1}{n}$ of their volume.
- This is an exponential contraction, so

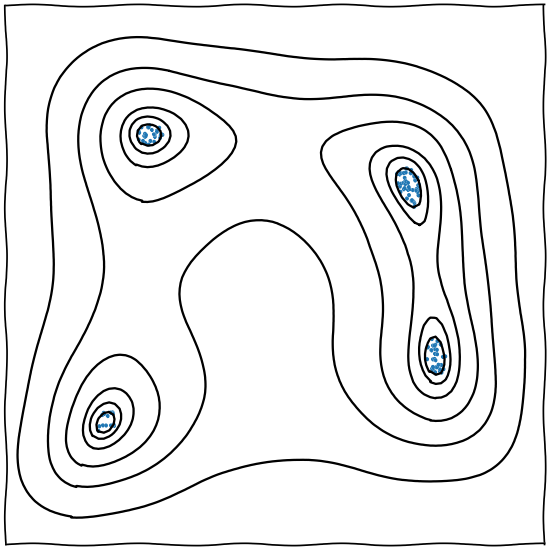
$$\int f(x)dV \approx \sum_i f(x_i)\Delta V_i, \quad V_i = V_0 e^{-i/n}$$



The nested sampling meta-algorithm: live points

- Start with n random samples over the space.
- 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.
- We can use this evolution to estimate volume *probabilistically*.
- At each iteration, the contours contract by $\sim \frac{1}{n}$ of their volume.
- This is an exponential contraction, so

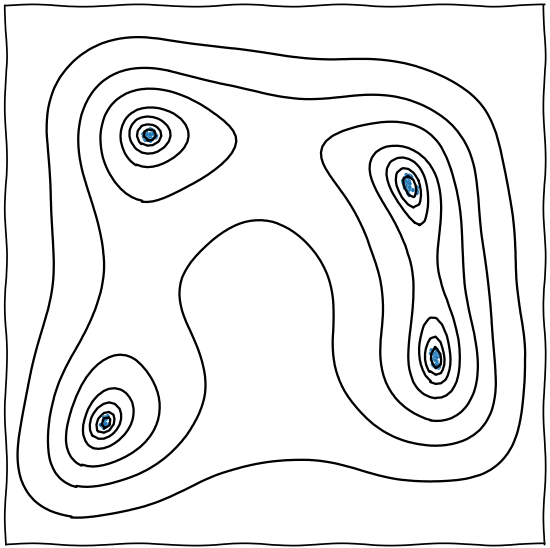
$$\int f(x)dV \approx \sum_i f(x_i)\Delta V_i, \quad V_i = V_0 e^{-i/n}$$



The nested sampling meta-algorithm: live points

- Start with n random samples over the space.
- 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.
- We can use this evolution to estimate volume *probabilistically*.
- At each iteration, the contours contract by $\sim \frac{1}{n}$ of their volume.
- This is an exponential contraction, so

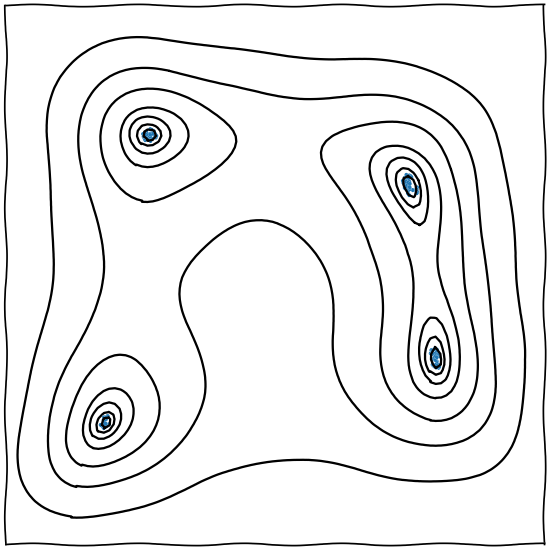
$$\int f(x)dV \approx \sum_i f(x_i)\Delta V_i, \quad V_i = V_0 e^{-i/n}$$



The nested sampling meta-algorithm: live points

- Start with n random samples over the space.
- 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.
- We can use this evolution to estimate volume *probabilistically*.
- At each iteration, the contours contract by $\sim \frac{1}{n} \pm \frac{1}{n}$ of their volume.
- This is an exponential contraction, so

$$\int f(x)dV \approx \sum_i f(x_i)\Delta V_i, \quad V_i = V_0 e^{-(i \pm \sqrt{i})/n}$$

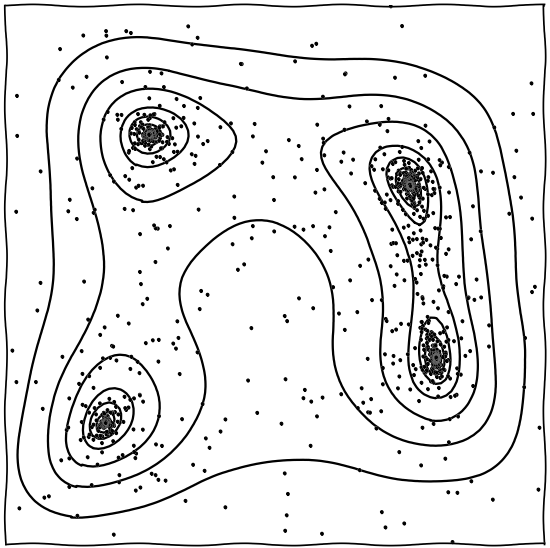


The nested sampling meta-algorithm: dead points

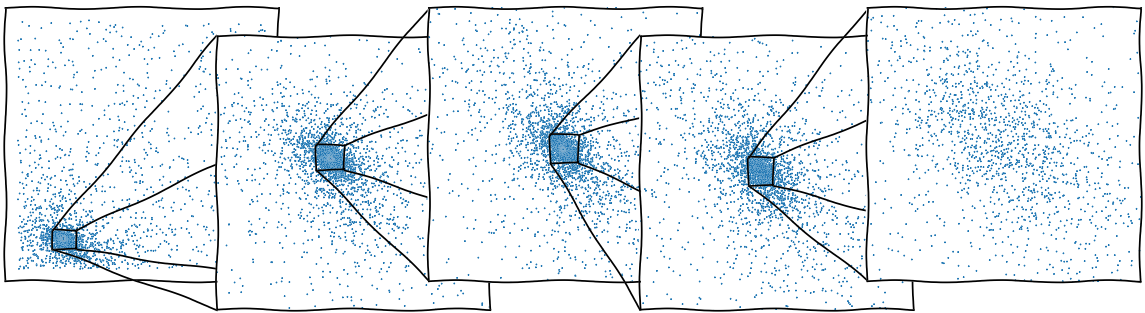
- At the end, one is left with a set of discarded "dead" points.
- Can be weighted to form posterior samples, prior samples, or anything in between.
- Nested sampling estimates the **density of states** and calculates partition functions

$$Z(\beta) = \sum_i f(x_i)^\beta \Delta V_i.$$

- The evolving ensemble of live points allows:
  - implementations to self-tune
  - exploration of multimodal functions
  - global and local optimisation



The dead measure



- 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).
- Full coverage of tails enables integration.
- Good for training emulators (HERA [2108.07282]).

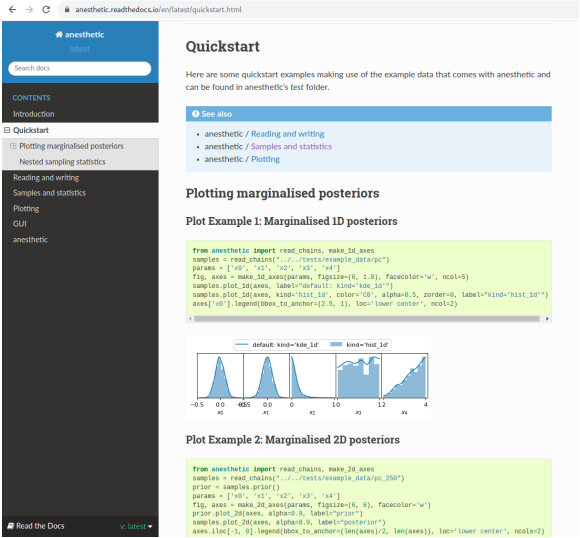
Applications

- training emulators.
- gridding simulations
- beta flows...
- good name for a band

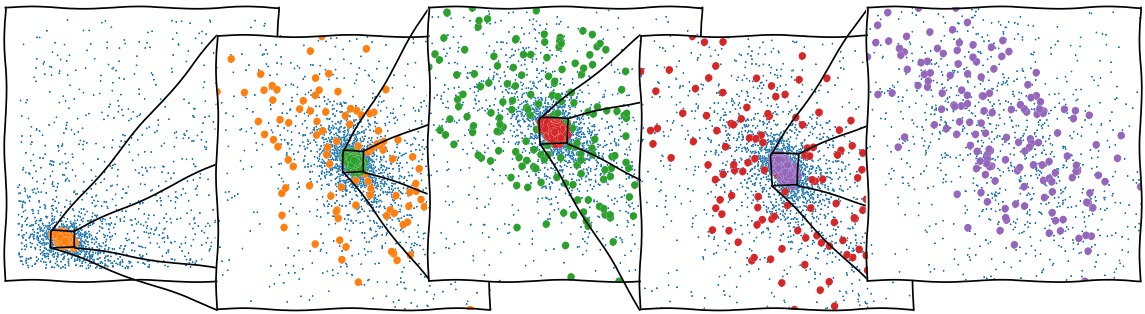
anesthetic: the nested sampling toolkit

Adam Ormondroyd & Lukas Hergt
PhD

- Post-processing nested sampling calculations requires care (e.g. weighted samples)
- `anesthetic` does this all for you
- pip installable (`pip install anesthetic`)
- docs: `anesthetic.readthedocs.io`
- nested sampling analysis (merging, sampling, weighting)
- dynamic replaying of nested sampling
- plotting
- built on `pandas` & `matplotlib` (extending to weighted dataframes)



Replaying nested sampling



- Can extract live points at iteration i from dead points
- Can extract posterior at any temperature
- Try out GUI for dynamically replaying run:
  - `$ anesthetic <file root> #bash`
  - `>>> samples.gui() #python`

Applications

- diagnosing run failures
- training emulators
- multiobjective optimisation

anesthetic vs getdist vs corner.py

- anesthetic
  - Defaults optimised for nested sampling
  - Colour choice better for priors
  - Not much smoothing
- getdist
  - State-of-the-art smoothing & edge correction
  - Often over-smoothed
- corner.py
  - Good for MCMC histograms
  - Can't do overlapping plots
  - Gotcha: Slightly different "2σ" convention

README.rst

Questions/Comments

Another posterior plotting tool?

> This is my posterior plotter. There are many like it, but this one is mine.

There are several excellent tools for plotting marginalised posteriors:

- getdist
- corner
- pygtc
- dynesty
- MontePython

Why create another one? In general, any dedicated user of software will find that there is some functionality that in their use case is lacking, and the designs of previous codes make such extensions challenging. In my case this was:

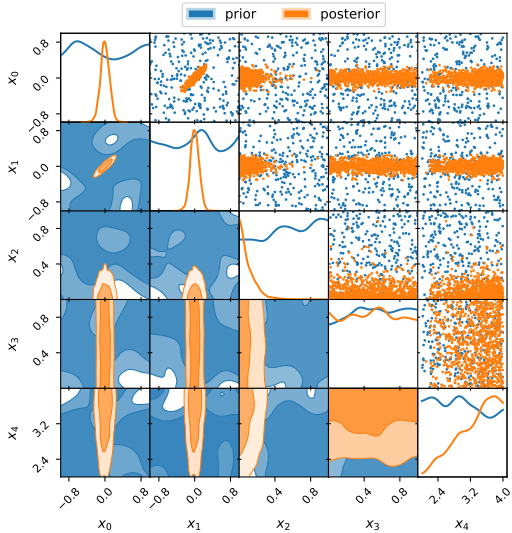
1. For large numbers of samples, kernel density estimation is slow, or inaccurate (particularly for samples generated from nested sampling). There are kernel density estimators, such as fastKDE, which ameliorate many of these difficulties.
2. Existing tools can make it difficult to define new parameters. For example, the default cosmomc chain defines `omegabh2`, but not `omegab`. The transformation is easy, since `omegab = omegabh2/ (H0/100)**2`, but implementing this transformation in existing packages is not so trivial. `anesthetic` solves this issue by storing the samples as a pandas array, for which the relevant code for defining the above new parameter would be

```
from anesthetic import read_chains
samples = read_chains(file_root)              # Load the samples
label = 'omegab'
tex = '$\Omega_b$'
h = (samples.H0/100)
samples[(label, tex)] = samples.omegabh2/h**2  # Define omegab
samples.plot_1d('omegab')                      # Simple 1D plot
```

3. Many KDE plotting tools have conventions that don't play well with uniformly distributed parameters, which presents a problem if you are trying to plot priors along with your posteriors. `anesthetic` has a sensible mechanism, by defining the contours by the amount of iso-probability mass they contain, but colouring the fill in relation to the probability density of the contour.

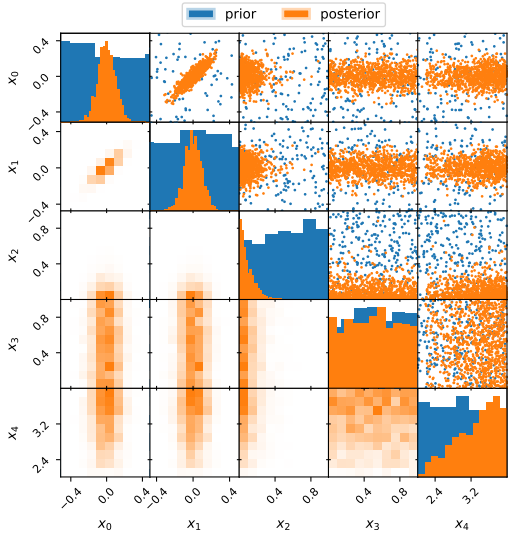
anesthetic vs getdist vs corner.py

- anesthetic
  - Defaults optimised for nested sampling
  - Colour choice better for priors
  - Not much smoothing
- getdist
  - State-of-the-art smoothing & edge correction
  - Often over-smoothed
- corner.py
  - Good for MCMC histograms
  - Can't do overlapping plots
  - Gotcha: Slightly different "2σ" convention



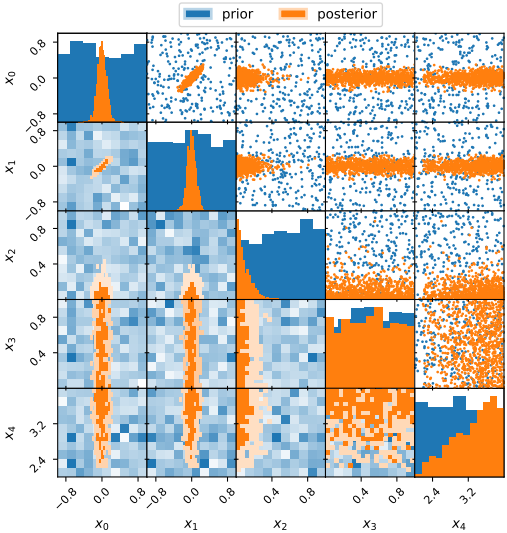
anesthetic vs getdist vs corner.py

- anesthetic
  - Defaults optimised for nested sampling
  - Colour choice better for priors
  - Not much smoothing
- getdist
  - State-of-the-art smoothing & edge correction
  - Often over-smoothed
- corner.py
  - Good for MCMC histograms
  - Can't do overlapping plots
  - Gotcha: Slightly different "2σ" convention



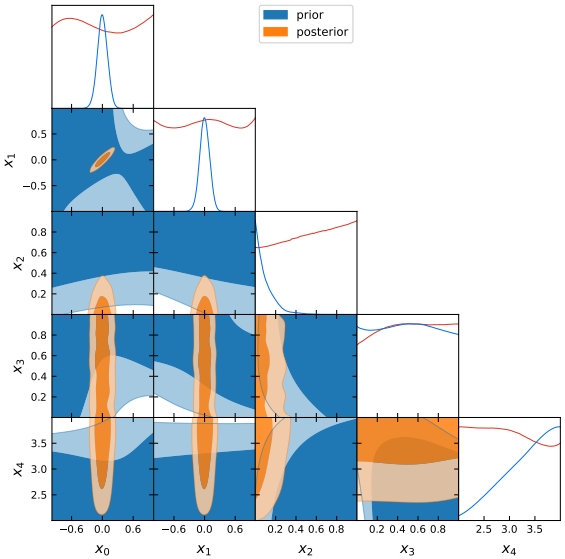
anesthetic vs getdist vs corner.py

- anesthetic
  - Defaults optimised for nested sampling
  - Colour choice better for priors
  - Not much smoothing
- getdist
  - State-of-the-art smoothing & edge correction
  - Often over-smoothed
- corner.py
  - Good for MCMC histograms
  - Can't do overlapping plots
  - Gotcha: Slightly different "2σ" convention



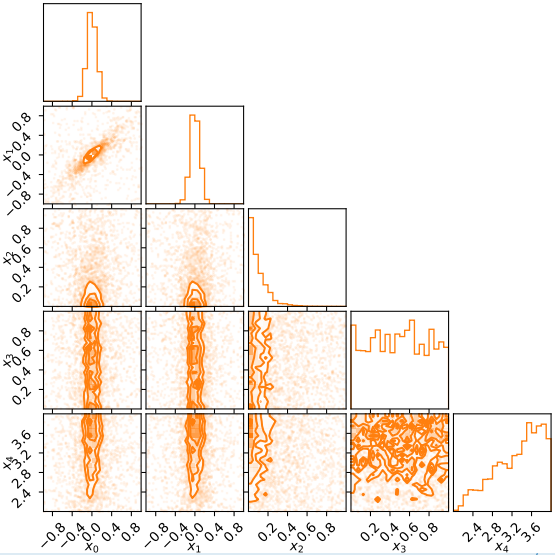
anesthetic vs getdist vs corner.py

- anesthetic
  - Defaults optimised for nested sampling
  - Colour choice better for priors
  - Not much smoothing
- getdist
  - State-of-the-art smoothing & edge correction
  - Often over-smoothed
- corner.py
  - Good for MCMC histograms
  - Can't do overlapping plots
  - Gotcha: Slightly different "2σ" convention



anesthetic vs getdist vs corner.py

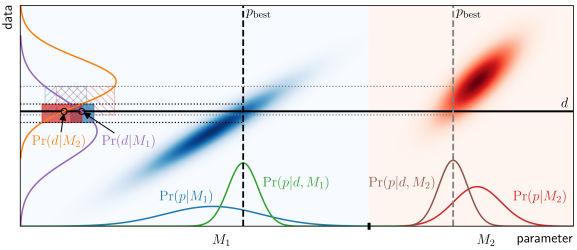
- anesthetic
  - Defaults optimised for nested sampling
  - Colour choice better for priors
  - Not much smoothing
- getdist
  - State-of-the-art smoothing & edge correction
  - Often over-smoothed
- corner.py
  - Good for MCMC histograms
  - Can't do overlapping plots
  - Gotcha: Slightly different "2σ" convention

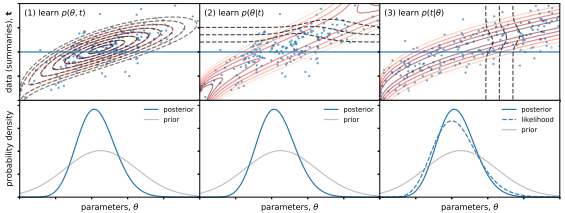


Likelihood-free inference (aka SBI)

Kilian Scheutwinkel
PhD

- How do you do inference if you don't know the likelihood $P(D|\theta)$?
- If you can forward simulate/model $\theta \to D$, then you have an implicit likelihood.
- LFI aims to (machine-)*learn* the likelihood from forward simulations $\{(\theta, D)\}$.
- Current REACH-related work
  - Anchal `swyft`
  - Kilian `PolySwyft`: likelihood-free nested sampling
- In my view this is the frontier of REACH data analysis
- We have a rudimentary version of this with work on "likelihood selection" [2204.04491]

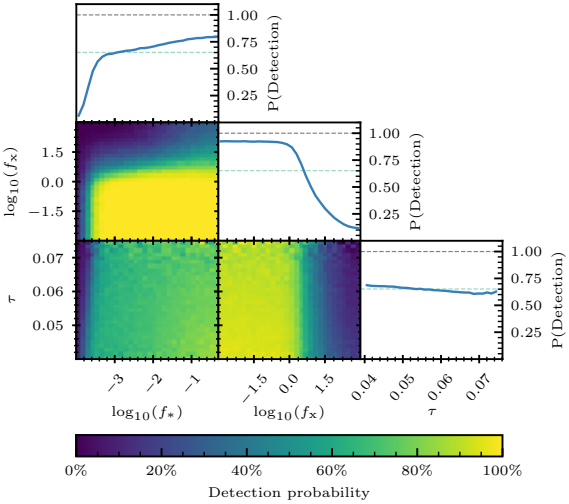




Fully Bayesian Forecasting [2309.06942]

Thomas Gessey-Jones
PhD

- Experimental design necessitates forecasting the constraints that future data might give.
- Have you ever done a Fisher forecast, and then felt Bayesian guilt?
- Simulation based inference gives us the language to marginalise over parameters θ and possible future data D.
- Evidence networks [2305.11241] (Jeffreys & Wandelt) give us the ability to do this at scale in the case of forecasting.
- Can answer questions such as "Given current knowledge/theoretical uncertainty $\pi(\theta)$, how probable is a detection?"
- Re-usable package: `prescience`



Future work/discussion points

- {multiobjective,}{antenna,}{optimisation} (John)
- Fully Bayesian Forecasts (Thomas)
- Simulation based inference (Kilian & Anchal)
- margarine (Harry)
- High-dimensional gradient-based nested sampling (Sam)