# High dimensional nested sampling

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 January 2023



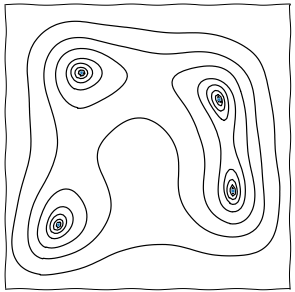


# What is Nested Sampling?

- Nested sampling is a radical, multi-purpose numerical tool.
- Given a (scalar) function $f$ with a vector of parameters $\theta$, it can be used for:

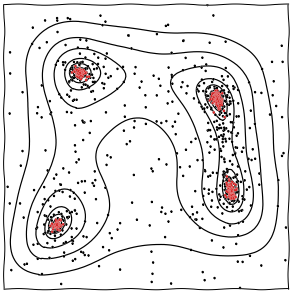
## Optimisation

$$\theta_{\max} = \max_{\theta} f(\theta)$$



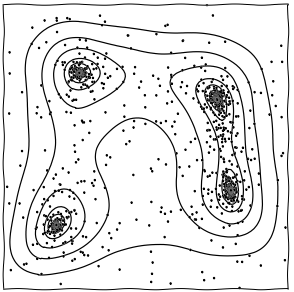
## Exploration

draw/sample $\theta \sim f$



## Integration

$$\int f(\theta) dV$$



# The three pillars of Bayesian inference

Where NS has been used historically

## Parameter estimation

What do the data tell us about the parameters of a model? *e.g. the size or age of a ΛCDM universe*

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

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

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

## Model comparison

How much does the data support a particular model? *e.g. ΛCDM vs a dynamic dark energy cosmology*

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

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

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

## Tension quantification

Do different datasets make consistent predictions from the same model? *e.g. CMB vs Type IA supernovae data*

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

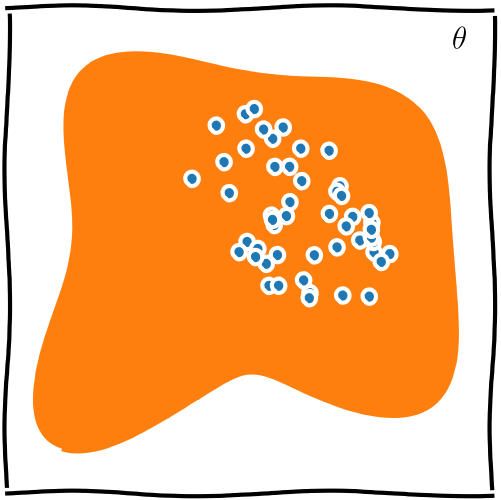
$$\log \mathcal{S} = \langle \log \mathcal{L}_{AB} \rangle_{\mathcal{P}_{AB}} - \langle \log \mathcal{L}_A \rangle_{\mathcal{P}_A} - \langle \log \mathcal{L}_B \rangle_{\mathcal{P}_B}$$

# Why do sampling?

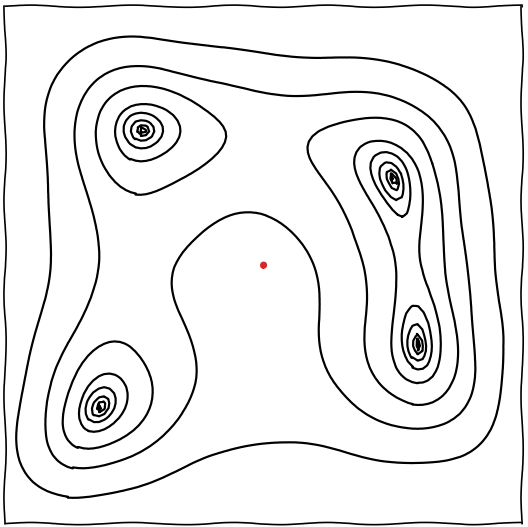
- The cornerstone of numerical Bayesian inference is working with **samples**.
- Generate a set of representative parameters drawn in proportion a distributionto the posterior $\theta \sim \mathcal{P}$.
- The magic of marginalisation $\Rightarrow$ perform usual analysis on each sample in turn.
- The golden rule is stay in samples until the last moment before computing summary statistics/triangle plots because

$$\boxed{f(\langle X \rangle) \neq \langle f(X) \rangle}$$

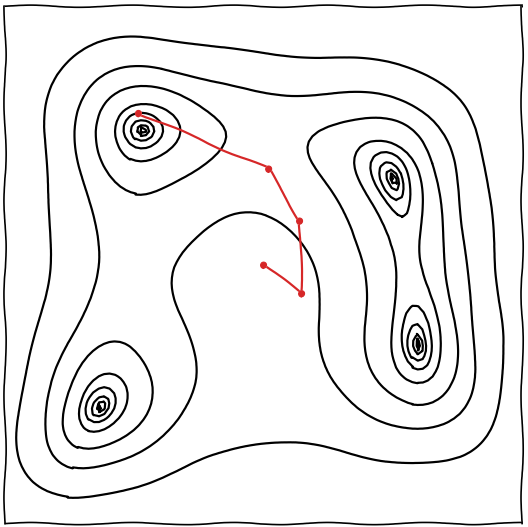
- Generally need $\sim \mathcal{O}(12)$ independent samples to compute a value and error bar.



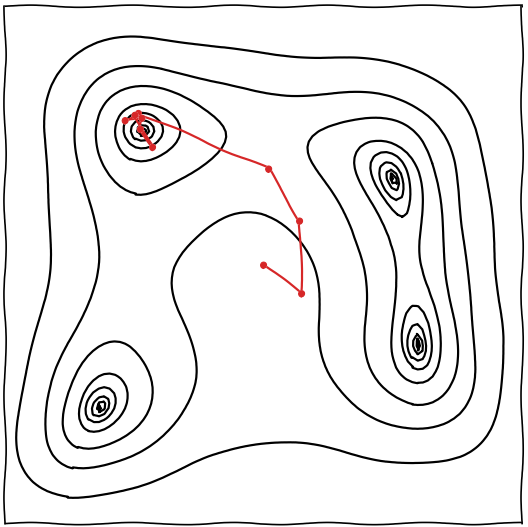
## MCMC



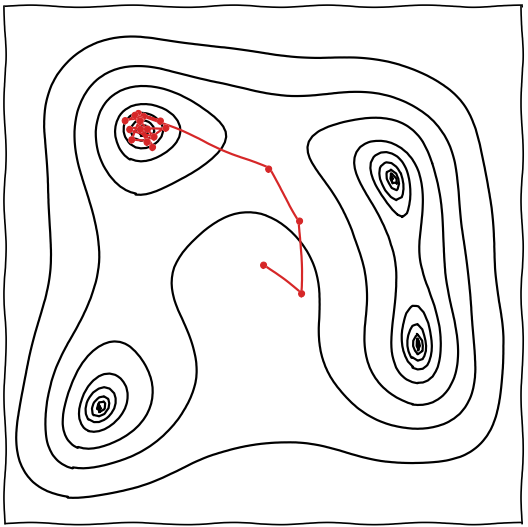
# MCMC



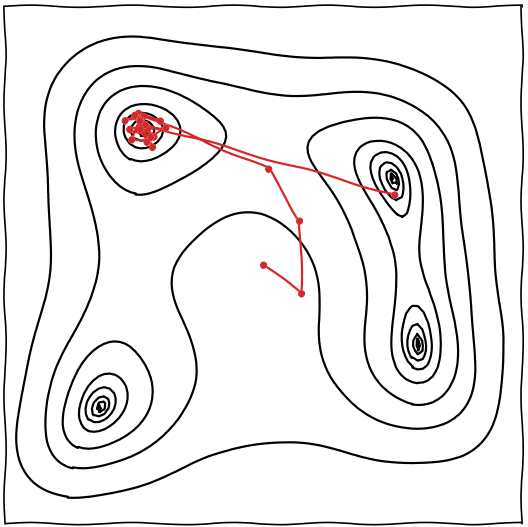
# MCMC



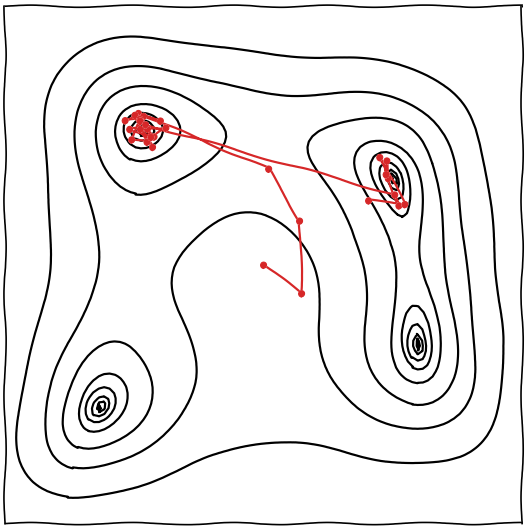
## MCMC



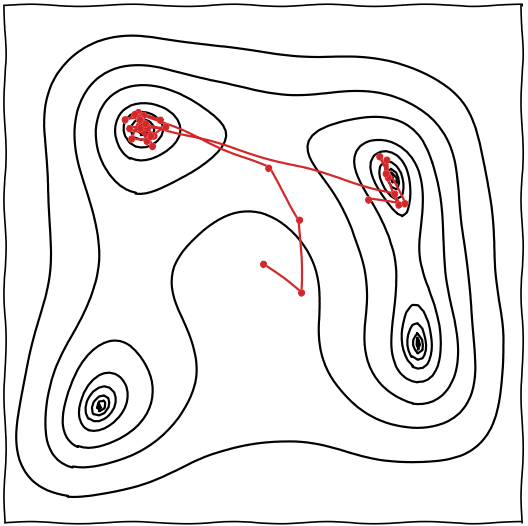
## MCMC



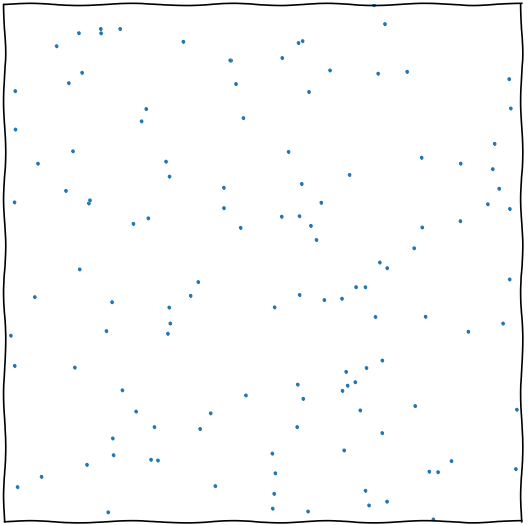
## MCMC



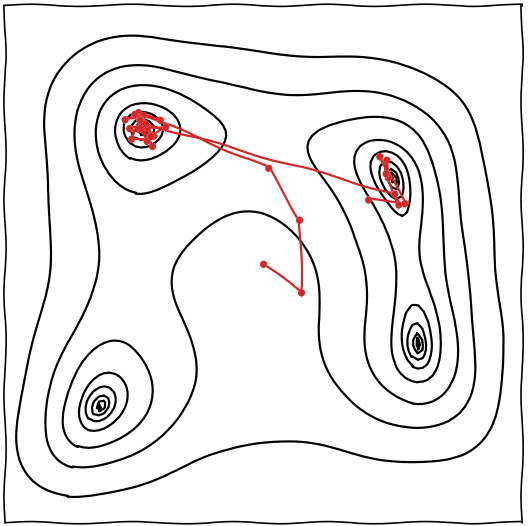
## MCMC



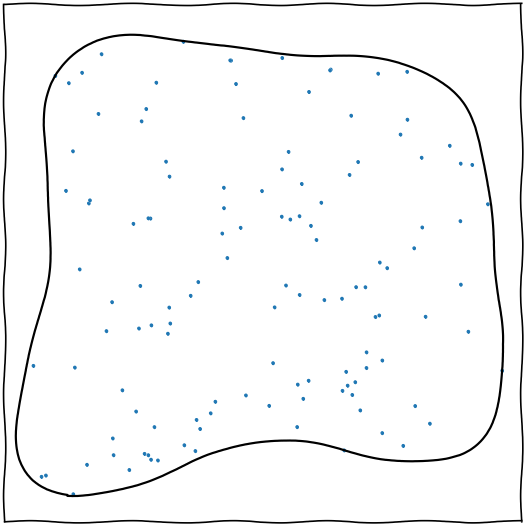
## Nested sampling



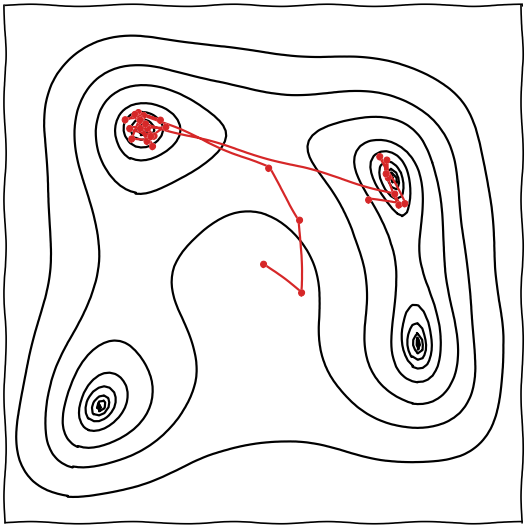
## MCMC



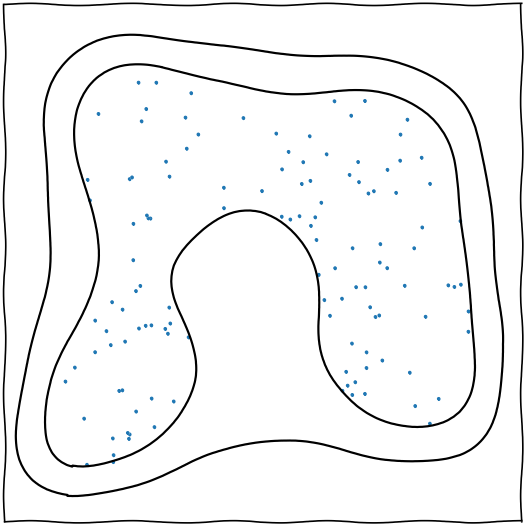
## Nested sampling



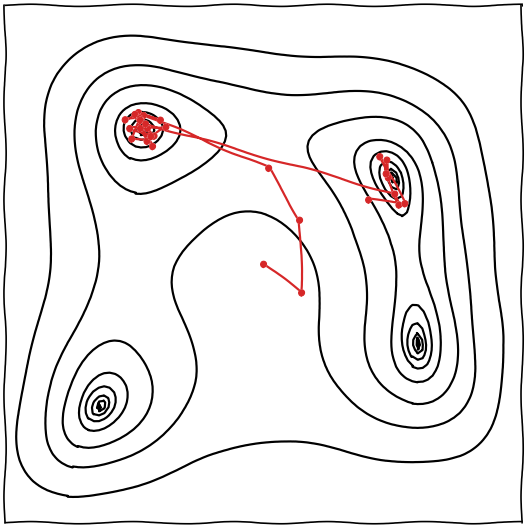
## MCMC



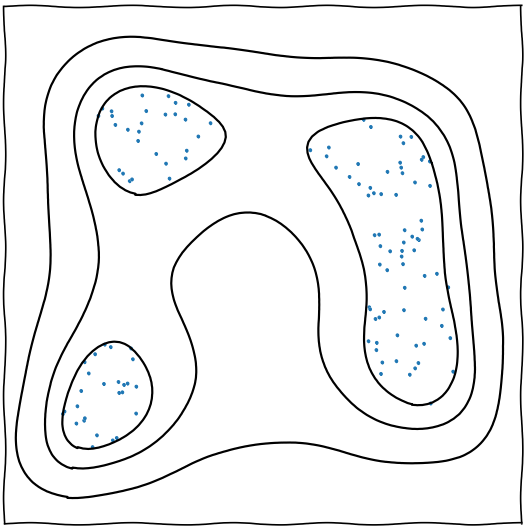
## Nested sampling



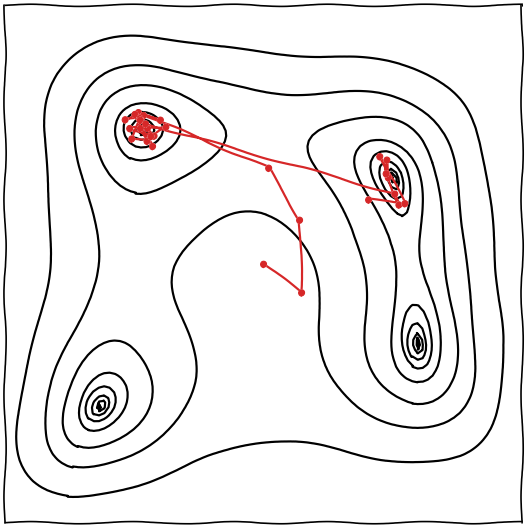
## MCMC



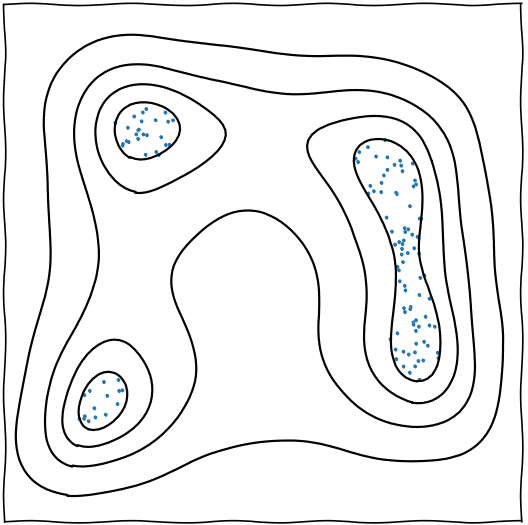
## Nested sampling



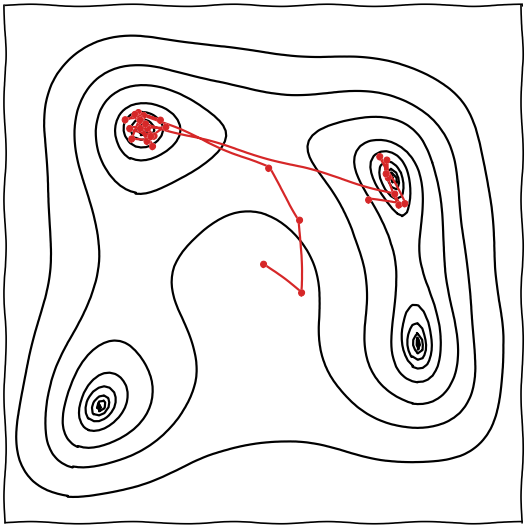
## MCMC



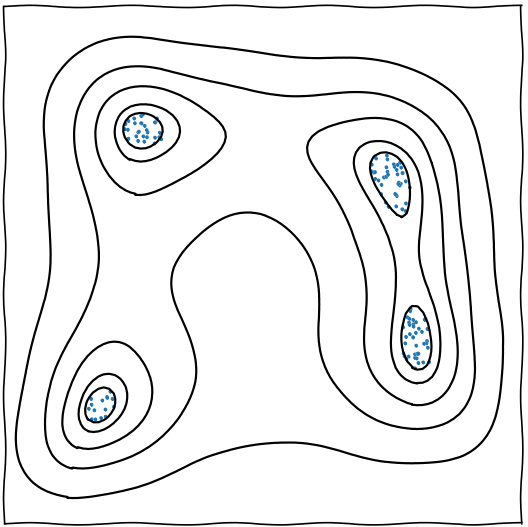
## Nested sampling



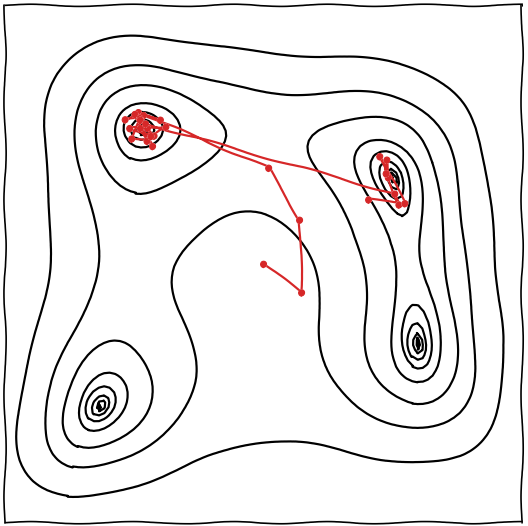
## MCMC



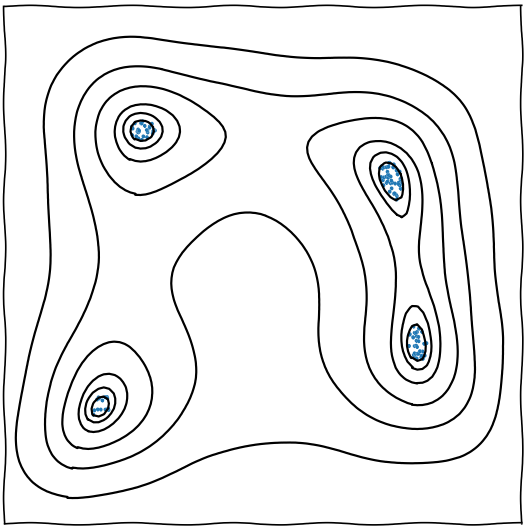
## Nested sampling



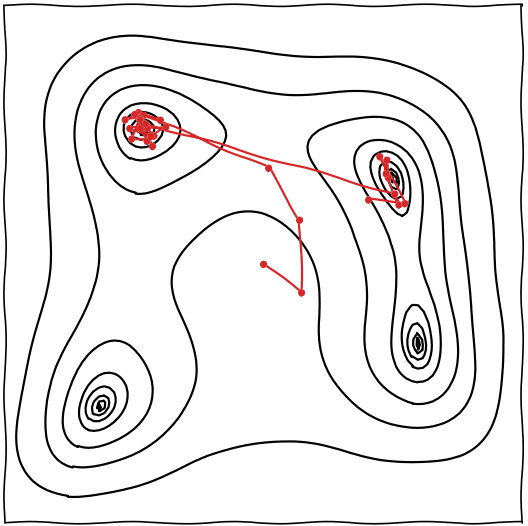
## MCMC



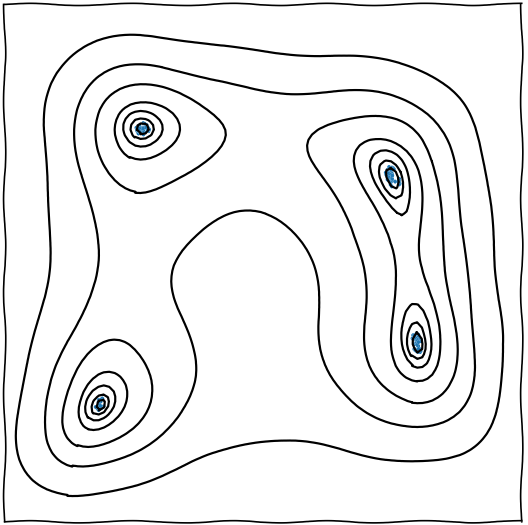
## Nested sampling



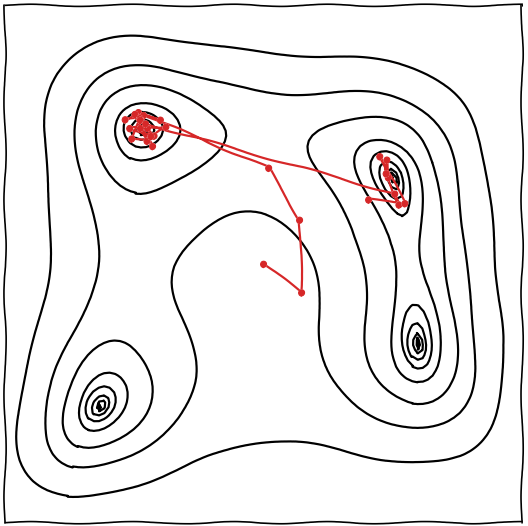
## MCMC



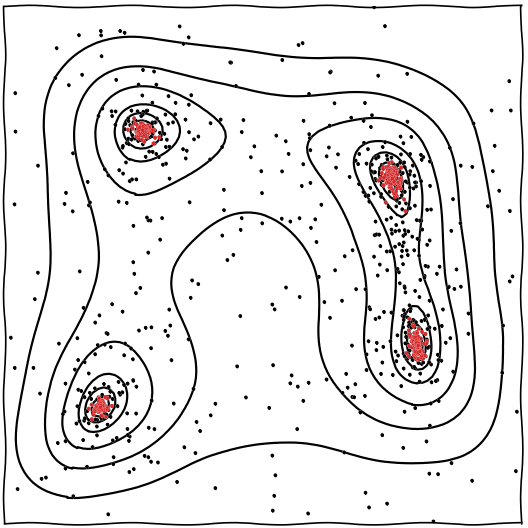
## Nested sampling



## MCMC

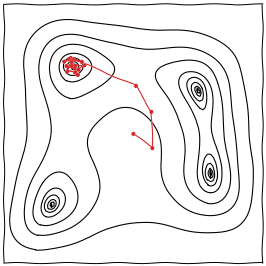


## Nested sampling



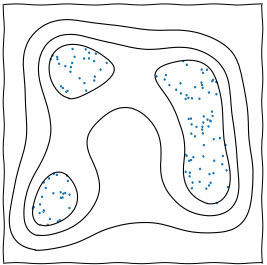
## MCMC

- Single "walker"
- Explores posterior
- Fast, if proposal matrix is tuned
- Parameter estimation, suspiciousness calculation
- Channel capacity optimised for generating posterior samples



## Nested sampling

- Ensemble of "live points"
- Scans from prior to peak of likelihood
- Slower, no tuning required
- Parameter estimation, model comparison, tension quantification
- Channel capacity optimised for computing partition function



# Nested sampling

- Sequentially update a set $S$ of $n$ samples:

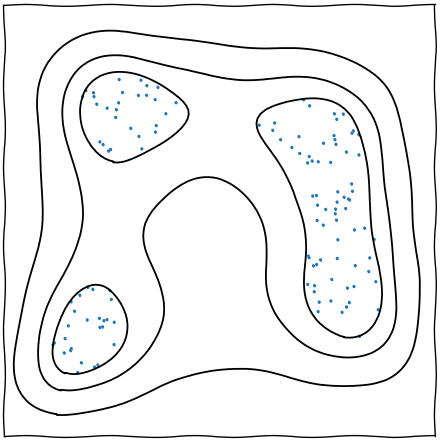
  $S_0$: Generate $n$ samples uniformly over the space (from the prior $\pi$).

  $S_{i+1}$: Delete the lowest likelihood sample in $S_i$, and replace it with a new uniform sample with higher likelihood.

- Requires one to be able to sample uniformly within a region, subject to a *hard likelihood constraint*:

$$\{\theta \sim \pi : \mathcal{L}(\theta) > \mathcal{L}_*.\}$$

- This procedure optimises (multimodally), and can calculate the evidence & posterior weights.
- The evolving ensemble of live points allows algorithms to perform self-tuning and mode clustering.

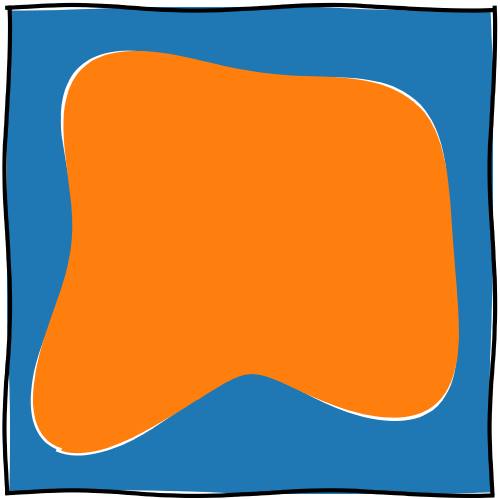


# Probabalistic volume estimation

- Key idea in NS: estimating volumes probabilistically

$$\frac{V_{\text{after}}}{V_{\text{before}}} \approx \frac{n_{\text{in}}}{n_{\text{out}} + n_{\text{in}}}$$

- This is the **only** way to calculate volume in high dimensions $d > 3$.
  - Geometry is exponentially inefficient
- If you want to innovate at the frontier of SBI+NS, this is a USP.

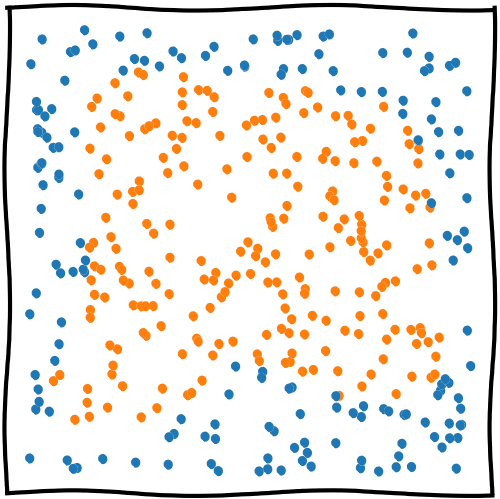


# Probabalistic volume estimation

- Key idea in NS: estimating volumes probabilistically

$$\frac{V_{\text{after}}}{V_{\text{before}}} \approx \frac{n_{\text{in}}}{n_{\text{out}} + n_{\text{in}}}$$

- This is the **only** way to calculate volume in high dimensions $d > 3$.
  - Geometry is exponentially inefficient
- If you want to innovate at the frontier of SBI+NS, this is a USP.

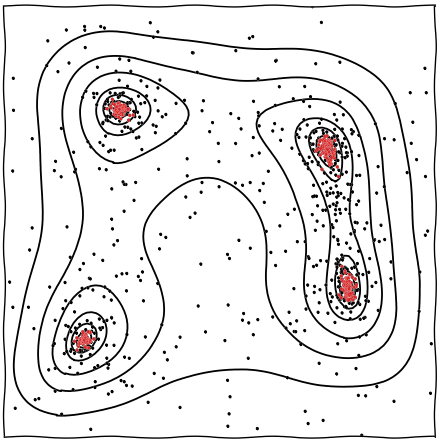


# Dead points: posteriors & evidences

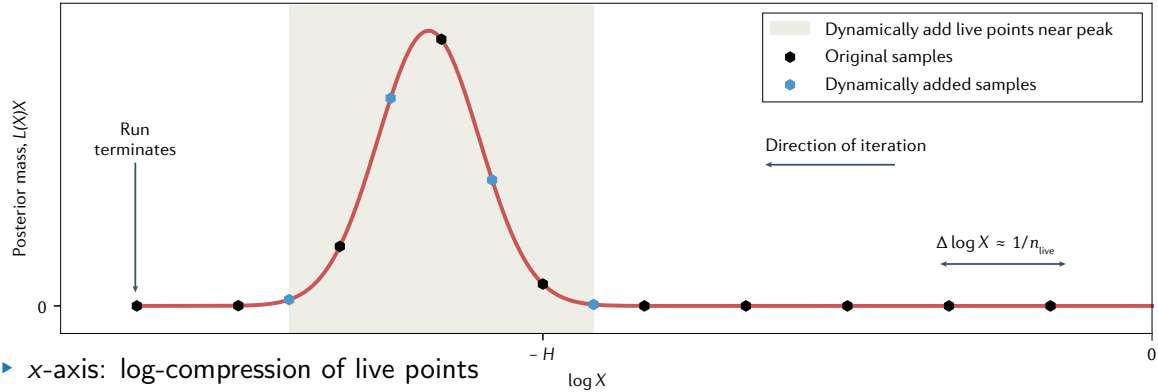
- At the end, one is left with a set of discarded points.
- These may be weighted to form weighted posterior samples using $w_i = \mathcal{L}_i \Delta X_i$.
- They can also be used to calculate the normalisation $\mathcal{Z} = \sum \mathcal{L}_i \Delta X_i$, or more generally $\sum_i f(\mathcal{L}_i) \Delta X_i$.
  - Nested sampling probabilistically estimates the volume of the parameter space

$$X_i \approx \left(\frac{n}{n+1}\right) X_{i-1} \quad \Rightarrow \quad X_i \approx \left(\frac{n}{n+1}\right)^i \approx e^{-i/n},$$

  - Nested sampling thus estimates the density of states,
  - it is therefore a partition function calculator $Z(\beta) = \sum_i \mathcal{L}_i^\beta \Delta X_i$.



# Time complexity of nested sampling




- $x$-axis: log-compression of live points
- Area $\propto$ posterior mass
- Shows Bayesian balance of likelihood vs prior
- Run proceeds right to left
- Run finishes after bump (typical set)

- Time complexity

$$T = n_{\text{live}} \times T_{\mathcal{L}} \times T_{\text{sampler}} \times \mathcal{D}_{\text{KL}}(\mathcal{P}||\pi)$$

- Error complexity $\sigma \propto \sqrt{\mathcal{D}_{\text{KL}}(\mathcal{P}||\pi)/n_{\text{live}}}$

# Sampling from a hard likelihood constraint

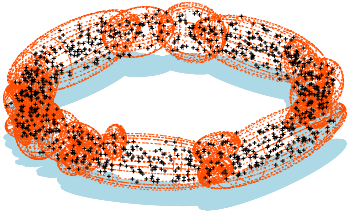
*"It is not the purpose of this introductory paper to develop the technology of navigation within such a volume. We merely note that exploring a hard-edged likelihood-constrained domain should prove to be neither more nor less demanding than exploring a likelihood-weighted space."*

*— John Skilling*

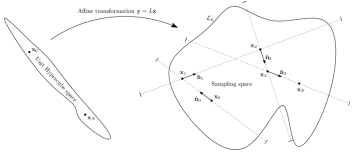
- A large fraction of the work in NS to date has been in attempting to implement a hard-edged sampler in the NS meta-algorithm $\{\theta \sim \pi : \mathcal{L}(\theta) > \mathcal{L}_*\}$.
- `https://projecteuclid.org/euclid.ba/1340370944`.
- There has also been much work beyond this (focus of this talk).

# Implementations of Nested Sampling [2205.15570](NatReview)

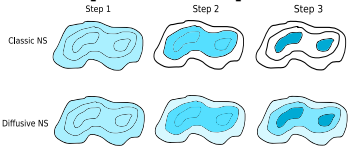
MultiNest [0809.3437]



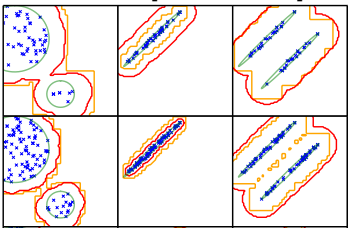
PolyChord [1506.00171]



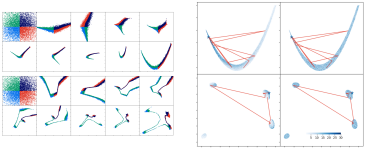

DNest [1606.03757]




UltraNest [2101.09604]

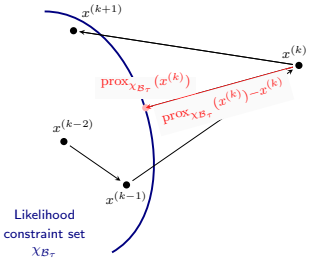


NeuralNest [1903.10860]



dynesty [1904.02180]

ProxNest [2106.03646]



# Types of nested sampler

- Broadly, most nested samplers can be split into how they create new live points.
- i.e. how they sample from the hard likelihood constraint $\{\theta \sim \pi : \mathcal{L}(\theta) > \mathcal{L}_*\}$.

## Rejection samplers

- e.g. `MultiNest`, `UltraNest`.
- Constructs bounding region and draws many invalid points until $\mathcal{L}(\theta) > \mathcal{L}_*$.
- Efficient in low dimensions, exponentially inefficient $\sim \mathcal{O}(e^{d/d_0})$ in high $d > d_0 \sim 10$.

## Chain-based samplers

- e.g. `PolyChord`, `ProxNest`.
- Run Markov chain starting at a live point, generating many valid (correlated) points.
- Linear $\sim \mathcal{O}(d)$ penalty in decorrelating new live point from the original seed point.

- Nested samplers usually come with:
  - *resolution* parameter $n_{\text{live}}$ (which improve results as $\sim \mathcal{O}(n_{\text{live}}^{-1/2})$.
  - set of *reliability* parameters [2101.04525], which don't improve results if set arbitrarily high, but introduce systematic errors if set too low.
  - e.g. `Multinest` efficiency `eff` or `PolyChord` chain length $n_{\text{repeats}}$.

# Nested sampling as an optimiser

- Nested sampling can be used in "pure optimisation mode" for a function $f(\theta)$ by:
- Turning off the stopping criterion (which only makes sense for likelihoods $f = \mathcal{L}$)
  - stopping instead after a fixed number of iterations.
- Pros:
  - The iteration number $i$ is interpretable as "log-volume compressed" $X = e^{-i/n_{\text{live}}}$
  - It is excellent at exploring multimodal functions (local optima)
  - The live points allow an element of multi-objective optimisation, e.g. given a maximum cost, what configurations of nuclear reactor are available?
- Cons:
  - It is not very fast as an optimiser!
  - this can be fixed by running a gradient descent/simplex from the final set of live points to "polish" the solution (PolyChord has this as an option with `settings.optimise=True`)

# How does Nested Sampling compare to other approaches?

- In all cases:
  - \+ NS can handle multimodal functions
  - \+ NS computes evidences, partition functions and integrals
  - \+ NS is self-tuning/black-box

Modern Nested Sampling algorithms can do this in $\sim \mathcal{O}(100s)$ dimensions

## Optimisation

- Gradient descent
  - \+ NS does not require gradients
- Genetic algorithms
  - \+ NS discarded points have statistical meaning

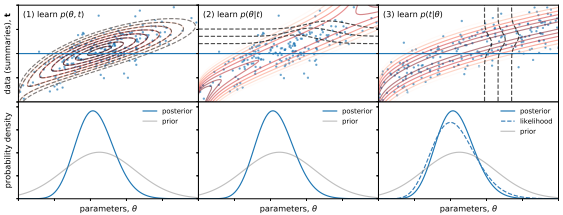
## Sampling

- Metropolis-Hastings?
  - – Very little beats a well-tuned, customised MH
  - \+ NS is self tuning
- Hamiltonian Monte Carlo?
  - – In millions of dimensions, HMC is king
  - \+ NS does not require gradients

## Integration

- Thermodynamic integration
  - \+ protective against phase trasitions
  - \+ No annealing schedule tuning
- Sequential Monte Carlo
  - – Some people (SMC experts) classify NS as a kind of SMC
  - \+ NS is athermal

# Nested Sampling with Likelihood Free Inference




Alsing *et al.* [1903.00007]

- In density estimation likelihood free inference, the output is to learn one/all of:

  Likelihood $P(D|\theta)$
  Posterior $P(\theta|D)$
  Joint $P(D, \theta)$

- In the first instance, nested sampling can be used to scan these learnt functions.
- Data are compressed, so joint space $(D, \theta)$ is navigable by off-the-shelf codes.
  - Sanity checking the solution.
  - Computing evidences/Kullback Liebler divergences from likelihoods.
- Its self-tuning capacity and ability to handle multi-modal distributions can be very useful for diagnosing incompletely learnt functions.
- Emulated likelihoods (e.g. normalising flows) are generally fast, so can deploy more likelihood hungry techniques like NS.

# Nested Sampling for Approximate Bayesian Computation/SBI

- Assume one has a generative model capable of turning parameters into mock data $D(\theta)$
- Given infinite computing power, ABC works by selecting $\{\theta : D(\theta) = D_{\text{observed}}\}$
- These are samples from the posterior, without using a likelihood.
- In practice $D = D_{\text{obs}}$ becomes $D \approx D_{\text{obs}}$
- i.e. $|D - D_{\text{obs}}| < \varepsilon$, or more generally $\boxed{\rho(D, D_{\text{obs}}) < \varepsilon}$, where $\rho$ is some suitably chosen objective function
- Main challenges are
  1. Choice of $\rho$/summary stats
  2. Choice of $\varepsilon$ schedule
  3. Rejection sampling

- Nested sampling fits this well: In principle, can just change the usual hard likelihood constraints $\{\theta \sim \pi : \mathcal{L}(\theta) > \mathcal{L}_*\}$ to

$$\{\theta \sim \pi : \rho(D(\theta), D_{\text{obs}}) < \varepsilon\}$$

(Brewer & Foreman-Mackey [1606.03757])

- Ongoing work with Andrew Fowlie & Sebastian Hoof
  - How to deal with nondeterminism
  - How to interpret $\rho$ as a "likelihood"
  - How to interpret the evidence $\mathcal{Z}$

# Nested Sampling: a user's guide

1. Nested sampling is a likelihood scanner, rather than posterior explorer.
   - This means typically most of its time is spent on burn-in rather than posterior sampling
   - Changing the stopping criterion from $10^{-3}$ to 0.5 does little to speed up the run, but can make results very unreliable
2. The number of live points $n_{\mathrm{live}}$ is a resolution parameter.
   - Run time is linear in $n_{\mathrm{live}}$, posterior and evidence accuracy goes as $\frac{1}{\sqrt{n_{\mathrm{live}}}}$.
   - Set low for exploratory runs $\sim \mathcal{O}(10)$ and increased to $\sim \mathcal{O}(1000)$ for production standard.
   - Extreme MPI parallelisation means walltime can be made constant.
3. Most algorithms come with additional reliability parameter(s).
   - e.g. `MultiNest`: eff, `PolyChord`: $n_{\mathrm{repeats}}$
   - These are parameters which have no gain if set too conservatively, but increase the reliability
   - Check that results do not degrade if you reduce them from defaults, otherwise increase.

# A note on dynesty

- `dynesty` is a pure Python re-implementation of many existing codes
  - (which also now implement dynamic nested sampling)
- This means that it just works$^{TM}$ (particularly on OSX).
- However:

1. Python overheads can be significant (particularly for fast likelihoods)
2. It is not very well parallelised in MPI (one of the key advantages of NS)
3. If you are using the techniques it has reimplemented (`MultiNest`, `PolyChord`, `UltraNest`), you should cite these as well as `dynesty`!

- If you are finding `dynesty` is slow, it may be worth switching!

# Key tools for Nested Sampling

| | |
|---|---|
| `anesthetic` | Nested sampling post processing [1905.04768] |
| `insertion` | cross-checks using order statistics [2006.03371] `github.com/williamjameshandley/anesthetic` |
| `nestcheck` | cross-checks using unthreaded runs [1804.06406] `github.com/ejhigson/nestcheck` |
| `MultiNest` | Ellipsoidal rejection sampling [0809.3437] `github.com/farhanferoz/MultiNest` |
| `PolyChord` | Python/C++/Fortran state of the art [1506.00171] `github.com/PolyChord/PolyChordLite` |
| `dynesty` | Python re-implementation of several codes [1904.02180] `github.com/joshspeagle/dynesty` |
| & `UltraNest` | `github.com/JohannesBuchner/UltraNest` [2101.09604] |
| `SuperNest` | Accelerated nested sampling with prior repartitioning [2212.01760] |

# Occam's Razor [2102.11511]

- Bayesian inference quantifies Occam's Razor:
  - *"Entities are not to be multiplied without necessity"* — William of Occam
  - *"Everything should be kept as simple as possible, but not simpler"* — "Albert Einstein"
- Properties of the evidence: rearrange Bayes' theorem for parameter estimation

$$\mathcal{P}(\theta) = \frac{\mathcal{L}(\theta)\pi(\theta)}{\mathcal{Z}} \qquad \Rightarrow \qquad \log \mathcal{Z} = \log \mathcal{L}(\theta) - \log \frac{\mathcal{P}(\theta)}{\pi(\theta)}$$

- Evidence is composed of a "goodness of fit" term and "Occam Penalty"

- RHS true for all $\theta$. Take max likelihood value $\theta_*$:

$$\log \mathcal{Z} = -\chi^2_{\min} - \text{Mackay penalty}$$

- Be more Bayesian and take posterior average to get the "Occam's razor equation"

$$\boxed{\log \mathcal{Z} = \langle \log \mathcal{L} \rangle_{\mathcal{P}} - \mathcal{D}_{\mathrm{KL}}}$$

- Natural regularisation which penalises models with too many parameters.

# Kullback Liebler divergence

- The KL divergence between prior $\pi$ and posterior $\mathcal{P}$ is is defined as:

$$\mathcal{D}_{\mathrm{KL}} = \left\langle \log \frac{\mathcal{P}}{\pi} \right\rangle_{\mathcal{P}} = \int \mathcal{P}(\theta) \log \frac{\mathcal{P}(\theta)}{\pi(\theta)} d\theta.$$

- Whilst not a distance, $\mathcal{D} = 0$ when $\mathcal{P} = \pi$.
- Occurs in the context of machine learning as an objective function for training functions.
- In Bayesian inference it can be understood as a log-ratio of "volumes":

$$\mathcal{D}_{\mathrm{KL}} \approx \log \frac{V_\pi}{V_\mathrm{P}}.$$

(this is exact for top-hat distributions).