DATA ANALYSIS RECIPES: USING MARKOV CHAIN MONTE CARLO*

David W. Hogg[1, 2, 3, 4] and Daniel Foreman-Mackey[1, 5]

[1] *Center for Computational Astrophysics, Flatiron Institute, 162 Fifth Ave, New York, NY 10010, USA*

[2] *Center for Cosmology and Particle Physics, Department of Physics, New York University, 726 Broadway, New York, NY 10003, USA*

[3] *Center for Data Science, New York University, 60 Fifth Ave, New York, NY 10011, USA*

[4] *Max-Planck-Institut für Astronomie, Königstuhl 17, D-69117 Heidelberg*

[5] *NASA Sagan Fellow; Department of Astronomy, University of Washington, Box 351580, Seattle, WA 98195, USA*

ABSTRACT

Markov Chain Monte Carlo (MCMC) methods for sampling probability density functions (combined with abundant computational resources) have transformed the sciences, especially in performing probabilistic inferences, or fitting models to data. In this primarily pedagogical contribution, we give a brief overview of the most basic MCMC method and some practical advice for the use of MCMC in real inference problems. We give advice on method choice, tuning for performance, methods for initialization, tests of convergence, troubleshooting, and use of the chain output to produce or report parameter estimates with associated uncertainties. We argue that autocorrelation time is the most important test for convergence, as it directly connects to the uncertainty on the sampling estimate of any quantity of interest. We emphasize that sampling is a method for doing integrals; this guides our thinking about how MCMC output is best used.

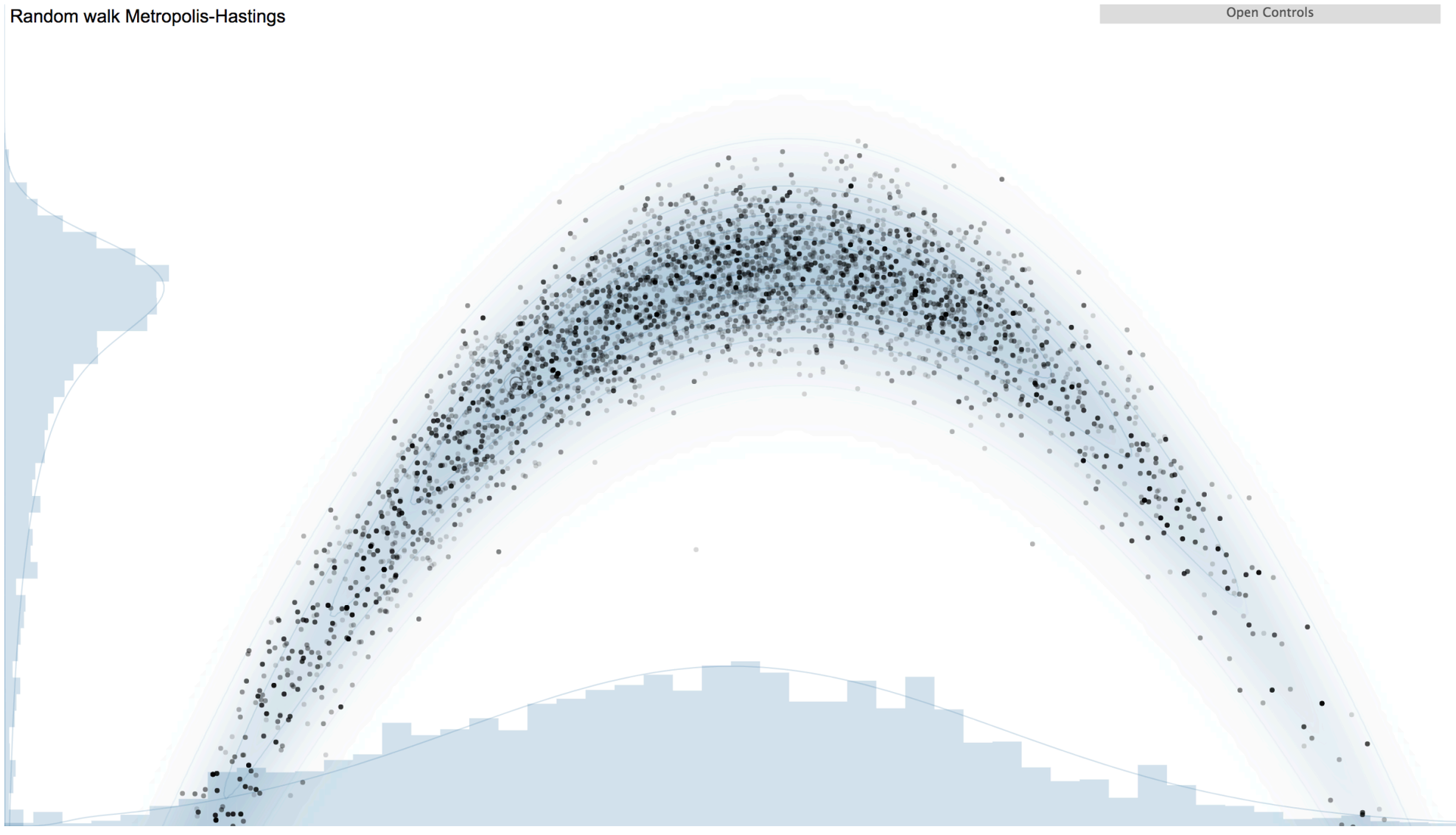
Outline

- MCMC is a sampler!
- Bayesian inference
- How MCMC works
- How to use MCMC
  - Troubleshooting
- Nested Sampling

MCMC is a sampler!

Sampling

sampling is concerned with the selection of a subset of individuals from within a statistical population to estimate characteristics of the whole population.



When do you need MCMC?

- If you are trying to find the optimum of the likelihood or the posterior pdf, you should use a optimizer, not a sampler

- If you want to make sure you search all of the parameter space, you should use a search algorithm, not a sampler

- MCMC is good at one thing, and one thing only: Sampling ill-normalized (or otherwise hard to sample) pdfs.

Bayesian inference

$$\underbrace{p(\mathrm{P}|\mathrm{D},\mathrm{M})}_{\text{posterior}} = \underbrace{p(\mathrm{D}|\mathrm{P},\mathrm{M})}_{\text{likelihood}} \times \underbrace{p(\mathrm{P}|\mathrm{M})}_{\text{prior}} \;/\; \underbrace{p(\mathrm{D}|\mathrm{M})}_{\text{evidence}}$$

P: parameter(s) of interest.

D: data under consideration.

Typically large array(s) of numbers

M: model under consideration.

Can be omitted since conditioned everywhere,
unless multiple models are considered.

Bayesian inference

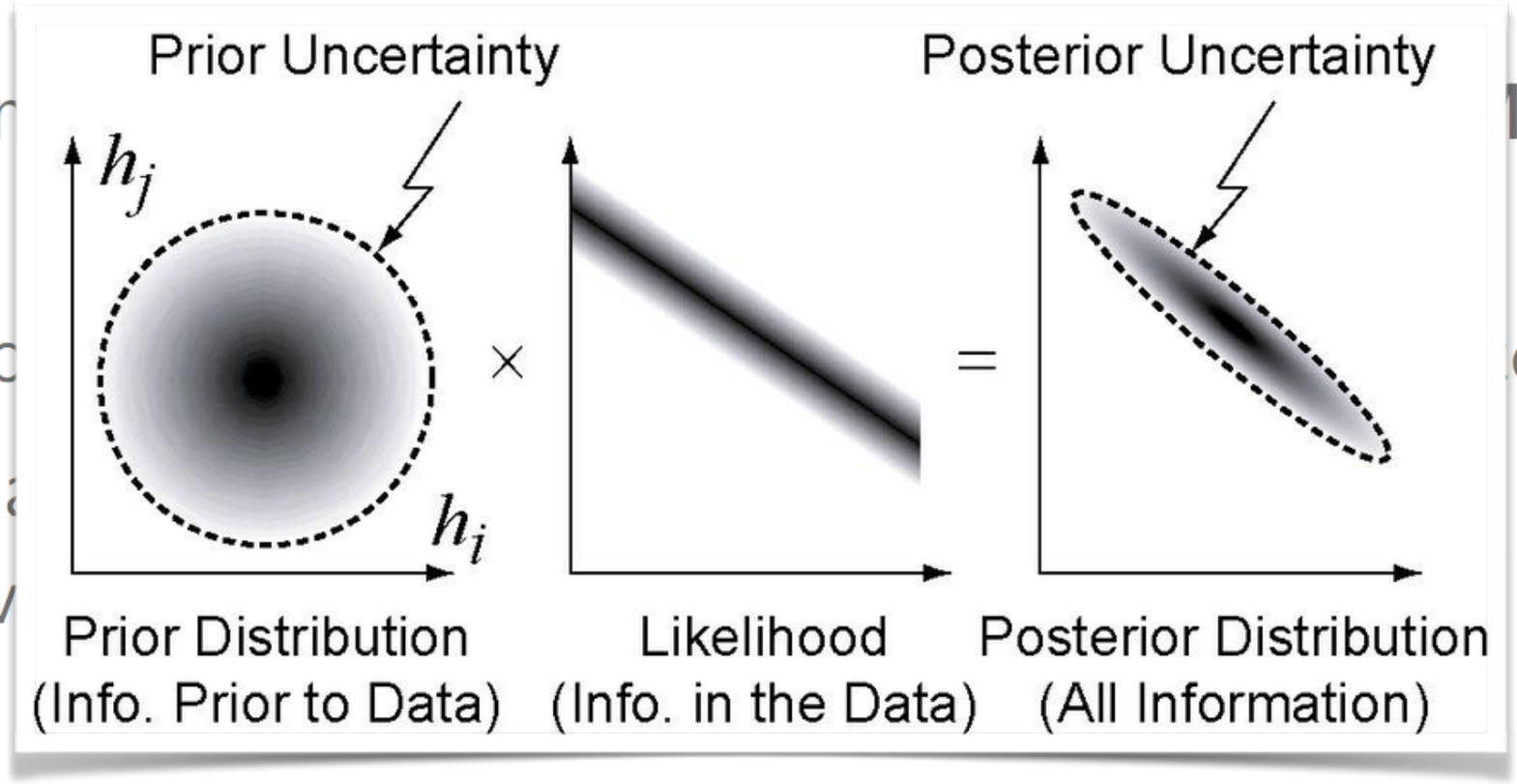
$$\underbrace{p(\mathrm{P}|\mathrm{D},\mathrm{M})}_{\text{posterior}} = \underbrace{p(\mathrm{D}|\mathrm{P},\mathrm{M})}_{\text{likelihood}} \times \underbrace{p(\mathrm{P}|\mathrm{M})}_{\text{prior}} \Big/ \underbrace{p(\mathrm{D}|\mathrm{M})}_{\text{evidence}}$$

- Knowledge about parameters **P** under model **M** before looking at the data **D**.
- Comes from theory, previous data, intuition, etc.
- May impact the results ("prior sensitivity").
  Even "weak" / "uninformative" priors matter!

Bayesian inference

$$\underbrace{p(\mathrm{P}|\mathrm{D},\mathrm{M})}_{\text{posterior}} = \underbrace{p(\mathrm{D}|\mathrm{P},\mathrm{M})}_{\text{likelihood}} \times \underbrace{p(\mathrm{P}|\mathrm{M})}_{\text{prior}} \Big/ \underbrace{p(\mathrm{D}|\mathrm{M})}_{\text{evidence}}$$

- Kr
- Co
- Ma
  Ev



c.

Bayesian inference

$$\underbrace{p(\mathrm{P}|\mathrm{D},\mathrm{M})}_{\text{posterior}} = \underbrace{p(\mathrm{D}|\mathrm{P},\mathrm{M})}_{\text{likelihood}} \times \underbrace{p(\mathrm{P}|\mathrm{M})}_{\text{prior}} \; / \; \underbrace{p(\mathrm{D}|\mathrm{M})}_{\text{evidence}}$$

- The probability of generating the data **D**
  with the parameter **P** under the model **M**

Bayesian inference

$$\underbrace{p(\mathrm{P}|\mathrm{D},\mathrm{M})}_{\text{posterior}} = \underbrace{p(\mathrm{D}|\mathrm{P},\mathrm{M})}_{\text{likelihood}} \times \underbrace{p(\mathrm{P}|\mathrm{M})}_{\text{prior}} \; / \; \underbrace{p(\mathrm{D}|\mathrm{M})}_{\text{evidence}}$$

- Joint PDF on the N parameters of interest given the data **D** and under the model **M**

$$p\big(\Theta_1 = \theta_1, \cdots, \Theta_N = \theta_N \;\big|\; D,\; M\big)$$

Bayesian inference

$$\underbrace{p(\mathrm{P}|\mathrm{D},\mathrm{M})}_{\text{posterior}} = \underbrace{p(\mathrm{D}|\mathrm{P},\mathrm{M})}_{\text{likelihood}} \times \underbrace{p(\mathrm{P}|\mathrm{M})}_{\text{prior}} \Big/ \underbrace{p(\mathrm{D}|\mathrm{M})}_{\text{evidence}}$$

- Integral of the un-normalized posterior PDF:

$$p(D|M) = \int \mathrm{d}P \; p(D,P|M) = \int \mathrm{d}P \; p(D|P,M) \; p(P)$$

- Overall "quality" of the model - only a normalization factor, does not affect the parameters constraints.

- Only important for model comparison

How MCMC works

- **Random Walk** in the parameter space to find maximum likelihood value

Metropolis - Hastings Algorithm

- Draw a proposal θ' from the proposal pdf $q(\theta' \mid \theta_k)$.
- Draw a random number $0 < r < 1$ from the uniform distribution.
- If $f(\theta')/f(\theta_k) > r$ then $\theta_{k+1} \leftarrow \theta'$; otherwise $\theta_{k+1} \leftarrow \theta_k$.

- We repeat this process with a another chain...and many more, until we sampled enough the parameter space

- Let's play! https://chi-feng.github.io/mcmc-demo/app.html

How MCMC works

- **Random Walk** in the parameter space to find maximum likelihood value

- Each chain is characterized by a step size (length of each step) and by the initial point

- Accept new point if likelihood is better than previous one.

- We repeat this process with a another chain...and many more, until we sampled enough the parameter space

- Let’s play! https://chi-feng.github.io/mcmc-demo/app.html

How to use MCMC

- Likelihoods and priors
- Autocorrelation & Convergence
- Tuning
- Initializing and Burn-in
- Results, Error bars, and Figures

My favorite (the simplest to use) package: emcee
http://dfm.io/emcee/current/

Likelihoods and priors

$$\underbrace{p(\mathrm{P}|\mathrm{D},\mathrm{M})}_{\text{posterior}} = \underbrace{p(\mathrm{D}|\mathrm{P},\mathrm{M})}_{\text{likelihood}} \times \underbrace{p(\mathrm{P}|\mathrm{M})}_{\text{prior}} \; / \; \underbrace{p(\mathrm{D}|\mathrm{M})}_{\text{evidence}}$$

MCMC cannot sample a likelihood (which is a probability for the data given parameters) . If you see somebody do so, it is a posterior probability for some implicit (and improper) "flat" priors.

Posterior Pseudo-code :

```
def ln_f(pars, data):
    x = ln_prior(pars)
    if not is_finite(x):
        return -Inf
    return x + ln_likelihood(data, pars)
```

Convergence

- A key question for an MCMC operator is how long to run to be sure of having reliable results.

- There is no simple answer because you can't really ever know that you have sampled the full posterior pdf.

- A simple way: compares the variance (in one parameter, or your most important parameter, or all parameters) within a chain to the variance across chains.

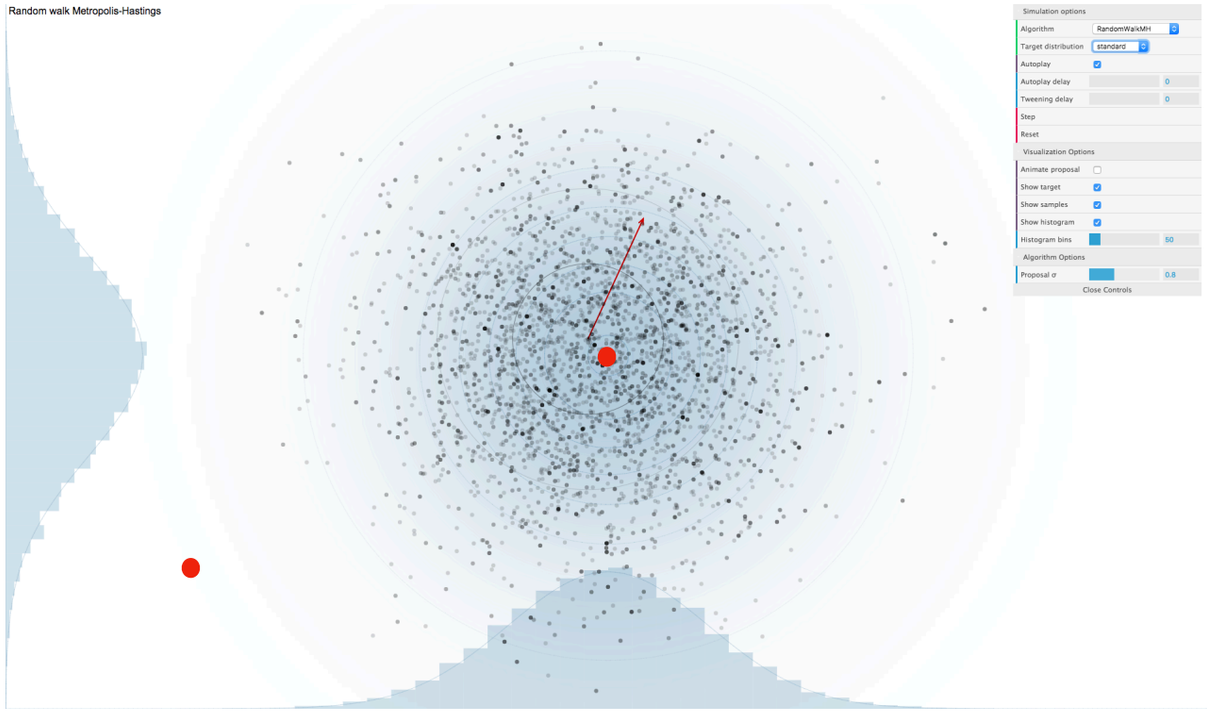
Tuning

Proposal distribution should be “just right”

- If the proposal distribution is too narrow, it proposes steps too small—almost all steps will be accepted but it will take a long time to move anywhere because of timidity.

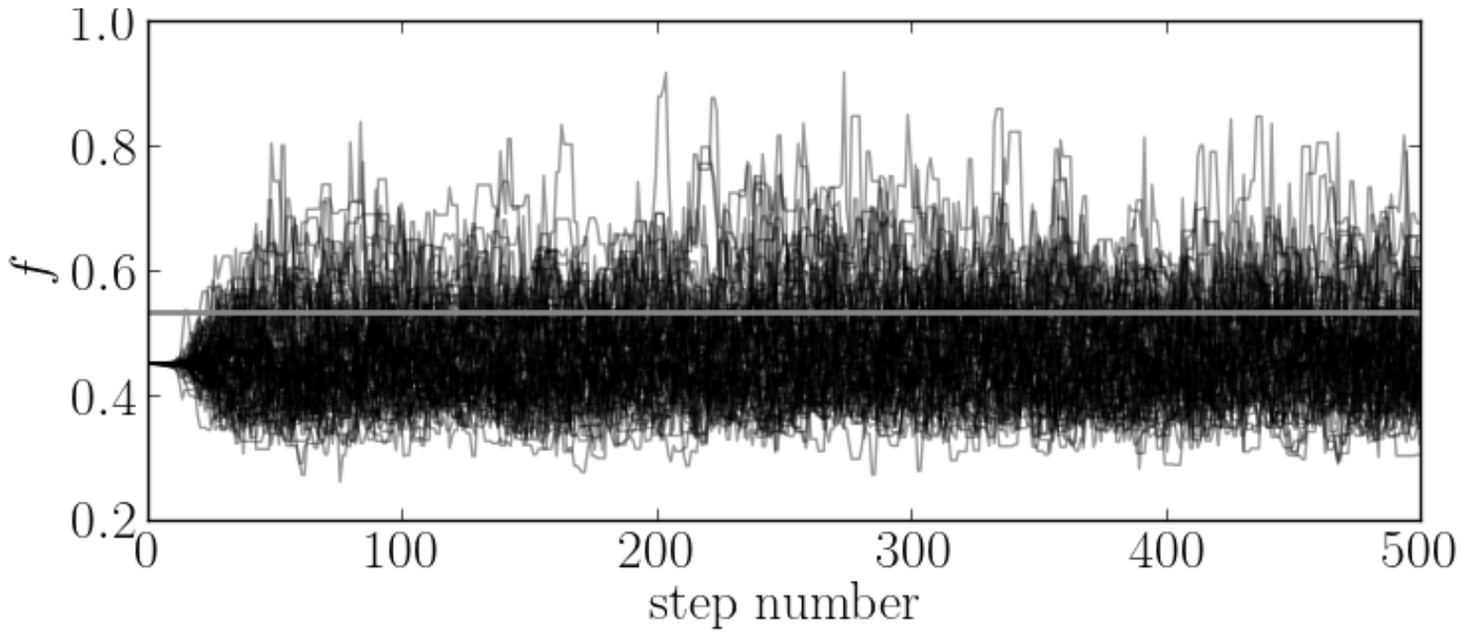
- If the proposal distribution is too wide, it proposes steps too large—the moves will cover parameter space easily, but almost no steps will be accepted; it will tend to jump to much lower probability regions.

Initialization and Burn-in



- **Bad initialization position**

Burn in

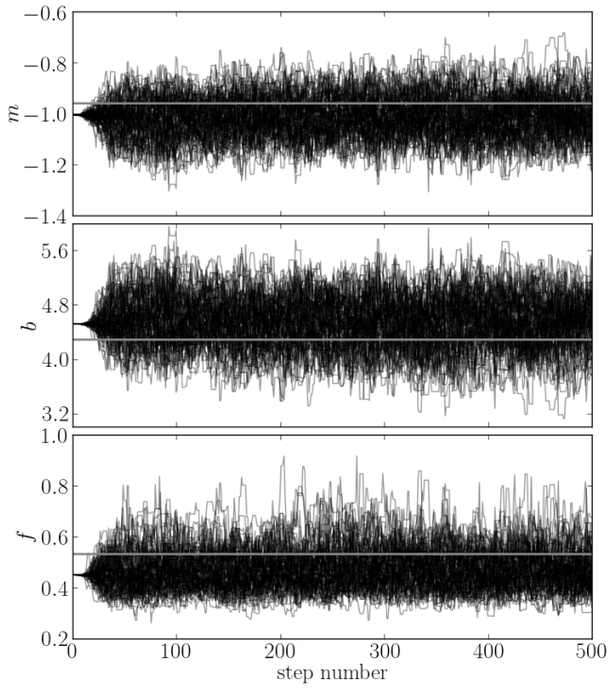


Results

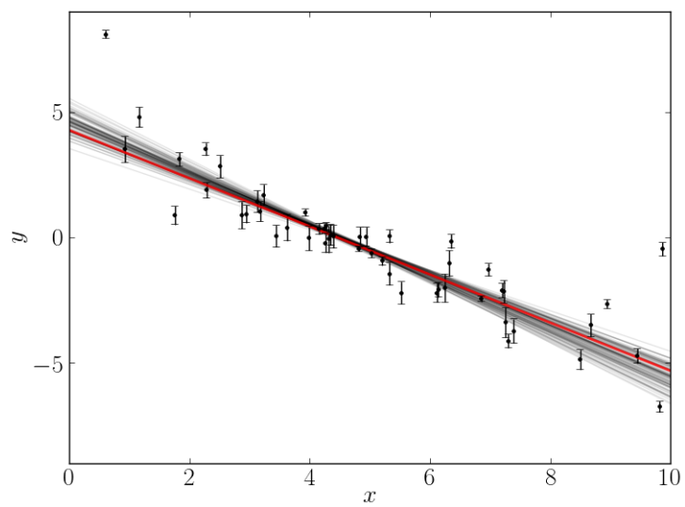
mean, median, quantile (68%, 95%) …

Useful figures should be check every time you run MCMC:

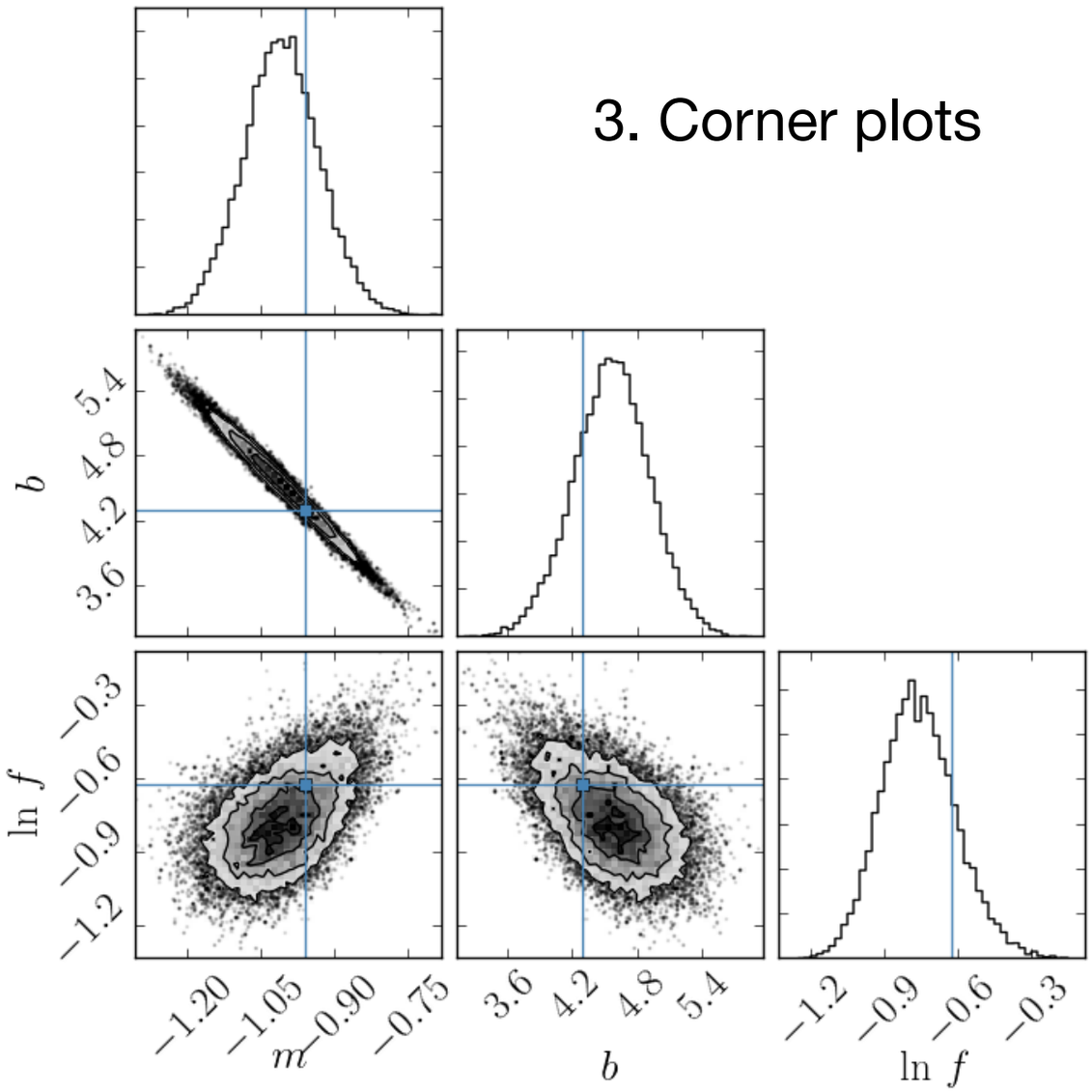
1. Trace plots



2. Posterior predictive plots



3. Corner plots



An example: http://dfm.io/emcee/current/user/line/

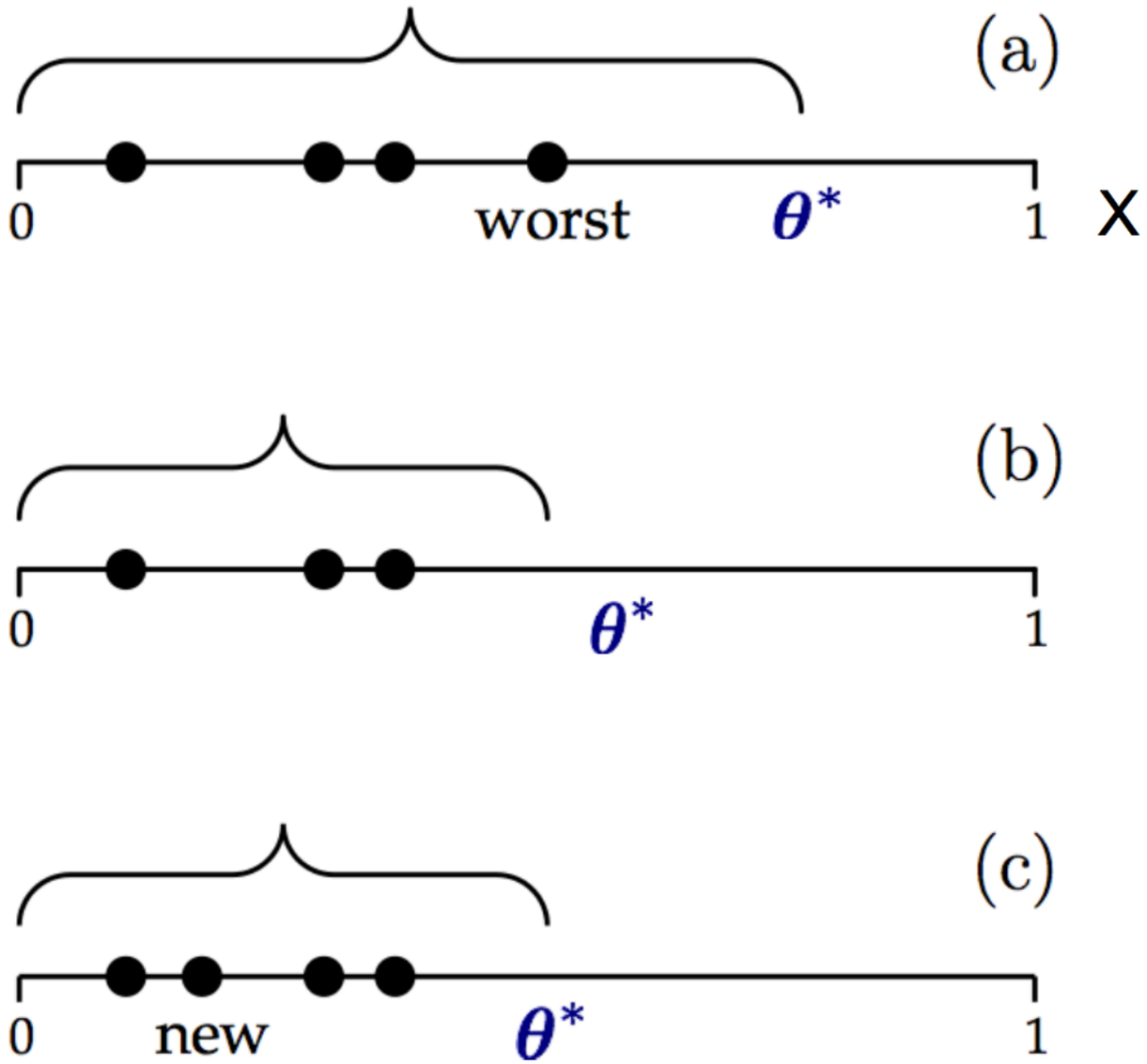
Troubleshooting

- *Functional testing* —In addition to a full set of unit tests for every part of your code.

- *Likelihood issues* — Error: Nan or infinite.
    - Check your prior, proper range of your parameters;
    - Check the likelihood value, it should be positive.
- *Bad initialization*

Nested Sampling

- For each live point compute the likelihood value (straightforward), thus obtaining **J** likelihood values
- Take the worst likelihood point L_{worst}, in this sample and remove it from the sample
- Store the likelihood value and the live point removed
- Draw a new live point that satisfies the new constrain
  $L > L_{worst}$



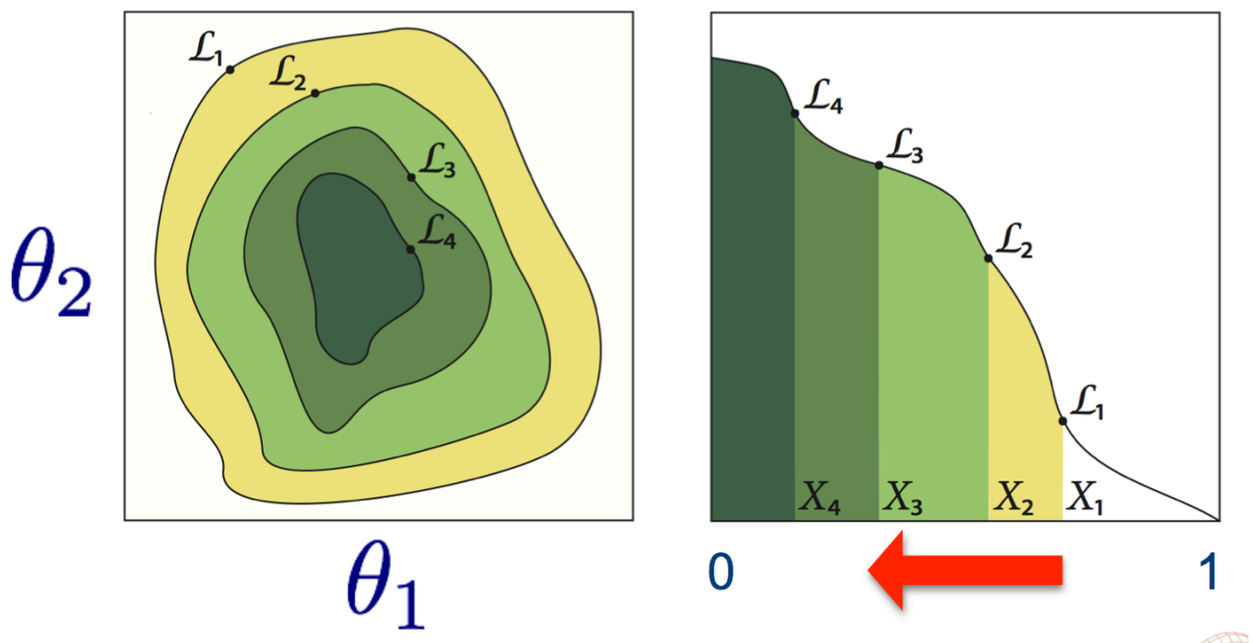
Nested Sampling

Prior $\pi(\theta)$

Evidence $\epsilon = \int L(\theta)\pi(\theta)d\theta$

$dX = \pi(\theta)d\theta$ ⟵ $\int_{-\infty}^{\infty} \pi(\theta)d(\theta) = 1$

$$\epsilon = \int_0^1 L(X)dX$$

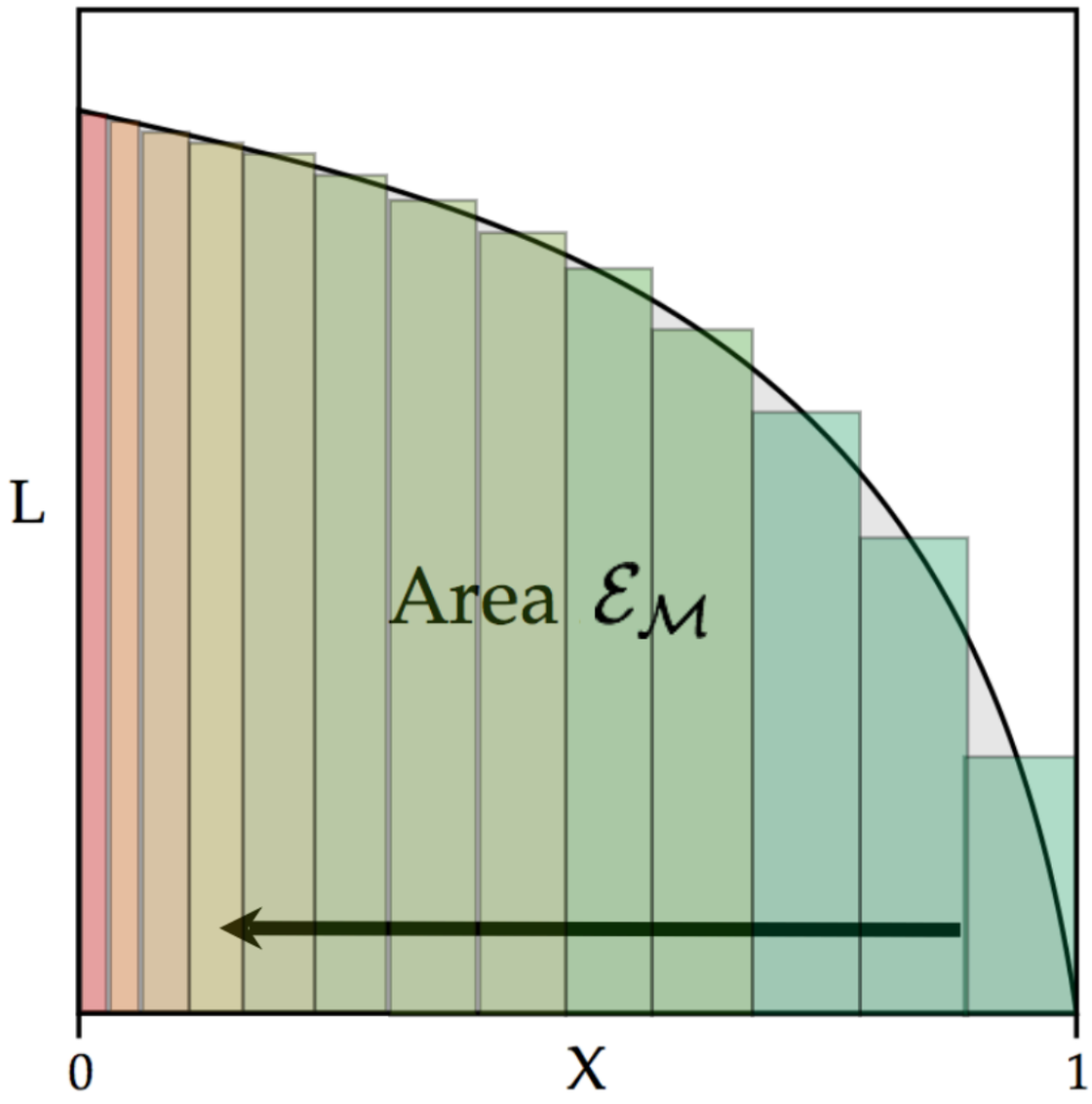


Nested Sampling

$$\mathcal{E} = \int_0^1 \mathcal{L}(X)\, dX$$

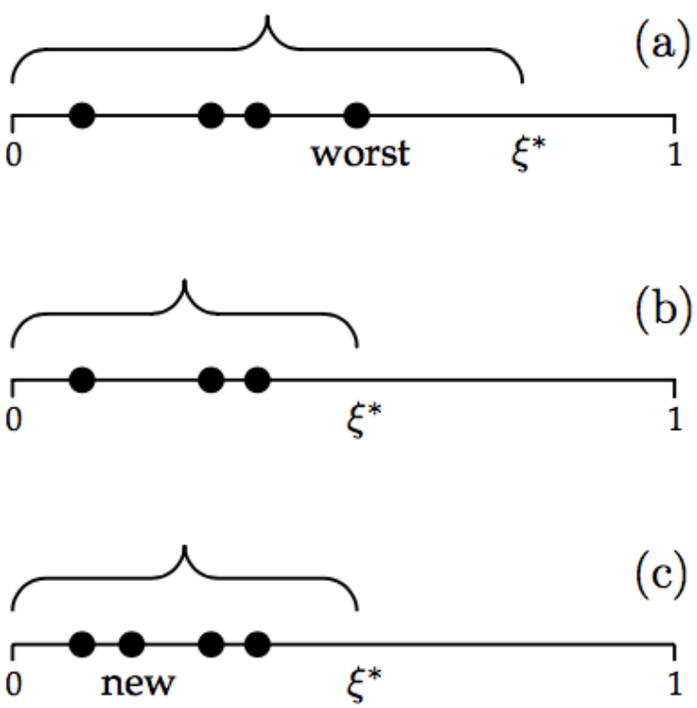
$$\mathcal{E} = \sum_{i=0}^{M} \mathcal{L}_i \Delta X_i$$

$0 < X_M < \ldots < X_2 < X_1 < 1$



Nested Sampling

Nested sampling: the basic idea



9.2.1 Iterating a sequence of objects

On entry, an iteration holds n objects restricted to $\xi < \xi^*$, as shown in Fig. 9.4(a). The worst of these, being the one with smallest likelihood and hence largest ξ, is selected. Located at the largest of n numbers uniformly distributed in $(0, \xi^*)$, it will lie about one part in n less than ξ^*. More technically, the shrinkage ratio $t = \xi/\xi^*$ is distributed as

$$\text{prob}(t) \;=\; n\, t^{n-1}\,, \tag{9.9}$$

with mean and standard deviation

$$\log t \;=\; (-1 \pm 1)/n\,. \tag{9.10}$$

Iteration proceeds by using the worst object's $(\xi, \mathcal{L})$ as the new $(\xi^*, \mathcal{L}^*)$. Meanwhile, the worst object, no longer obeying the constraint, is discarded. There are now $n-1$ surviving objects, still distributed uniformly over ξ but confined to a shrunken domain bounded by the new constraint ξ^*; this is illustrated in Fig. 9.4(b). The new domain is nested within the old, hence the name 'nested sampling'. The next step is to generate a replacement object, sampled uniformly over the prior but constrained within this reduced domain. For now, we assume that we are able to do this. Having done it, the iteration again holds n objects restricted to $\xi < \xi^*$, as in Fig. 9.4(c), just like on entry except for the 1-part-in-n shrinkage. The loop is complete, and the next iteration can be started.

Successive iterations generate a sequence of discarded objects on the edges of progressively smaller nested domains. At iterate k,

$$\mathcal{L}_k = \mathcal{L}^* \quad \text{and} \quad \xi_k = \xi^* = \prod_{j=1}^{k} t_j\,, \tag{9.11}$$

in which each shrinkage ratio t_j is independently distributed with the pdf of eqn (9.9) with the statistics of eqn (9.10). It follows that

$$\log \xi_k \;=\; (-k \pm \sqrt{k}\,)/n\,. \tag{9.12}$$

Nested Sampling

- Distributions sampled efficiently and Bayesian evidence computation very accurate (typically requires **100** times less samples than thermodynamic integration to reach same accuracy, + error bar)
- Sampling method insensitive to any phase change in likelihood i.e. can sample very well multi-modal distributions
- Posterior probability values (required for parameter estimation) are a simple by product!

$$p_i = \frac{\mathcal{L}_i \Delta X_i}{\mathcal{Z}}$$