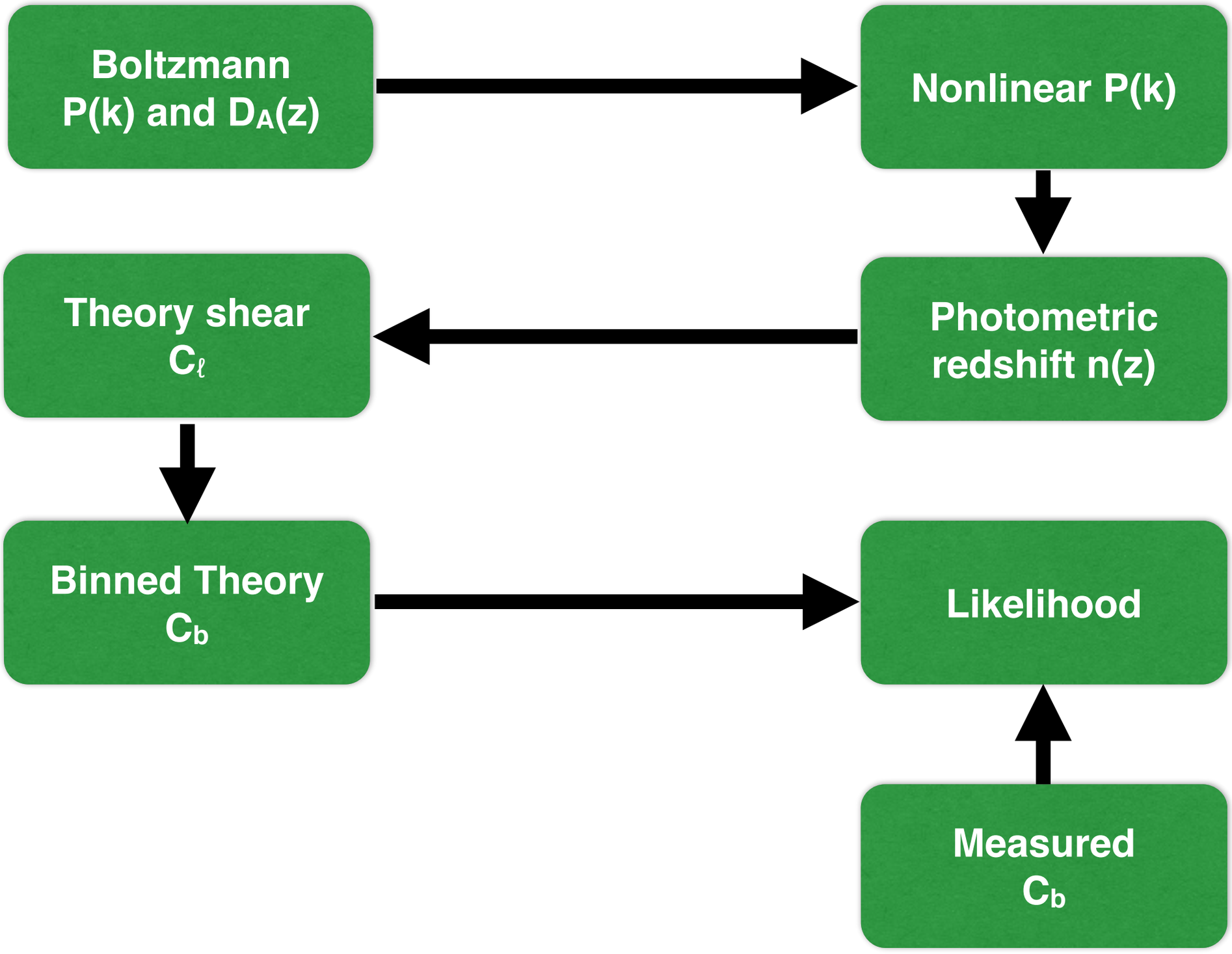
Observations to Models
Lecture 3

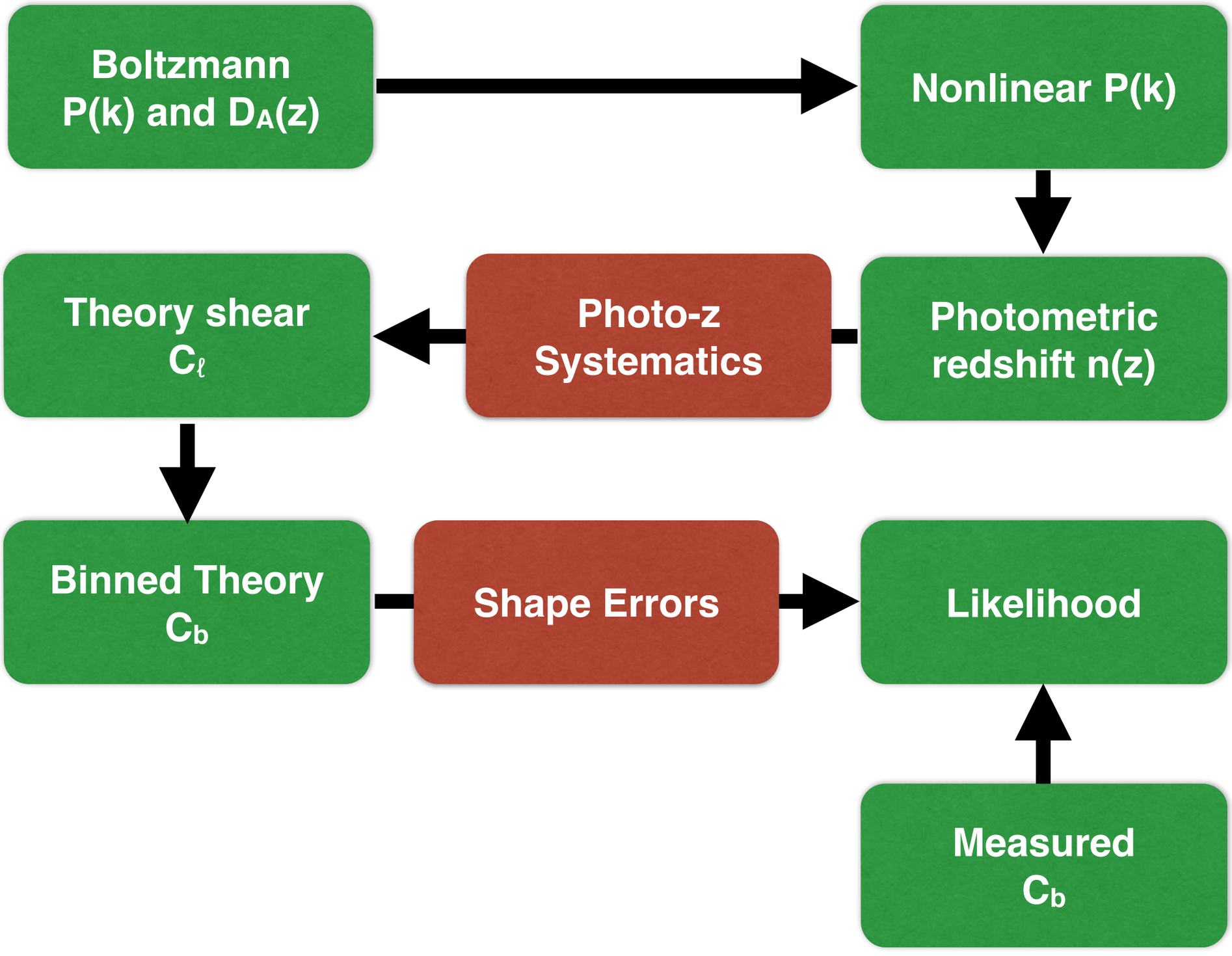
Exploring likelihood spaces with CosmoSIS

Joe Zuntz
University of Manchester

Defining Likelihood Pipelines



Defining Likelihood Pipelines



Modular Likelihood Pipelines

- Each box is an independent piece of code (*module*)
- With well-defined inputs and outputs
    - This lets you replace, compare, insert, and combine code more easily

CosmoSIS

- CosmoSIS is a parameter estimation framework
- Connect inference methods to cosmology likelihoods

https://bitbucket.org/joezuntz/cosmosis

Setting up CosmoSIS on teaching machines

- Open VirtualBox

- Click “start”

- Password cosmosis

- Click Terminal

> su

- use password **cosmosis**

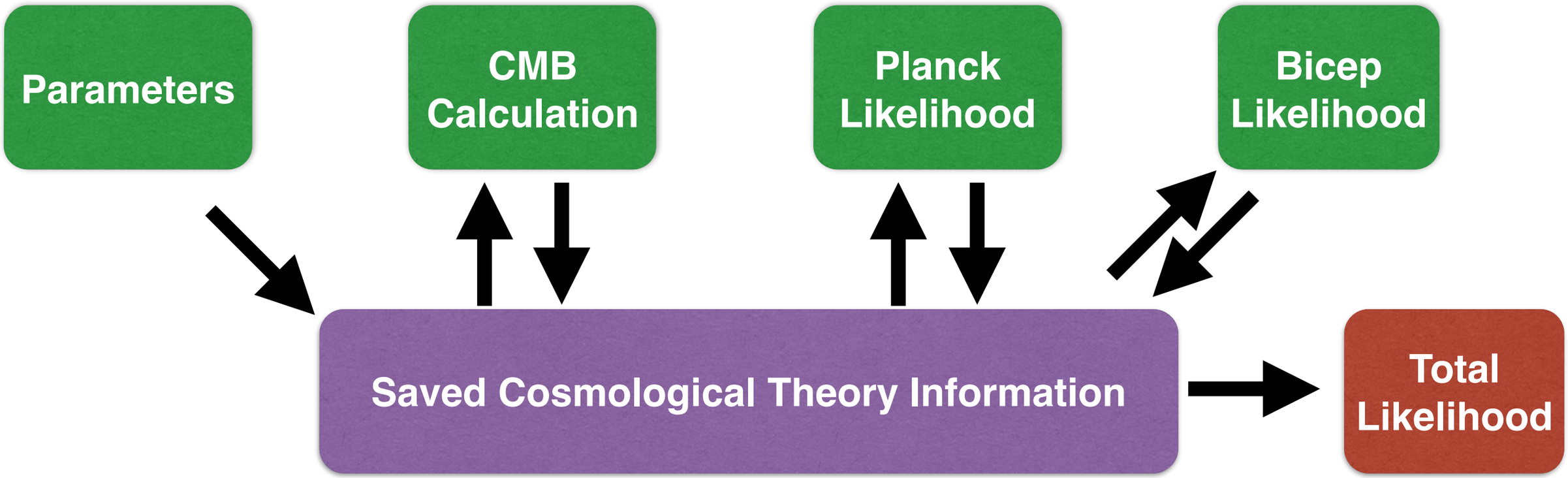
> cd /opt/cosmosis

> source config/setup-cosmosis

Pipeline Example

```
> cosmosis demos/demo2.ini

> postprocess -o plots -p demo2 demos/demo2.ini
```



https://bitbucket.org/joezuntz/cosmosis/wiki/Demo2

Basic Inference Methods

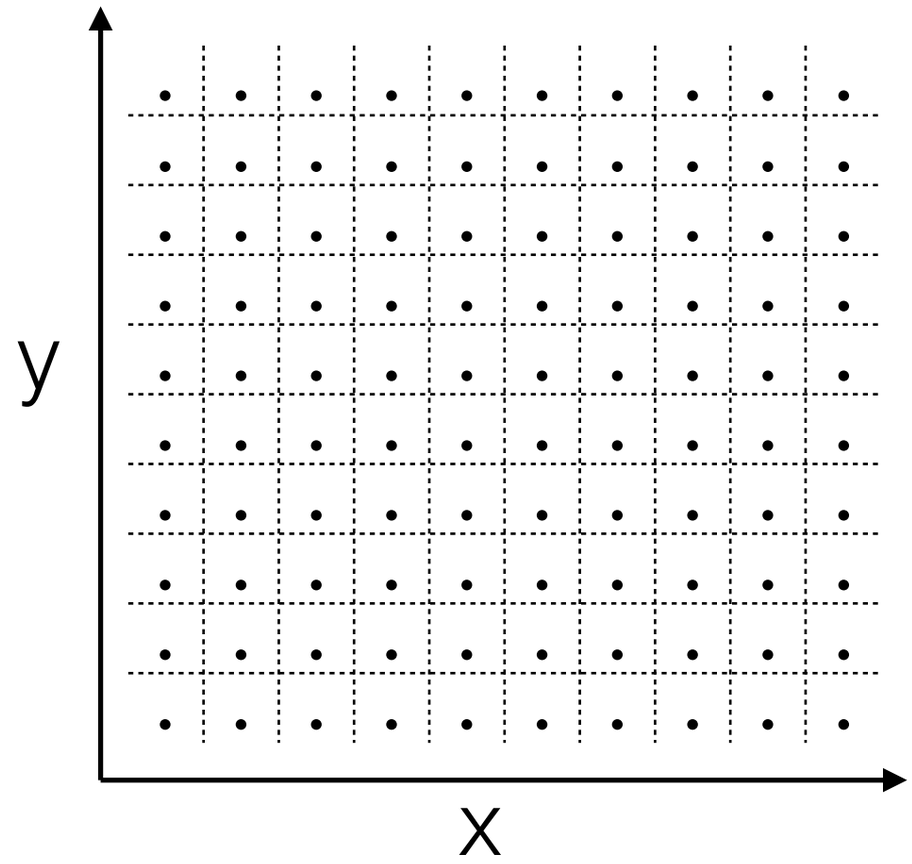
- You've made a posterior function from a likelihood function and a prior
    - Now what?
- Explore the parameter space and build constraints on parameters
    - Remember: "the answer" is a probability distribution

Basic Inference: Grids

- Low number of parameters
  ⇒can try grid of all possible parameter combinations

- Number of posterior evaluations
  = (grid points per param)$^{(\text{number params})}$

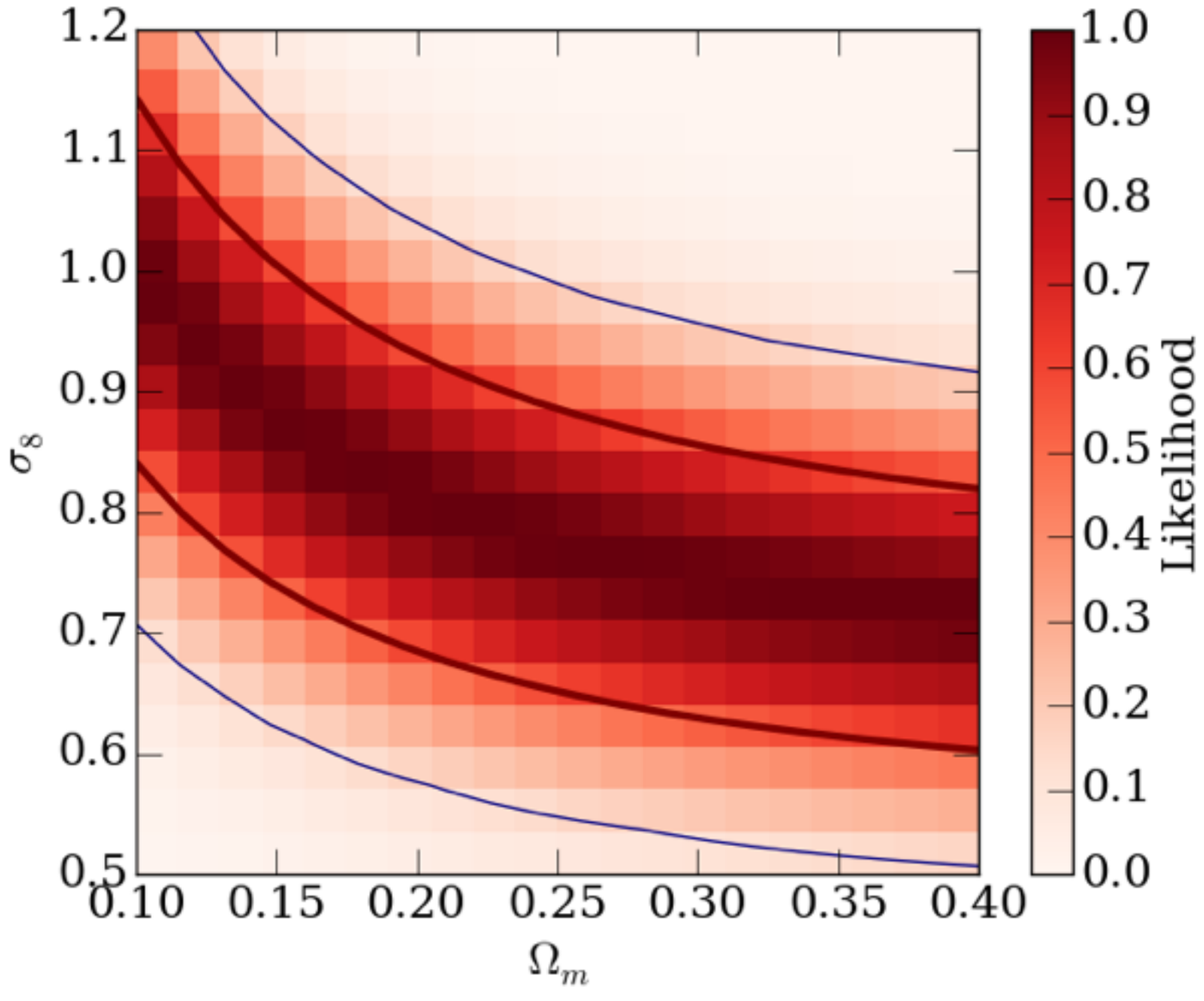
- Approximate PDF integrals as sums



Basic Inference: Grids

- Get constraints on lower dimensions by summing along axes

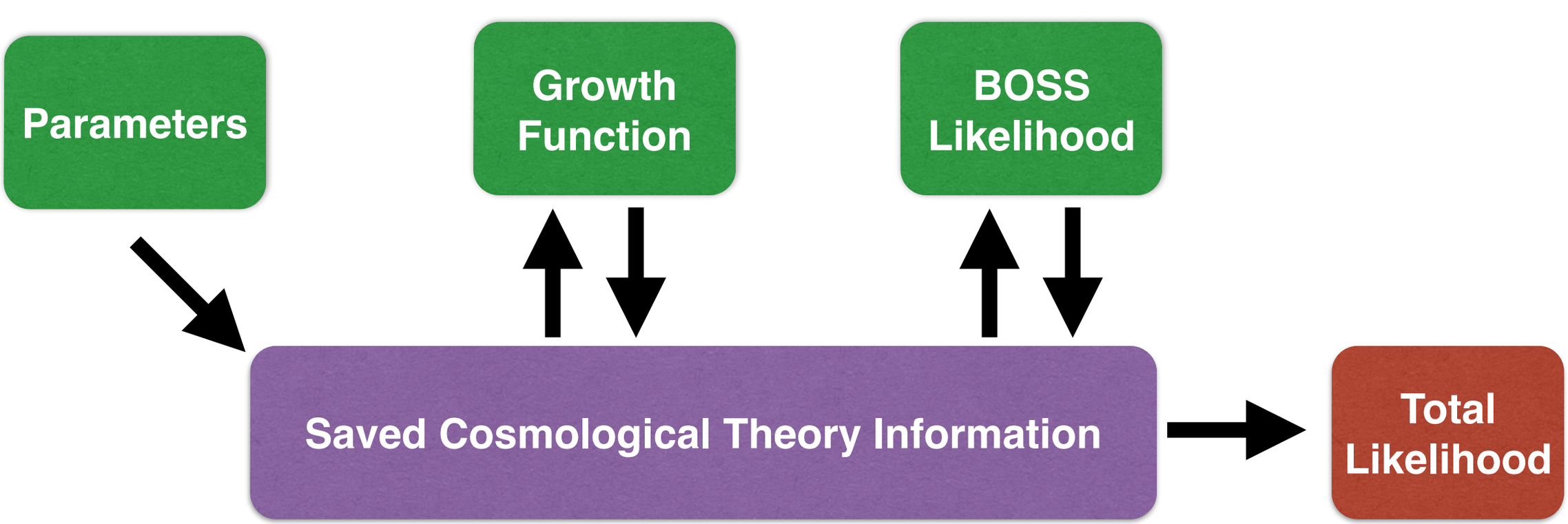
$$P(X) = \int P(XY)\mathrm{d}y$$
$$\approx \sum_i \delta Y_i P(X, Y_i)$$
$$\propto \sum_i \delta P(X, Y_i)$$



Grid Example

```
> cosmosis demos/demo7.ini

> postprocess -o plots -p demo7 demos/demo7.ini
```



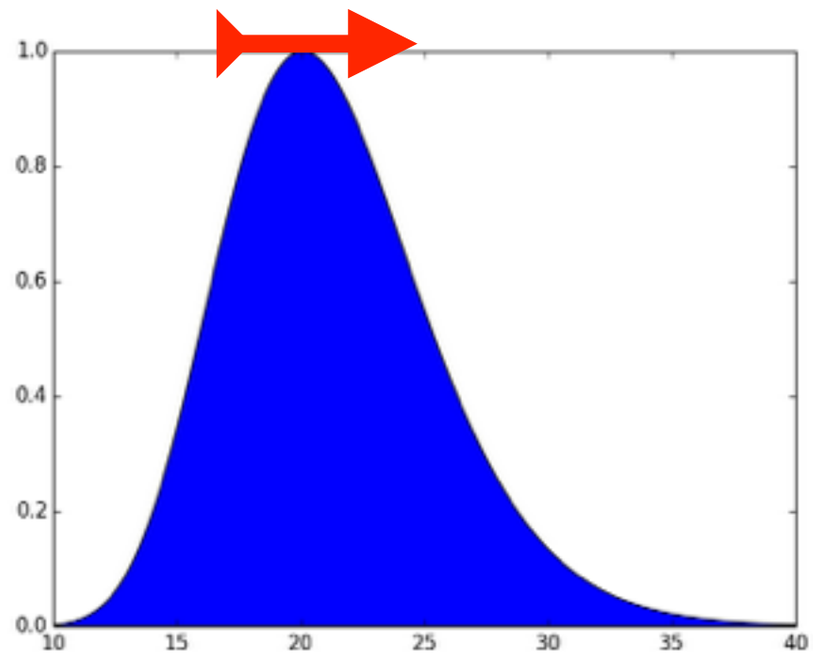
https://bitbucket.org/joezuntz/cosmosis/wiki/Demo7

Basic Inference: Maximum Likelihood/Posterior

- The modal value of a distribution can be of interest, especially for near-symmetric distributions

- Find mode by solving gradient=0

- Maximum-likelihood (ML)
  Maximum a Posteriori (MAP)



Basic Inference: Maximum Likelihood/Posterior

- In 1D cases solve:

$$\frac{\mathrm{d}P}{\mathrm{d}x} = 0 \quad \text{or} \quad \frac{\mathrm{d}\log P}{\mathrm{d}x} = 0$$

- Multi-dimensional case:

$$\underline{\nabla} P = 0 \quad \text{or} \quad \underline{\nabla} \log P = 0$$

Basic Inference: Maximum Likelihood/Posterior

- Around the peak of a likelihood the curvature is related to distribution width

- Usually easiest to do with logs, for example for a Gaussian:

$$\left.\frac{\mathrm{d}^2 \log P}{\mathrm{d}x^2}\right|_{x=\mu} = -\frac{1}{\sigma^2}$$

ML Exercise

- Find the maximum likelihood of our LMC cepheid likelihood, assuming fixed σ_i:

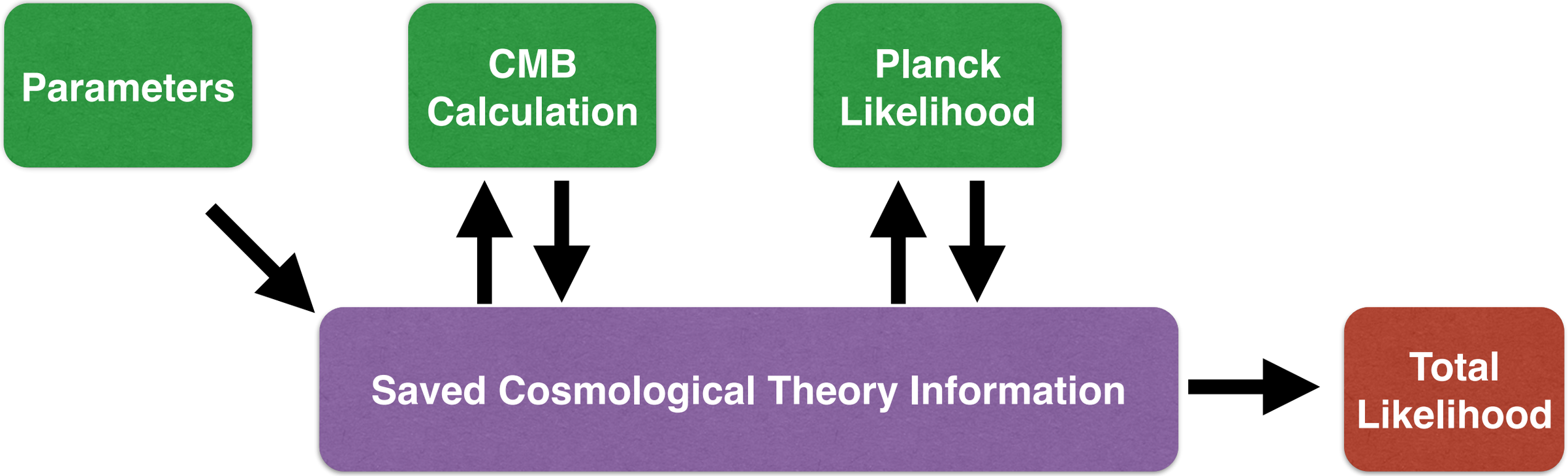
$$P(V_i^{\rm obs}|\mathbf{p}) \propto \frac{1}{\sigma_{\rm int}^2 + \sigma_i^2} \exp -0.5 \left(\frac{(V_i^{\rm obs} - (\alpha + \beta \log_{10} P_i))^2}{\sigma_{\rm int}^2 + \sigma_i^2} \right)$$

- Much easier to use log(P)

ML Example

```
> cosmosis demos/demo4.ini

> postprocess -o plots -p demo4 demos/demo4.ini
```



https://bitbucket.org/joezuntz/cosmosis/wiki/Demo4

Basic Inference: Maximum Likelihood/Posterior

- In simple cases you can solve these analytically. Otherwise need some kind of optimization

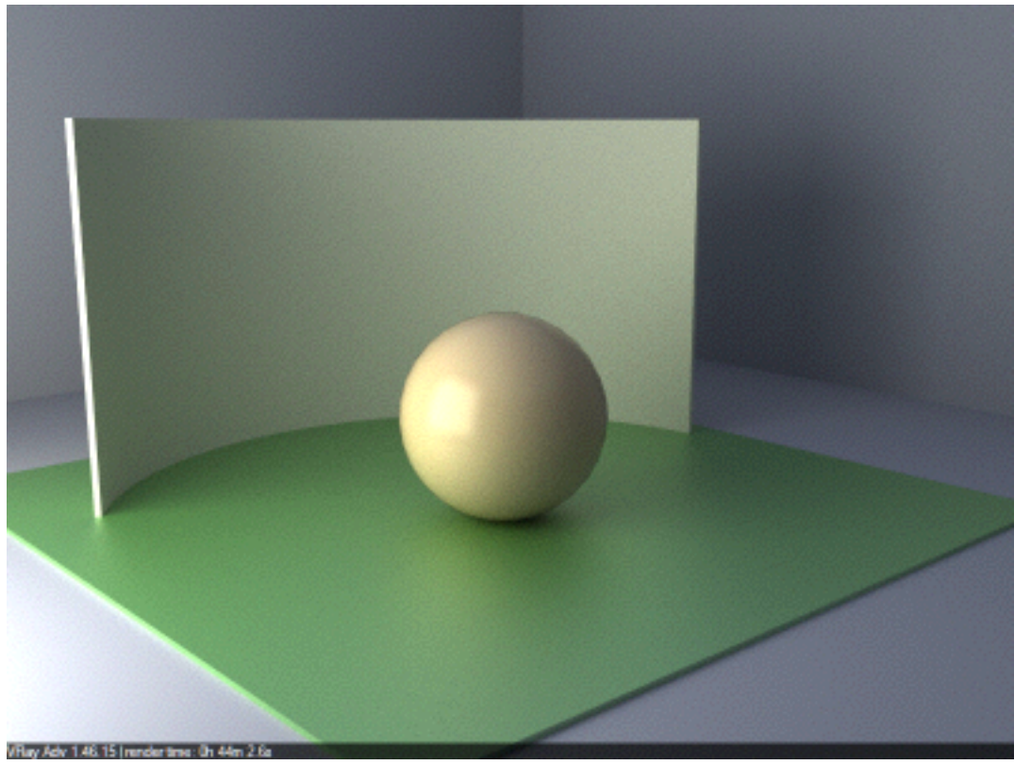
  - Numerical solvers - various algorithms; google scipy.minimize for a nice list

  - Usually easier with gradients
    e.g. Newton-Raphson

  - See also Expectation-Maximization algorithm

Monte Carlo Methods

- Collection of methods involving repeated sampling to get numerical results
    - e.g. chance of dealing two pairs in five cards?



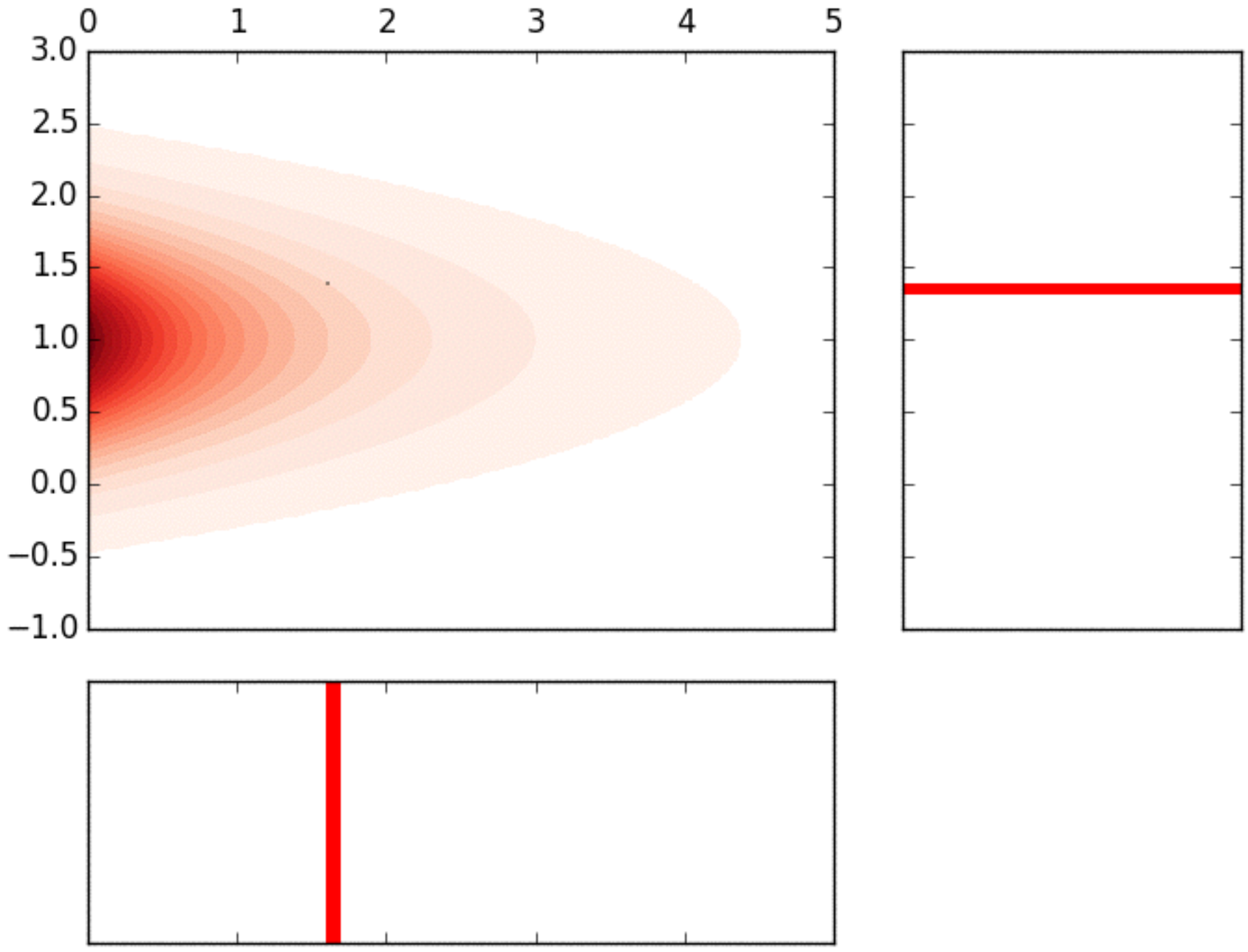
- Multi-dimensional integrals, graphics, fluid dynamics, genetics & evolution, AI, finance

Sampling Overview

- “Sample” = “Generate values with given distribution”
  - Easy: distribution written down analytically
  - Harder: can only evaluate distribution for given parameter choice

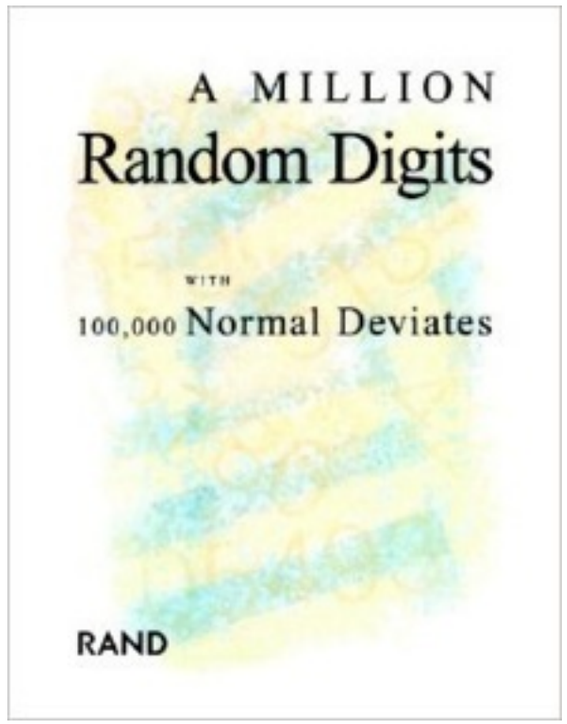
Sampling Motivation

- Simulate systems
- Short-cut to expectations if you hate maths
- Approximate & characterize distributions
    - Estimate distributions mean, std, etc.
    - Make constraint plots
    - Compare with other data


0
1
2
3
4
5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0

Direct Sampling

- Simple analytic distribution:
  can generate samples
  pseudo-randomly



- Actually incredibly complicated!

- ```
  import scipy.stats
  X = scipy.stats.poisson(5.0)
  x = X.rvs(1000)
  ```

- Be aware of random seeds
  ```

Direct Sampling

- e.g. approx solution to Lecture 1 homework problem

```
import numpy as np
dice = np.random.randint(20,size=1000000) + 1
dice_cubed = dice**3
print dice_cubed.mean()
```

- Often easier than doing sums/integrals

Monte Carlo Markov Chains

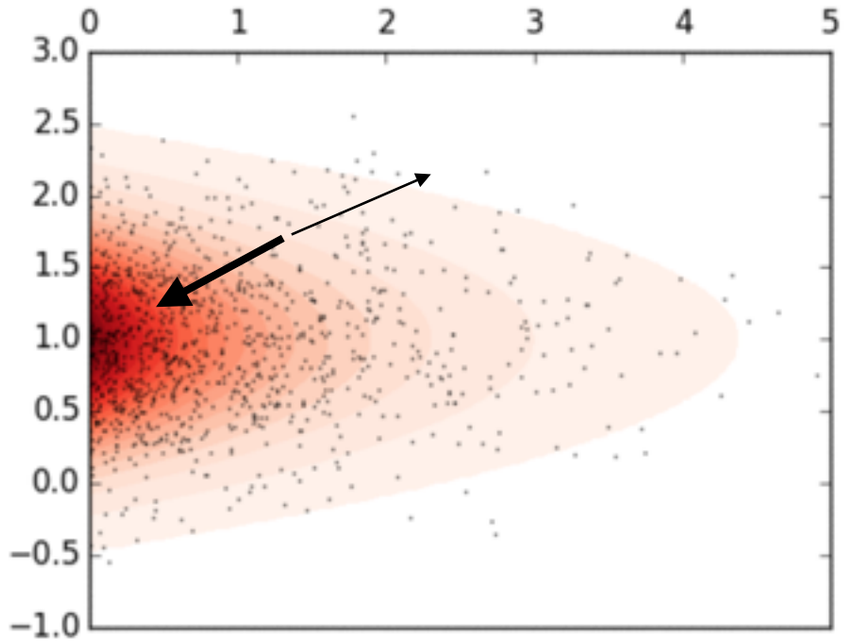
- Often cannot directly sample from a distribution
- But can evaluate P($\underline{x}$) for any given $\underline{x}$
- Markov chain = memory-less sequence

$$x_n = f(x_{n-1})$$

- MCMC: Random generation rule for f(x) which yields x_n with stationary distribution P(x)
  i.e. chain histogram tends to P

MCMC

- Jump around parameter space
- Number of jumps near x ~ P(x)



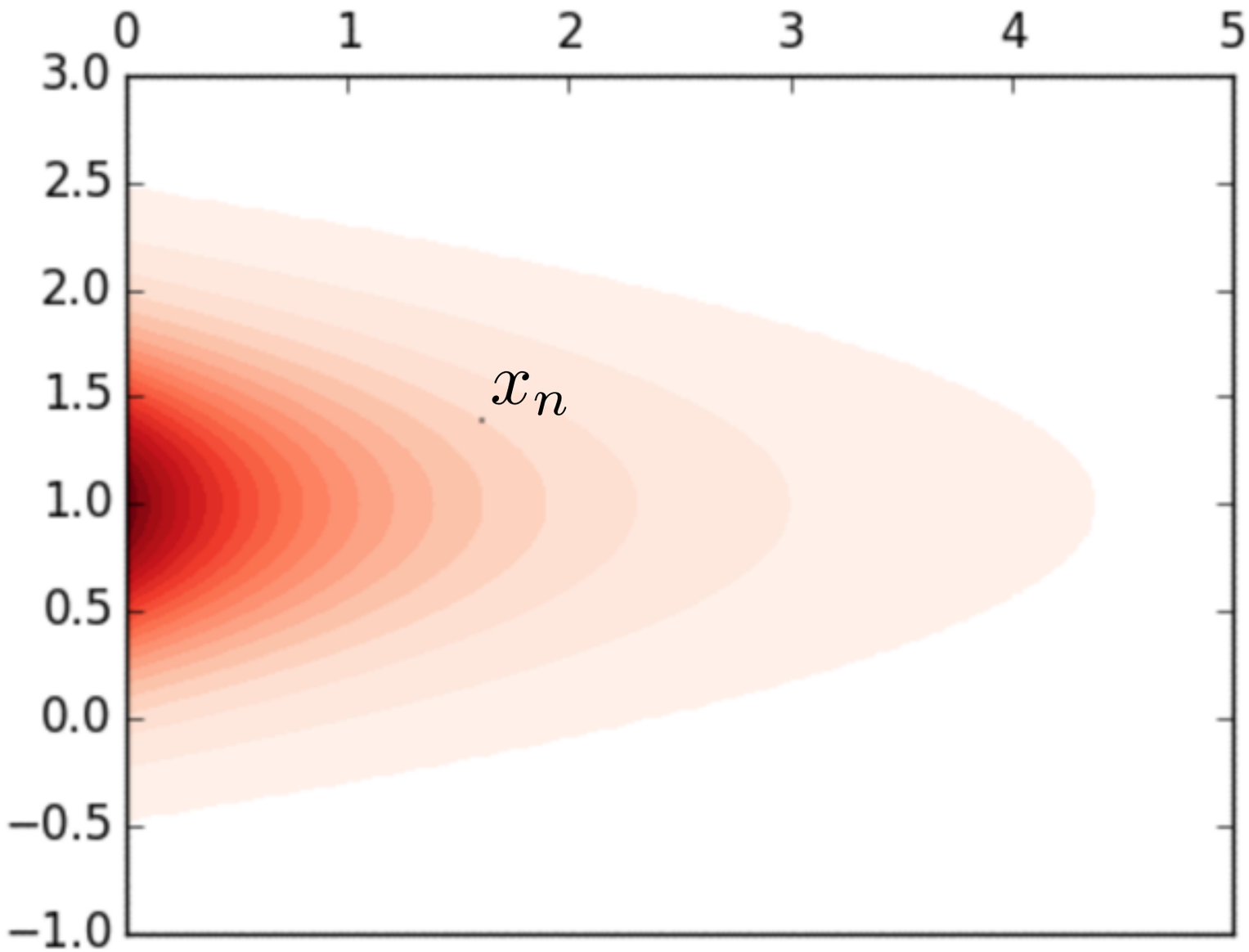
- Standard algorithms: x = single parameter set
  Advanced algorithms: x = multiple parameter sets

Metropolis-Hastings

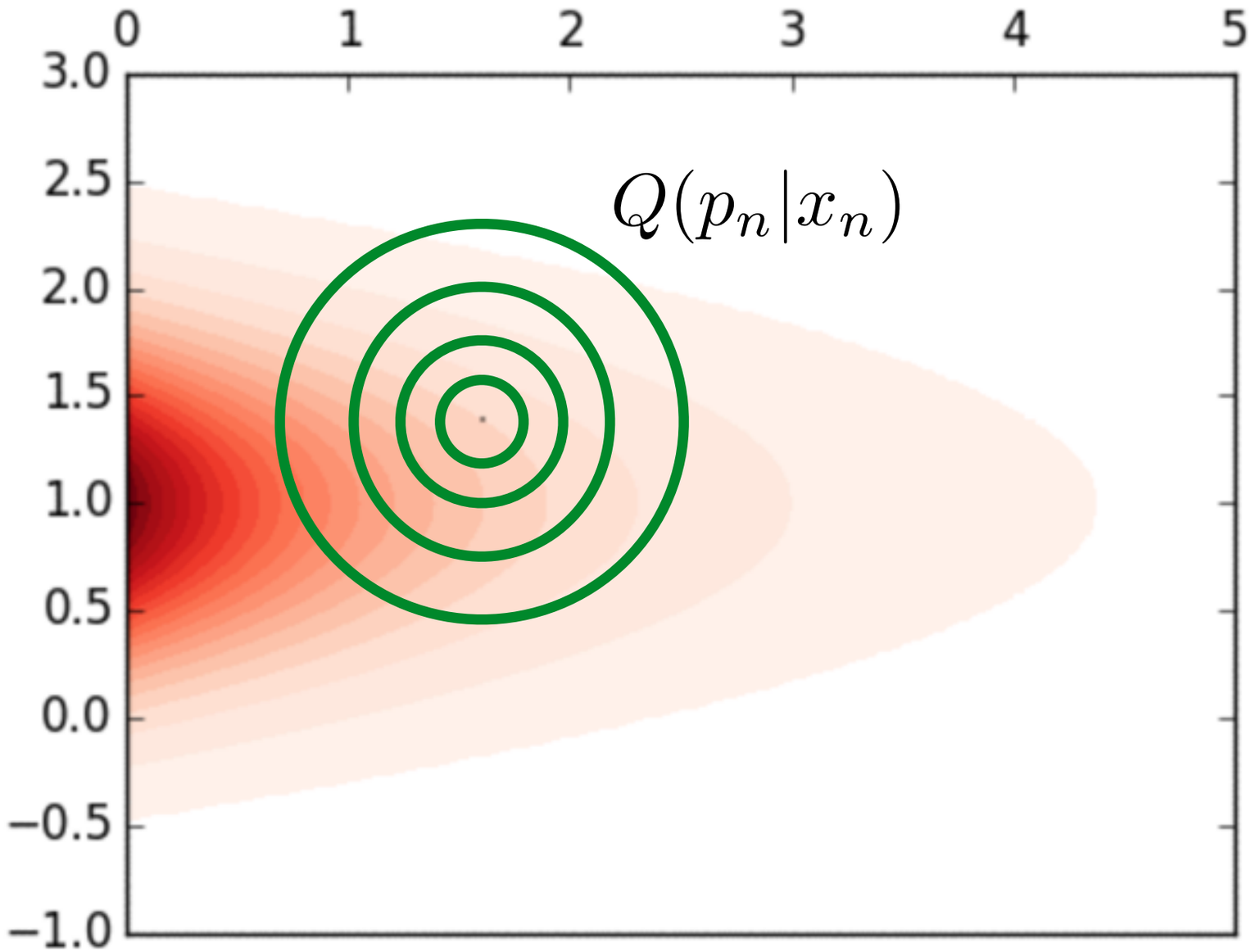
- Given current sample x_n in parameter space
  - Generate new proposed sample $p_n \sim Q(p_n|x_n)$
  - For simplest algorithm $Q(p|x) = Q(x|p)$

$$x_{n+1} = \begin{cases} p_n & \text{with probability } \max\left(\frac{P(p_n)}{P(x_n)}, 1\right) \\ x_n & \text{otherwise} \end{cases}$$

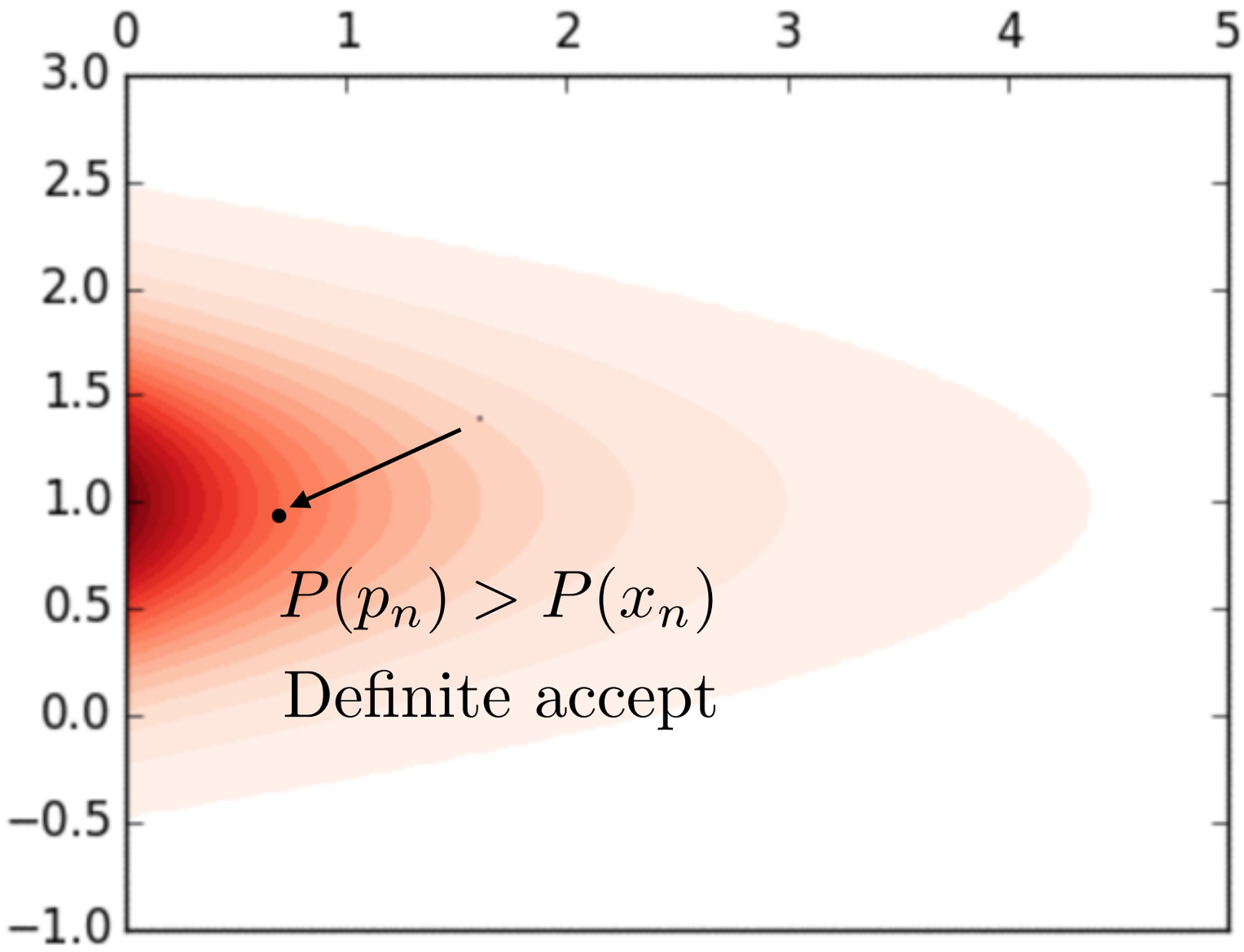
Metropolis-Hastings



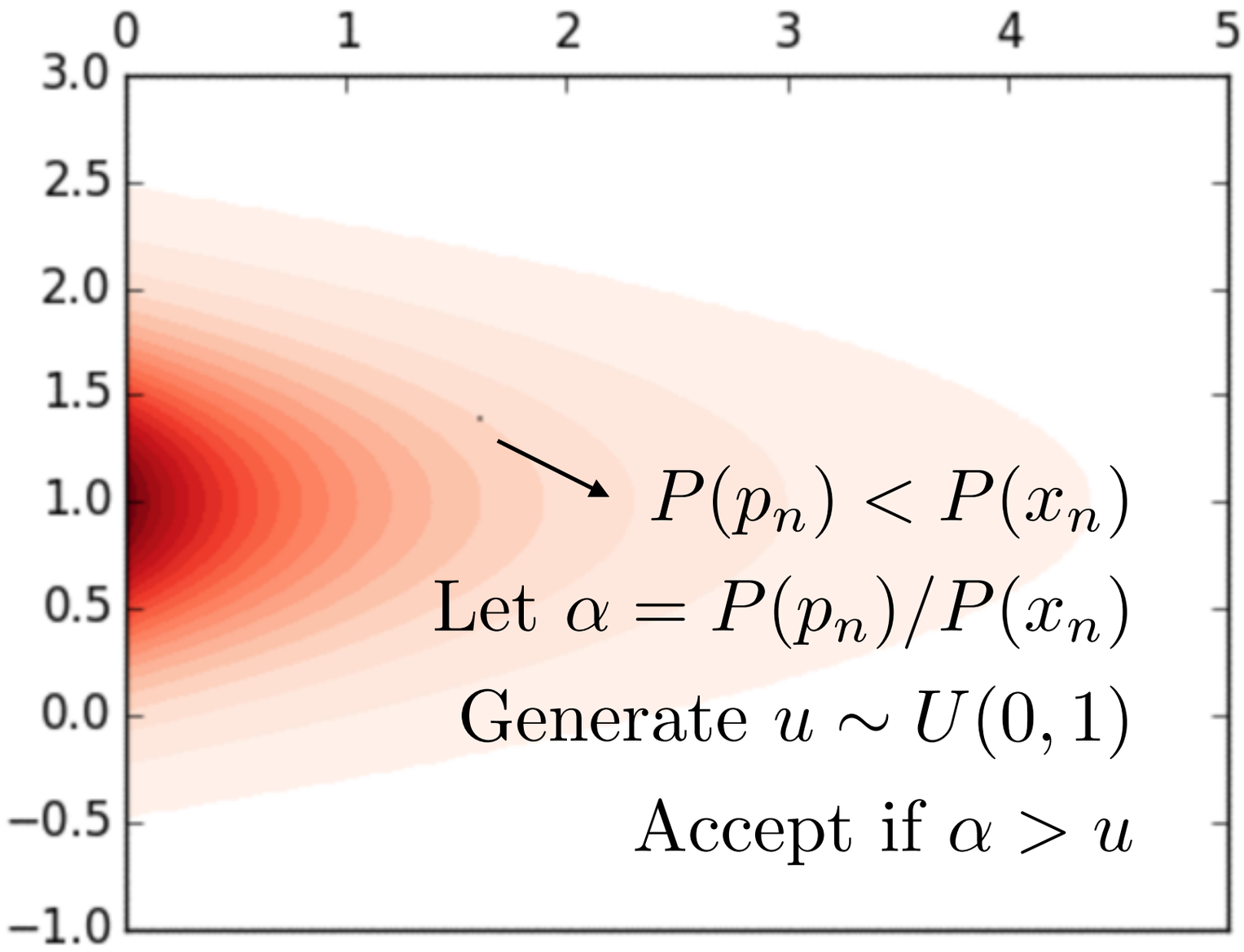
Metropolis-Hastings



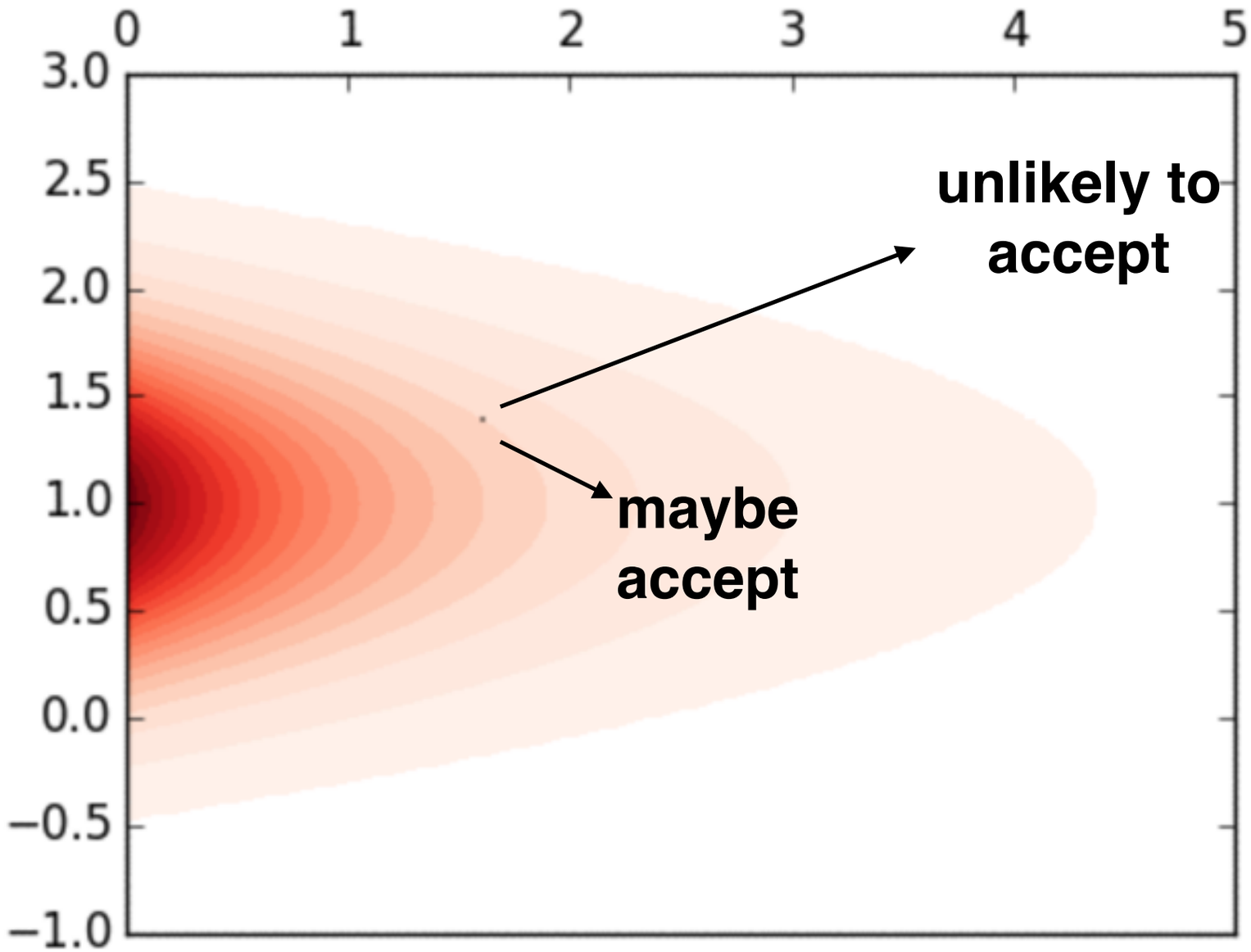
Metropolis-Hastings



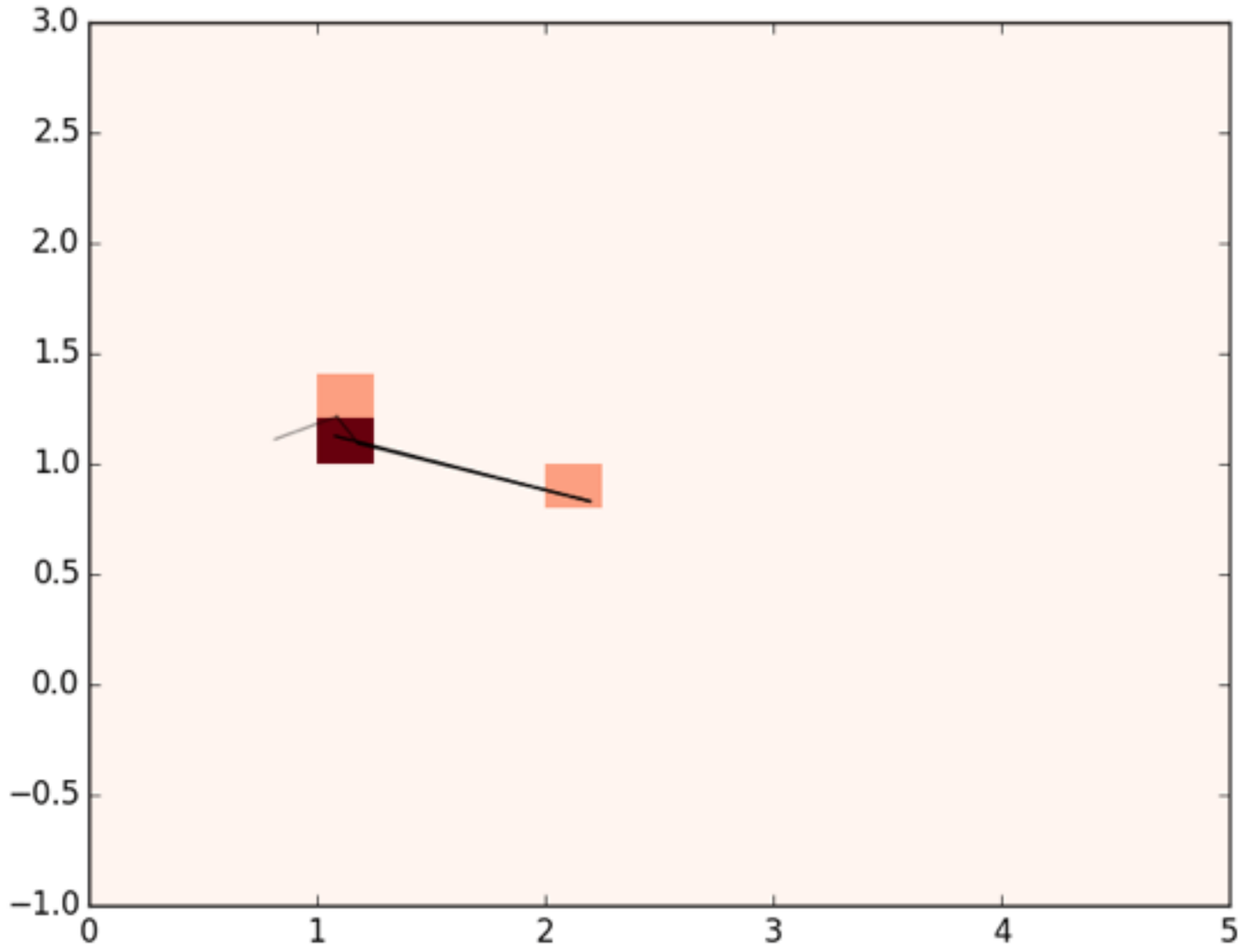
Metropolis-Hastings



Metropolis-Hastings



Metropolis-Hastings



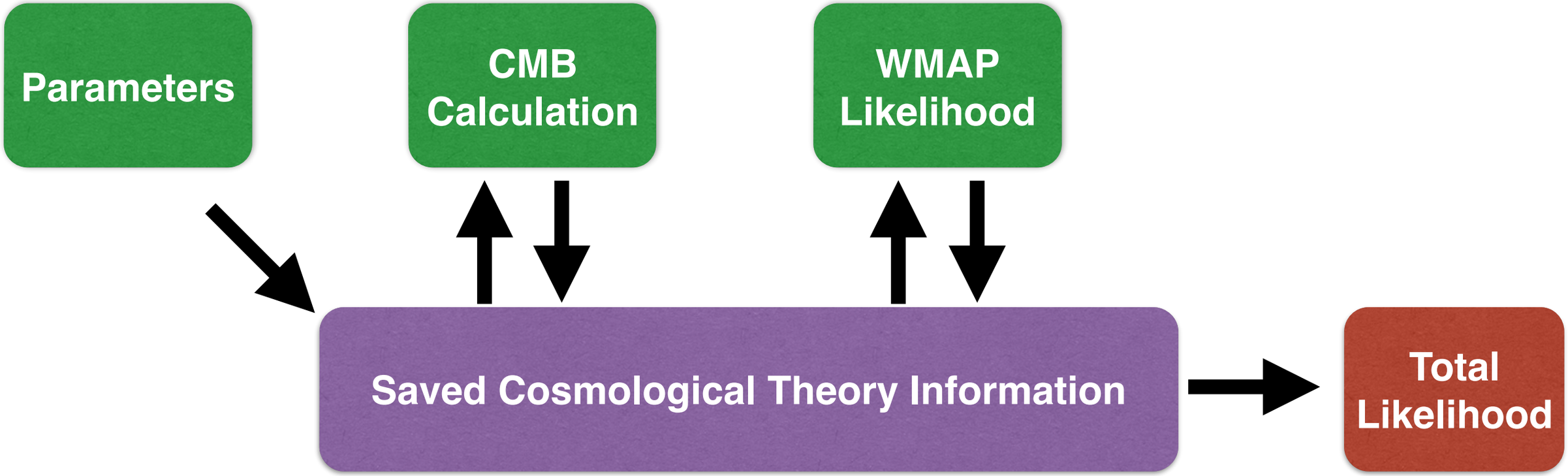
Metropolis-Hastings Proposal

- MCMC will always converge in ∞ samples
- Proposal function Q determines efficiency of MCMC
- Ideal proposal has cov(Q) ~ cov(P)
- (Multivariate) Gaussian centred on x_n usually good
  Tune σ (+covariance) to match P
- Ideal acceptance fraction ~ 0.2 - 0.3

Metropolis Example

```
> cosmosis demos/demo10.ini

> #This will take too long! Figure out how to
  change demo 7 to use Metropolis!
```



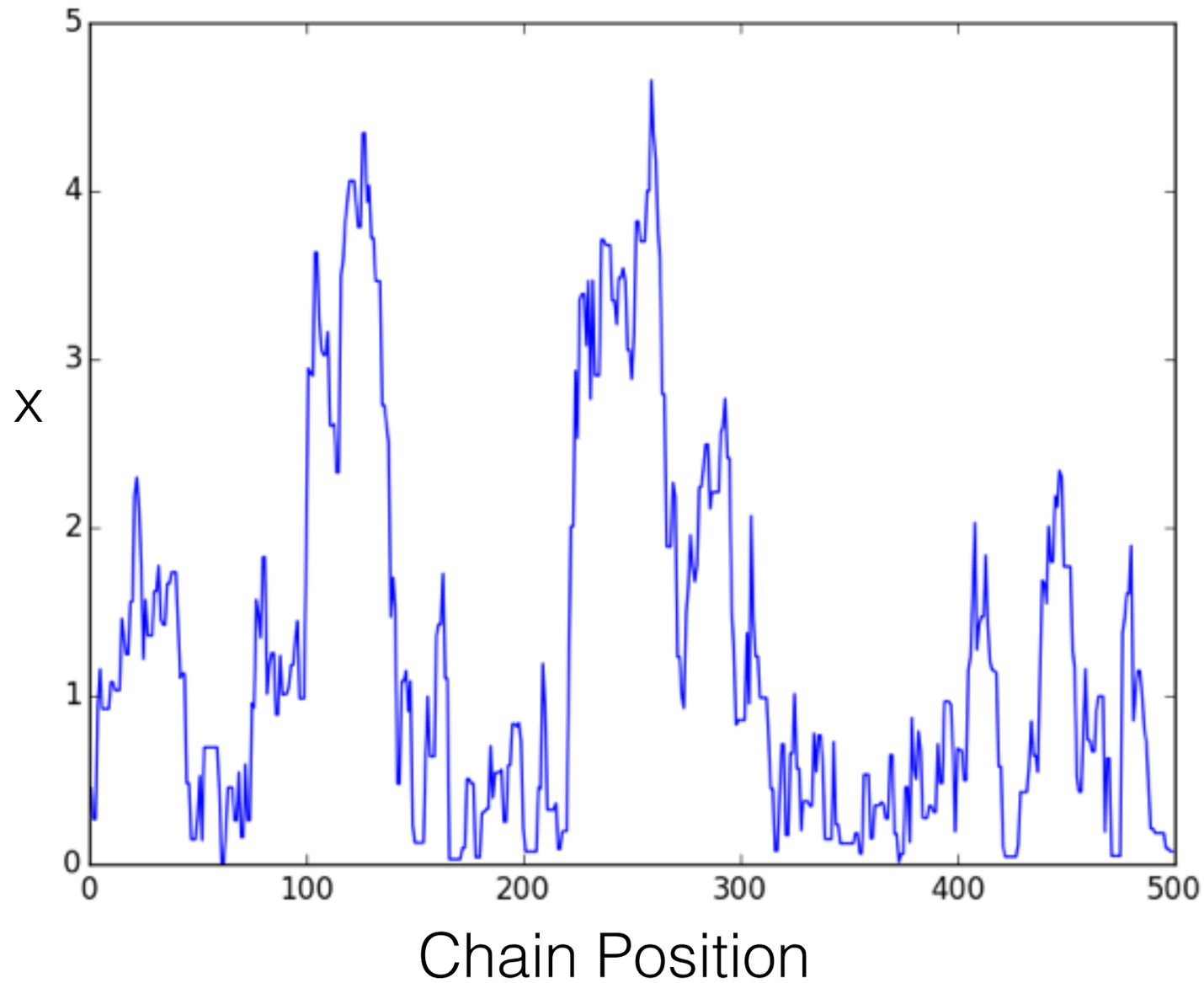
https://bitbucket.org/joezuntz/cosmosis/wiki/Demo10

Metropolis-Hastings Convergence

- Have I taken enough samples?
- Many convergence tests available
    - Easiest is visual!

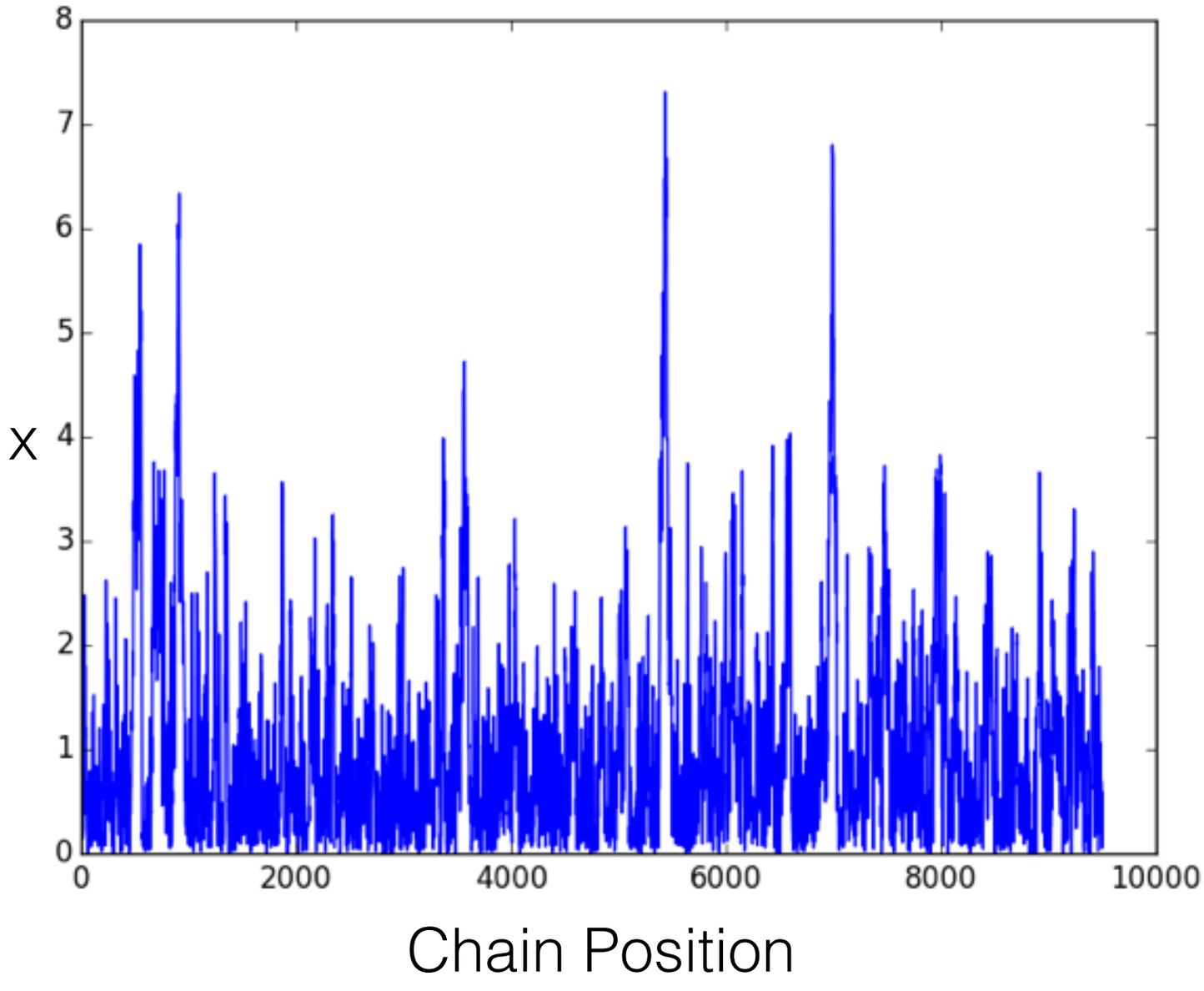
Metropolis-Hastings Convergence

- Bad Mixing
- Structure and correlations visible



Metropolis-Hastings Convergence

- Good Mixing
- Looks like noise
- Must be true for all parameters in space



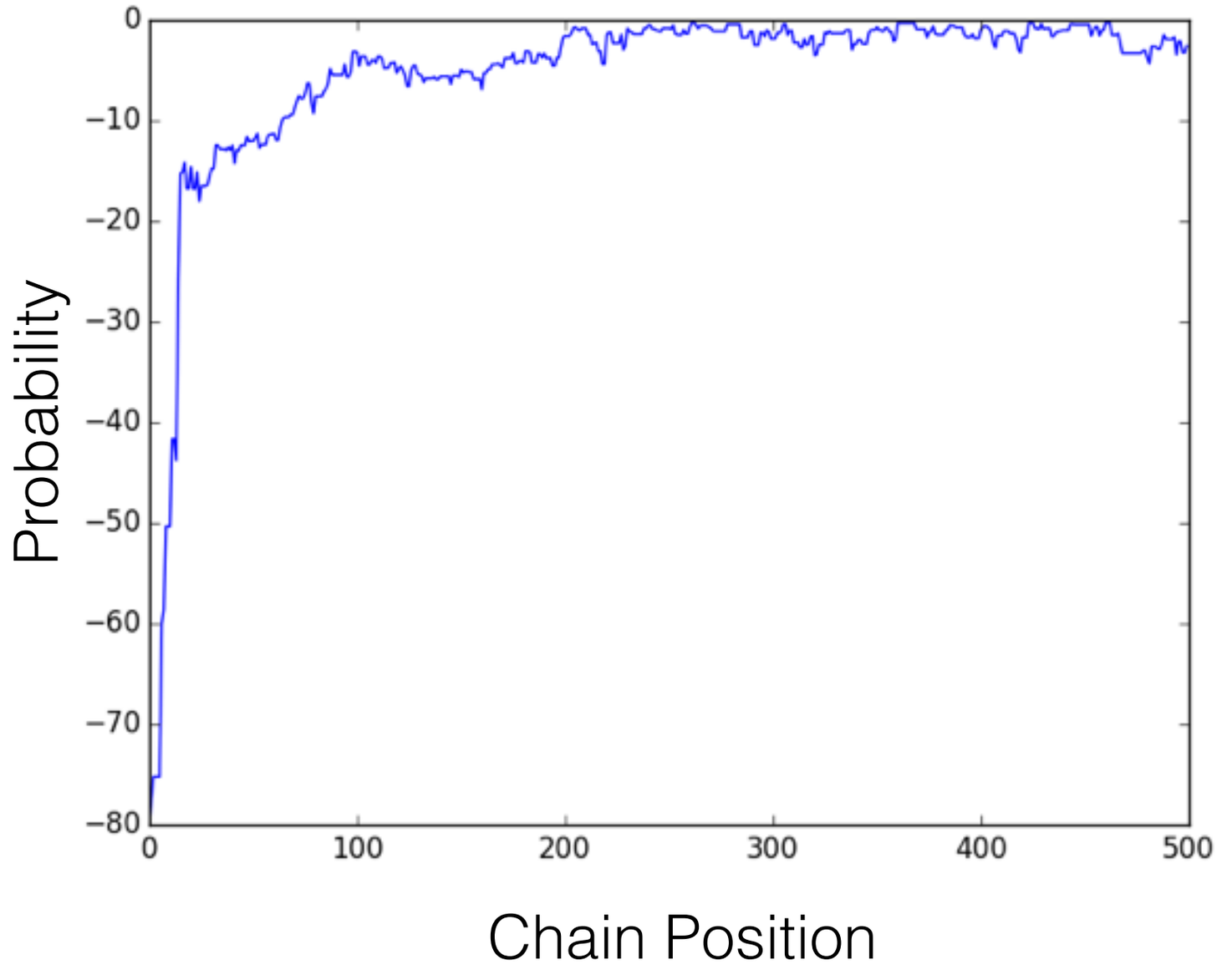
Metropolis-Hastings Convergence

- Run several chains and compare results
    - (Variance of means) / (Mean of variances)
      Less than e.g. 3%

Metropolis-Hastings Burn in

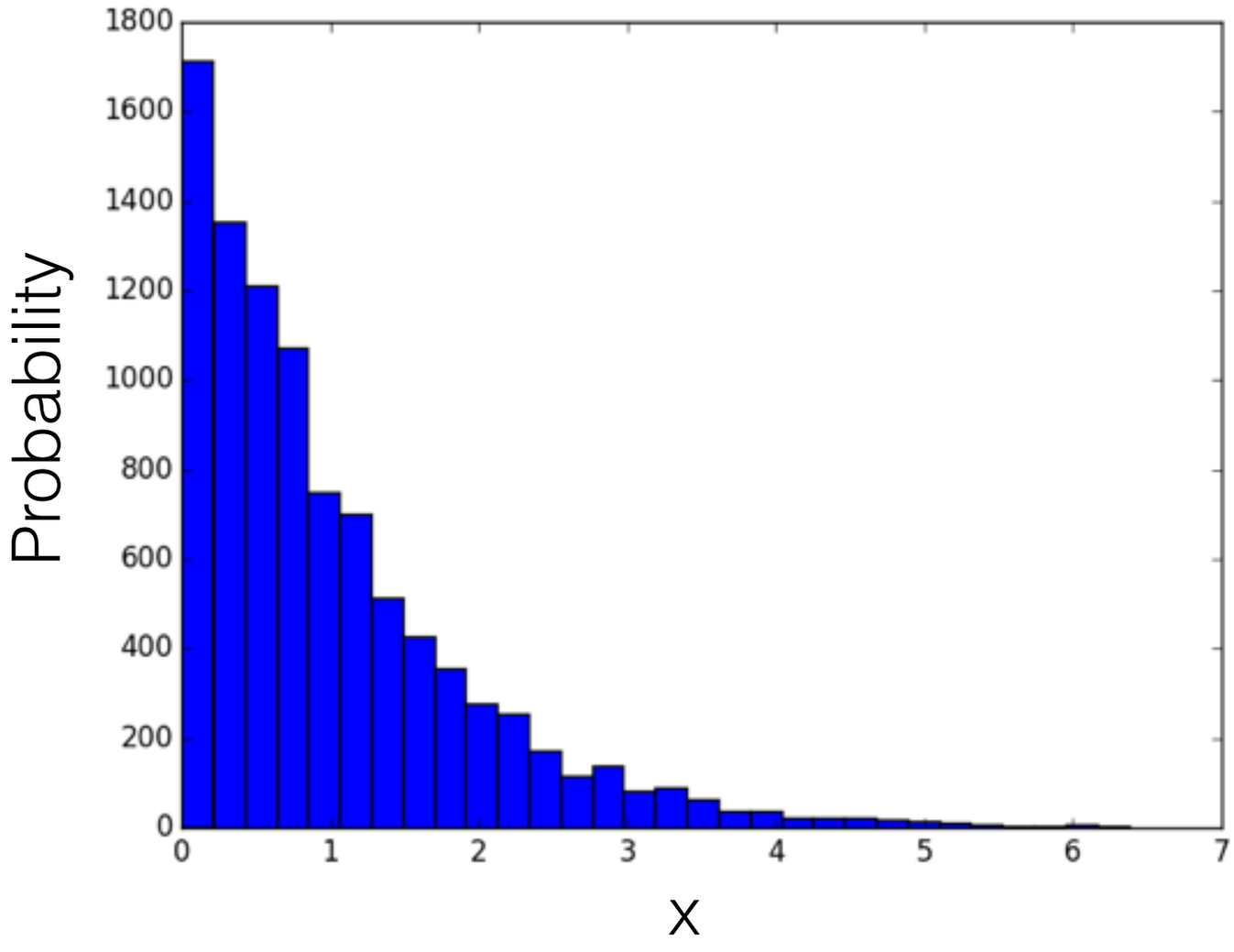
- Chain will explore peak only after finding it

- Cut off start of chain



Metropolis-Hastings Analysis

- Probability proportional to chain multiplicity
- Histogram chain, in 1D or 2D
- Can also thin e.g. remove every other sample



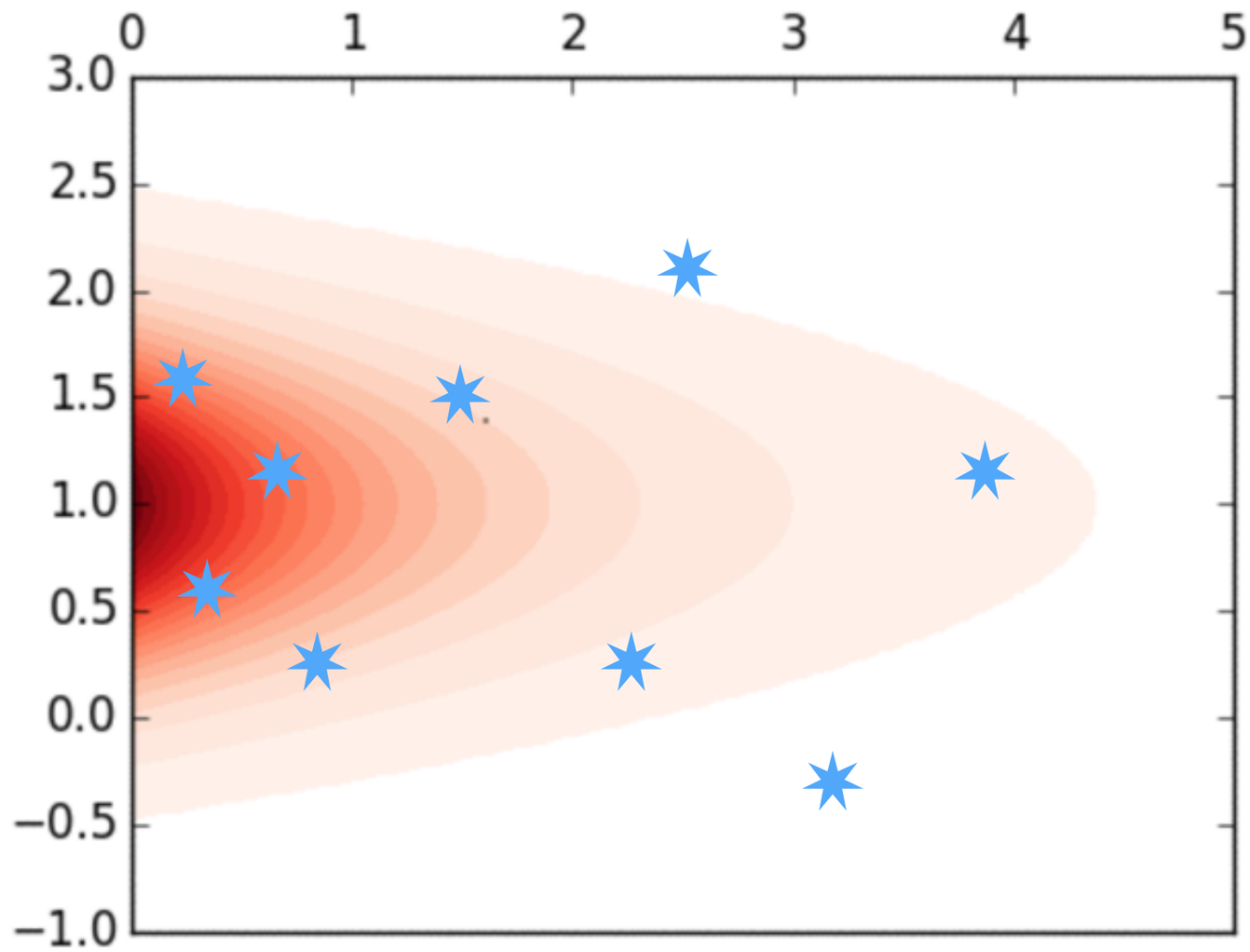
Metropolis-Hastings Analysis

- Can also get expectations of derived parameters

$$E[f(x)] = \int P(x) f(x) \mathrm{d}x \approx \frac{1}{N} \sum_i f(x_i)$$

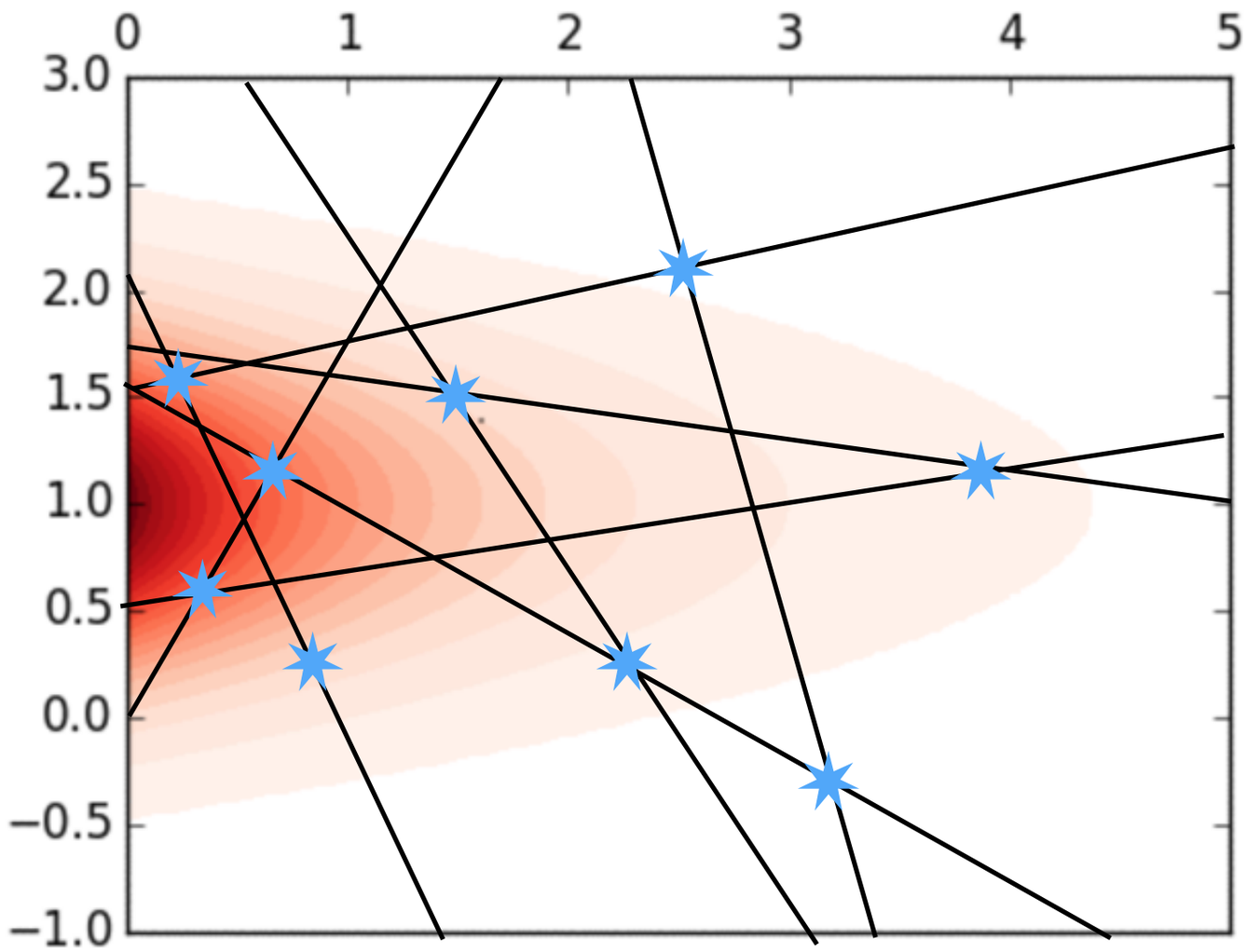
Emcee Sampling

- Goodman & Weare algorithm
- Group of live “walkers” in parameter space
- Parallel update rule on connecting lines - affine invariant
- Popular in astronomy for nice and friendly python package



Emcee Sampling

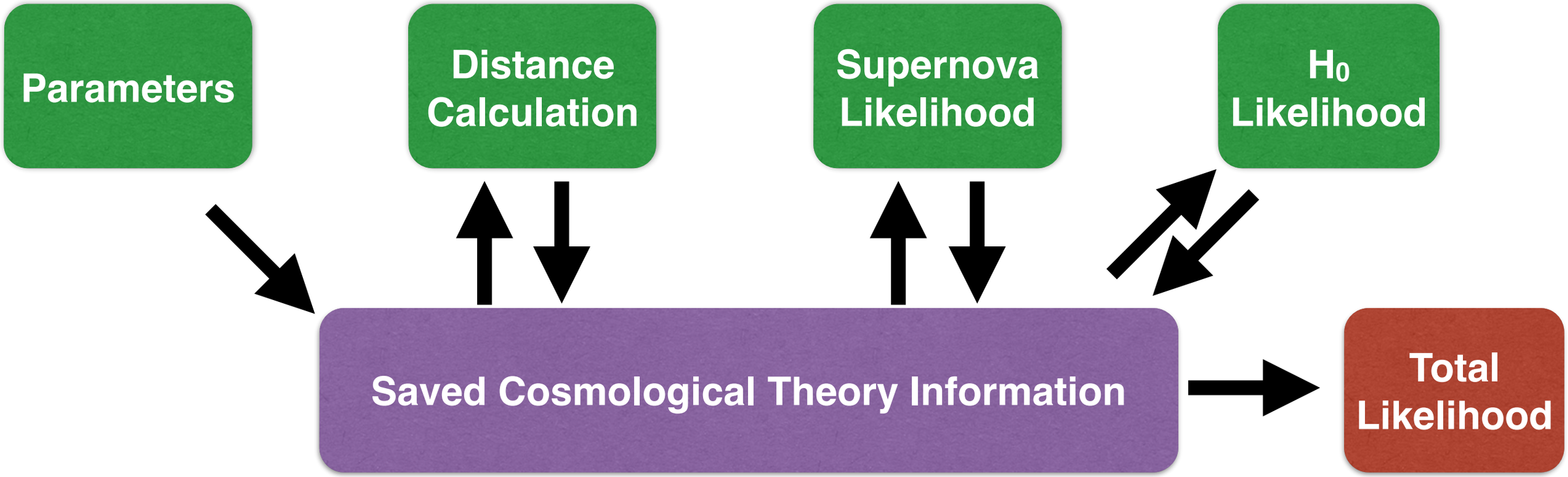
- Goodman & Weare algorithm
- Group of live “walkers” in parameter space
- Parallel update rule on connecting lines - affine invariant
- Popular in astronomy for nice and friendly python package



Emcee Example

```
> cosmosis demos/demo5.ini

> postprocess demos/demo5.ini -o plots -p demo5
```



https://bitbucket.org/joezuntz/cosmosis/wiki/Demo5

Model Selection

- Given two models, how can we compare them?
- Simplest approach = compare ML
    - Does not include uncertainty or Occam's Razor
- Recall that all our probabilities have been conditional on the model, as in Bayes:

$$P(p|M) = \frac{P(d|pM)P(p|M)}{P(d|M)}$$

Model Selection: Bayesian Evidence

- Can use Bayes Theorem again, on model level:

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

- Only really meaningful when comparing models:

$$\frac{P(M_1|d)}{P(M_2|d)} = \frac{P(d|M_1)}{P(d|M_2)} \frac{P(M_1)}{P(M_2)}$$

Model Priors

Bayesian Evidence Values

Model Selection: Bayesian Evidence

- Likelihood of parameters within model:

$$P(d|pM)$$

- Evidence of model:

$$P(d|p)$$

Model Selection: Bayesian Evidence

- Evidence is the bit we ignored before when doing parameter estimation
- Given by an integral over prior space

$$P(d|M) = \int P(d|pM)P(p|M)\mathrm{d}p$$

- Hard to evaluate - posterior usually small compared to prior

Model Selection: Evidence Approximations

- Nice evidence approximations for some cases:
  - Savage-Dickey Density ratio
    (for when one model is a subset of another)

  - Akaike information criterion AIC
    Bayesian information criterion BIC
    Work in various circumstances

Model Selection: Nested Sampling

$$\int L(\theta)p(\theta)\mathrm{d}\theta = \int L(X)\mathrm{d}X$$

$$\approx \sum L_i \delta X_i$$

$$\mathrm{d}X \equiv P(\theta)\mathrm{d}\theta$$

X = remaining prior volume

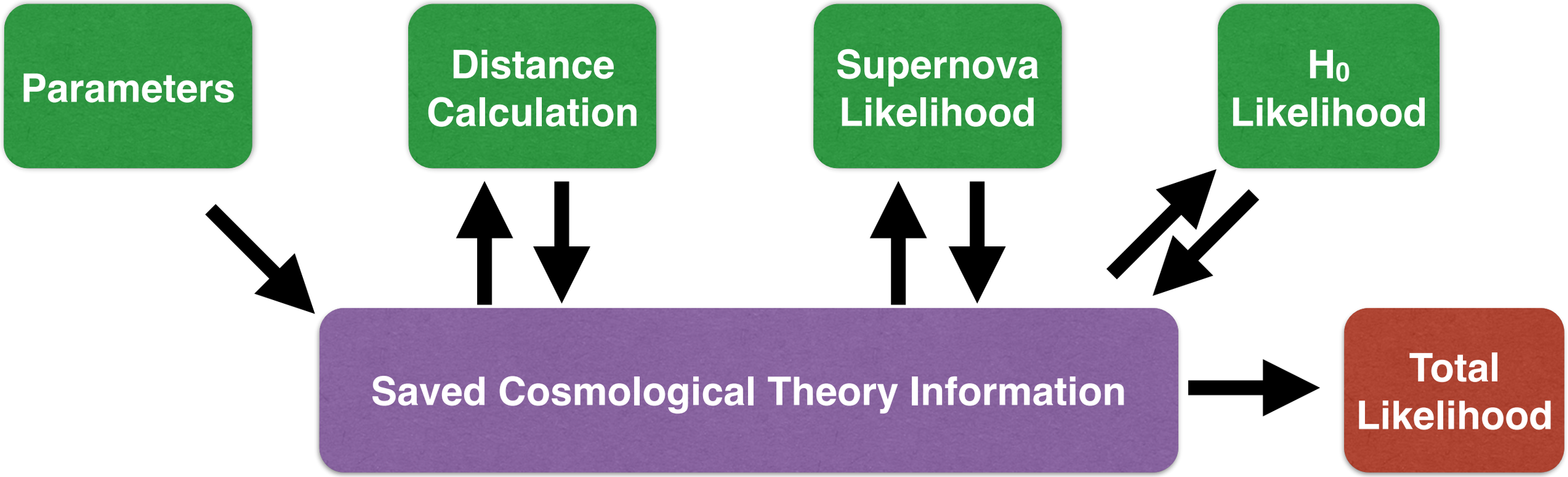
Model Selection: Nested Sampling

- Also uses ensemble of live points
    - Computes constraints as well as evidence
- Each iteration, replace lowest likelihood point with one higher up
    - By cleverly sampling into envelope of active points
- Multinest software is extremely clever
    - C, F90, Python bindings

Multinest Example

```
> cosmosis demos/demo9.ini

> postprocess -o plots -p demo9 demos/demo9.ini
  --extra demos/extra9.py
```



https://bitbucket.org/joezuntz/cosmosis/wiki/Demo9

Exercise

- With a prior s ~ N(0.5, 0.1) and other priors of your choice, write a Metropolis Hastings sampler to draw from the LMC Cepheid likelihood. Plot 1D and 2D constraints on the parameters.