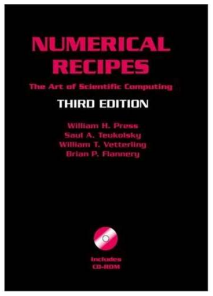
9. Monte Carlo Simulation Methods

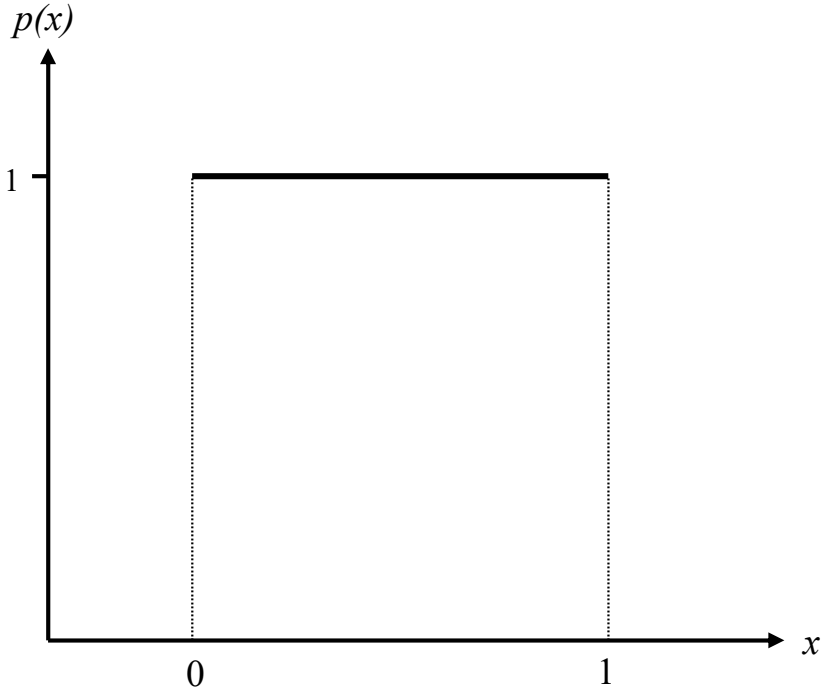
9.1 Uniform random numbers

Generating uniform random numbers, drawn from the pdf U[0,1], is fairly easy. Any scientific Calculator will have a RAN function…

Better examples of U[0,1] random number generators can be found in Numerical Recipes.

http://www.numerical-recipes.com/

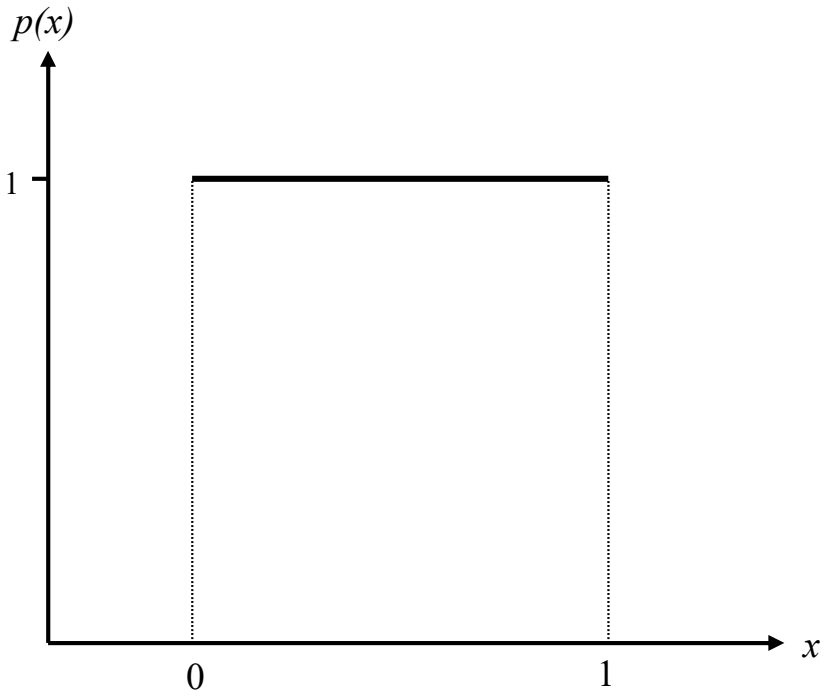




9.1 Uniform random numbers

Algorithms only generate **pseudo-random** numbers: very long (deterministic) sequences of numbers which are approximately random (i.e. no discernible pattern).

The better the RNG, the better it approximates U[0,1]

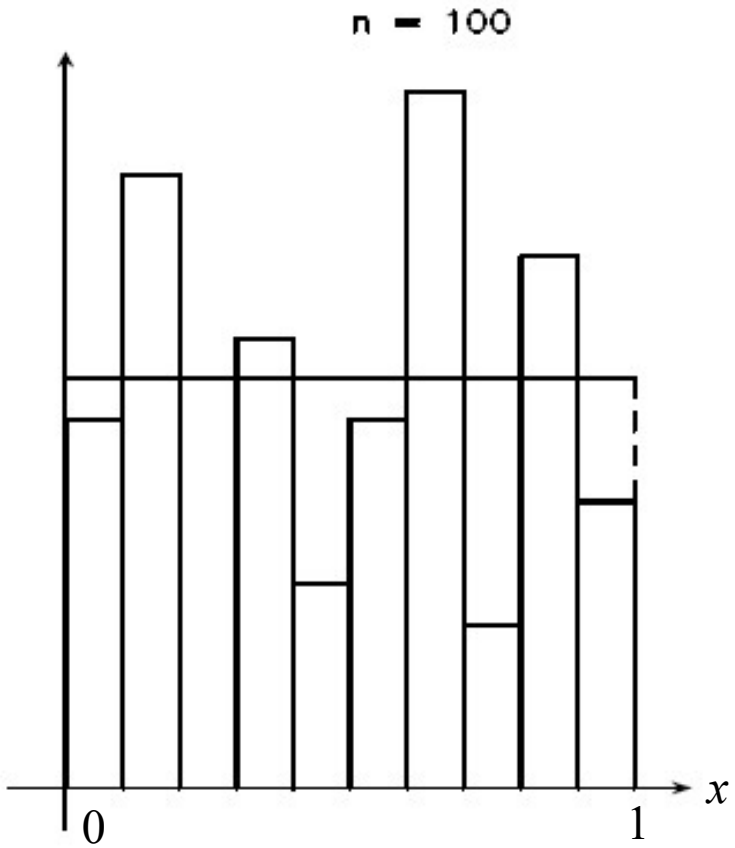


We can test pseudo-random numbers for randomness in several ways:

(a) Histogram of sampled values.

We can use hypothesis tests to see if the sample is consistent with the pdf we are trying to model.

e.g. Chi-squared test, applied to the
to the numbers in each histogram bin.



We can test pseudo-random numbers for randomness in several ways:

(a) Histogram of sampled values.

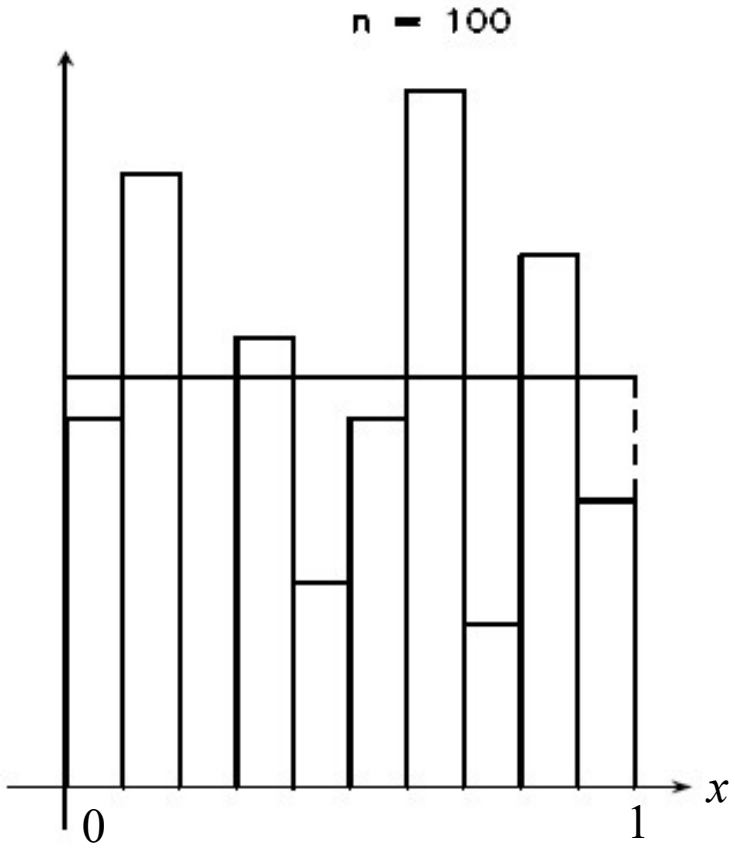
We can use hypothesis tests to see if the sample is consistent with the pdf we are trying to model.

e.g. Chi-squared test, applied to the to the numbers in each histogram bin.

$$\chi^2 = \sum_{i=1}^{n_{\text{bin}}} \left(\frac{n_i^{\text{obs}} - n_i^{\text{pred}}}{\sigma_i} \right)^2$$

Assume the bin number counts are subject to Poisson fluctuations, so that $\sigma_i^{\,2} = n_i^{\text{pred}}$

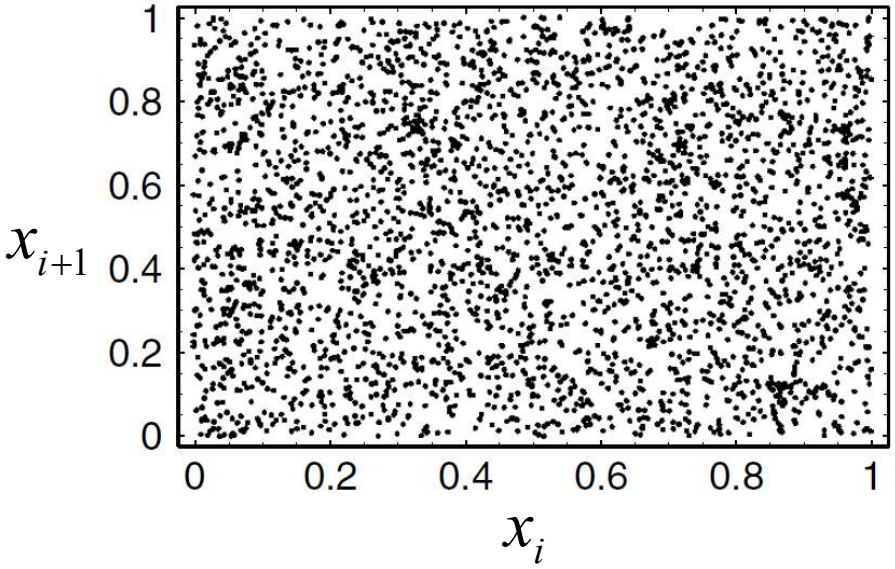
Note: no. of degrees of freedom $= n_{\text{bin}} - 1$ since we know the total sample size.

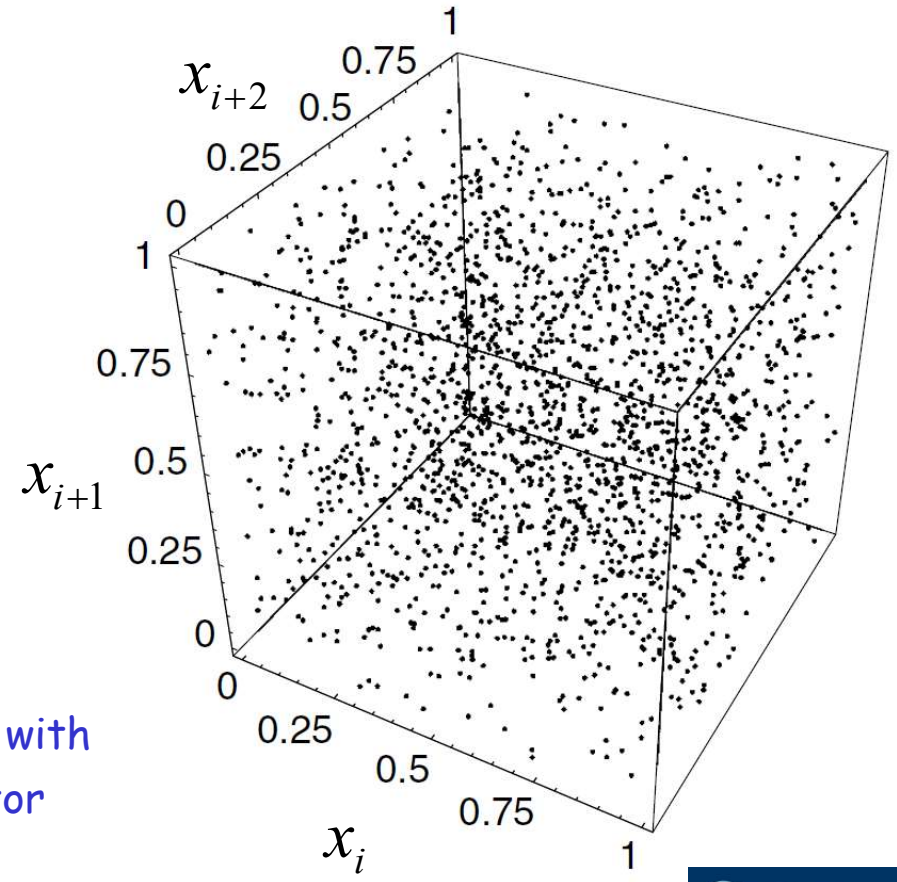


(b) Correlations between neighbouring pseudo-random numbers

Sequential patterns in the sampled values would show up as structure in the phase portraits – scatterplots of the i^{th} value against the $(i+1)^{th}$ value etc.

(see Gregory, Chapter 5)





This method exposed a problem with RANDU random number generator

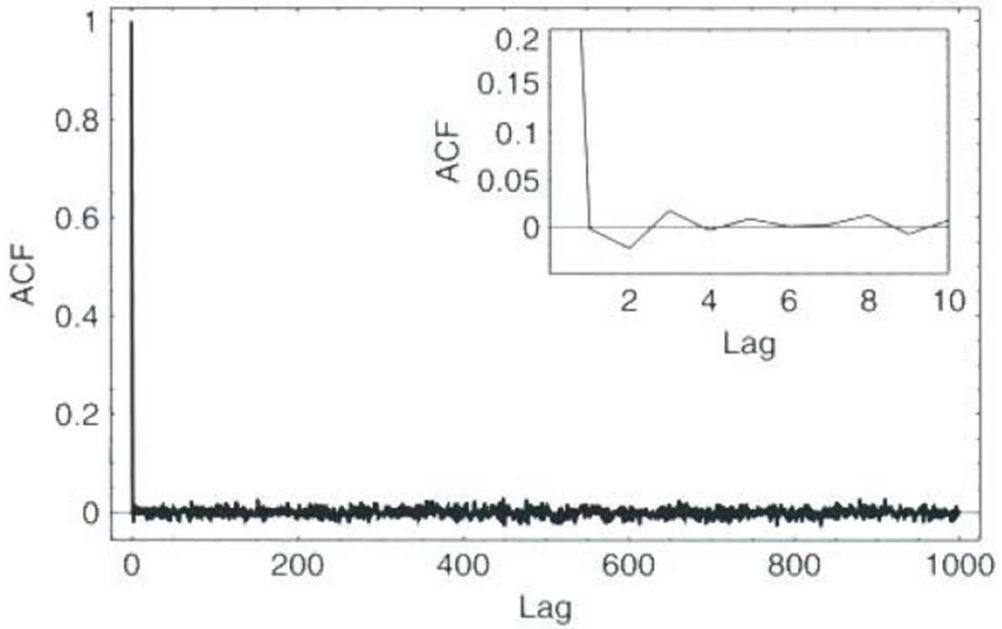
(b) Correlations between neighbouring pseudo-random numbers

Sequential patterns in the sampled values would show up as structure in the phase portraits – scatterplots of the i^{th} value against the $(i+1)^{th}$ value etc.

We can compute the Auto-correlation function

$$\rho(j) = \frac{\sum\left[(x_i - \bar{x})(x_{i+j} - \bar{x})\right]}{\sqrt{\sum (x_i - \bar{x})^2}\sqrt{\sum (x_{i+j} - \bar{x})^2}}$$

j is known as the Lag



If the sequence is uniformly random, we expect

$$\begin{cases} \rho(j) = 1 & \text{for } j = 0 \\ \rho(j) = 0 & \text{otherwise} \end{cases}$$

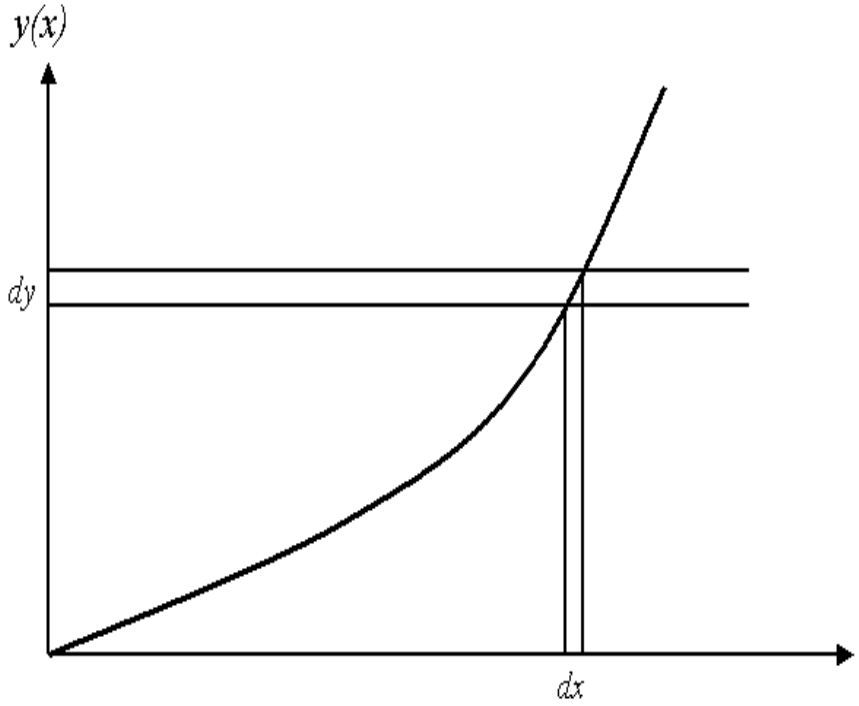
9.2 Variable transformations

Generating random numbers from other pdfs can be done by transforming random numbers drawn from simpler pdfs.

The procedure is similar to changing variables in integration.

Suppose, e.g. $x \sim p(x)$

Let $y = y(x)$ be **monotonic**



Then $$p(y)dy = p(x)dx$$

Probability of number between y and $y+dy$

Probability of number between x and $x+dx$

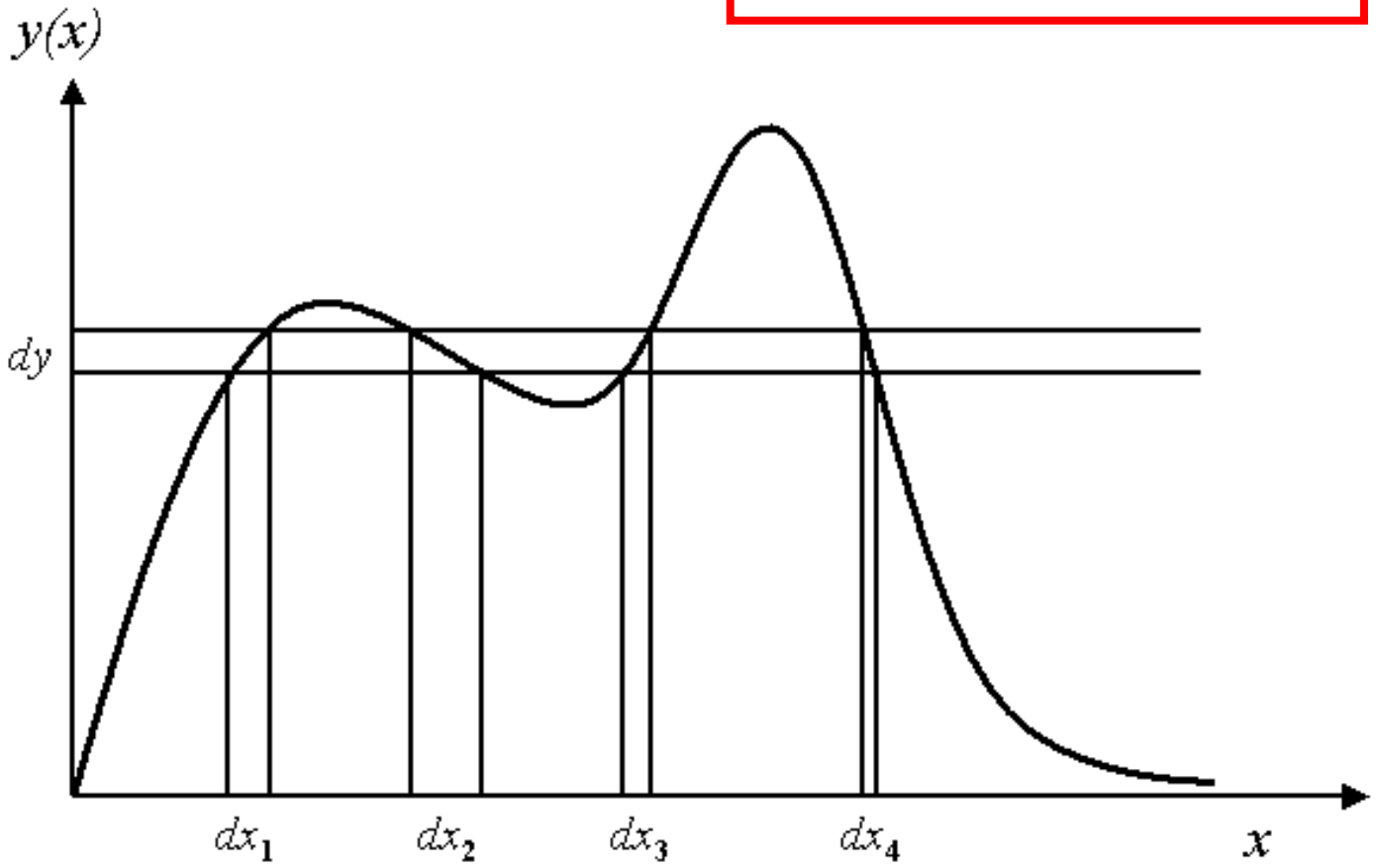
$$p(y) = \frac{p(x(y))}{|dy/dx|}$$

Because probability must be positive

We can extend the expression given previously to the case where $y(x)$ is not monotonic, by calculating

$$p(y)dy = \sum_i p(x_i)dx_i \quad \text{so that}$$

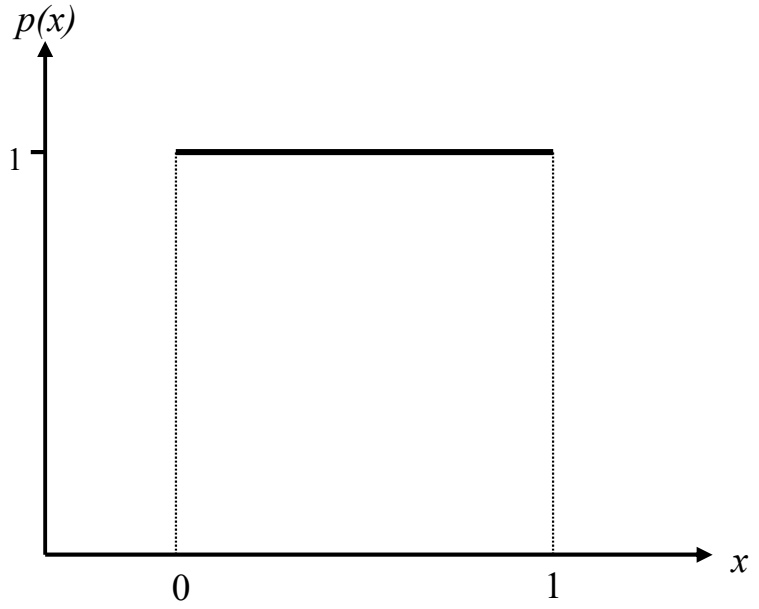
$$p(y) = \sum_i \frac{p(x_i(y))}{|dy/dx_i|}$$



Example 1

Suppose we have $x \sim U[0,1]$

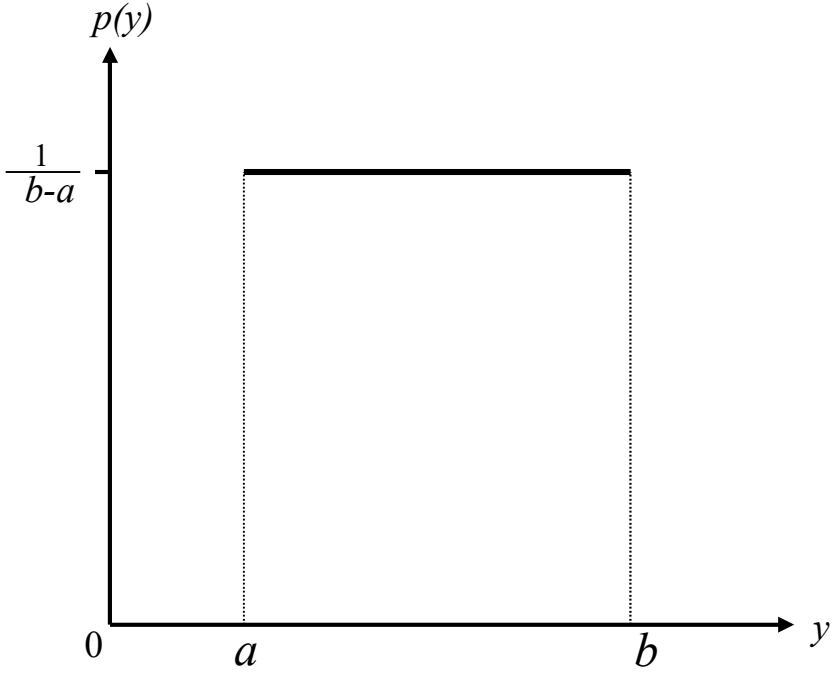
Then $p(x) = \begin{cases} 1 & \text{for} \quad 0 < x < 1 \\ 0 & \text{otherwise} \end{cases}$



Define $y = a + (b - a)x$

So $\frac{dy}{dx} = (b - a)$

i.e. $p(y) = \begin{cases} 1/b-a & \text{for} \quad a < y < b \\ 0 & \text{otherwise} \end{cases}$



or $y \sim U[a,b]$

Normal pdf with mean zero and standard deviation unity

Example 2

Numerical recipes provides a program to turn $x \sim U[0,1]$ into $y \sim N[0,1]$

Suppose we want $z \sim N[\mu, \sigma]$

We define $z = \mu + \sigma y$ so that $\frac{dz}{dy} = \sigma$

Now $$p(y) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2} y^2\right)$$

so $$p(z) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2}\left[\frac{z-\mu}{\sigma}\right]^2\right)$$

Question 18: If $x \sim U[0,1]$ and $y = -\ln x$, the pdf of y is

A $p(y) = e^{y}$

B $p(y) = e^{-y}$

C $p(y) = -\ln y$

D $p(y) = \ln y$

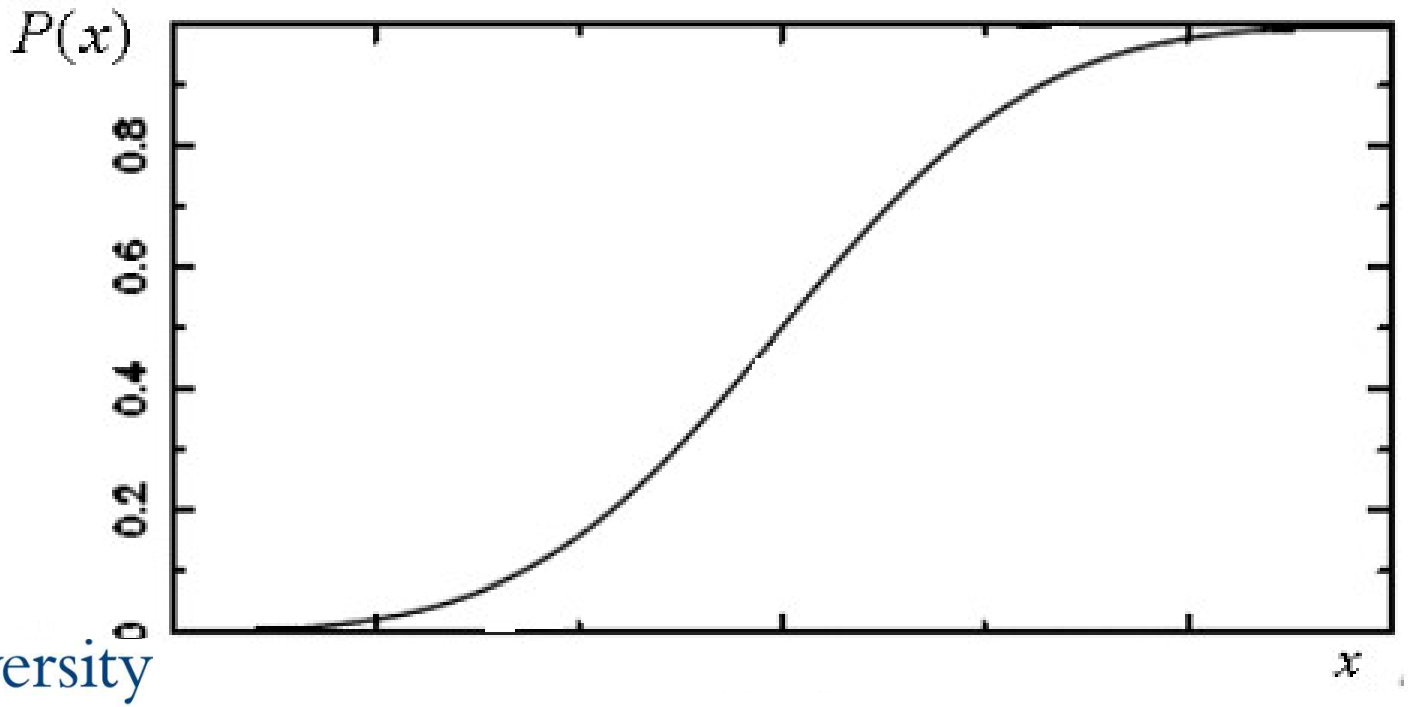
9.3 Probability integral transform

One particular variable transformation merits special attention.

Suppose we can compute the CDF of some desired random variable

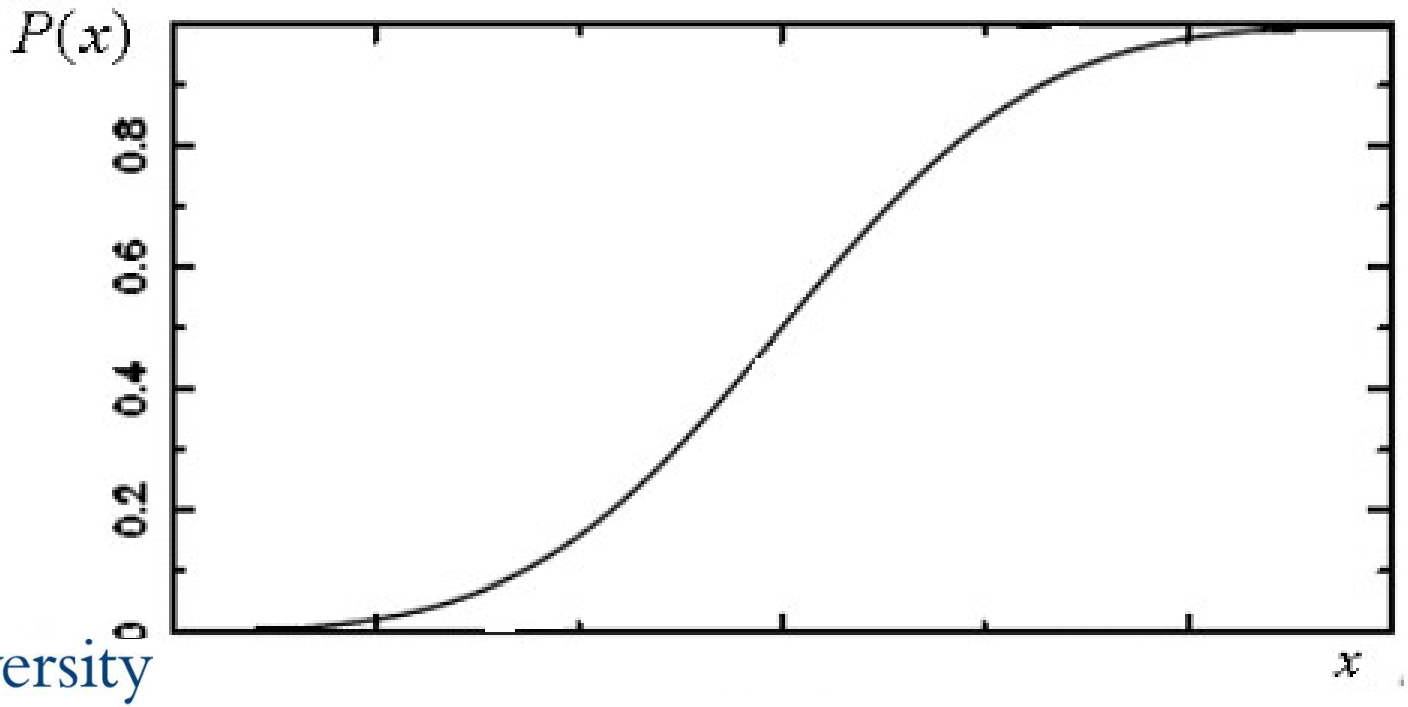
Cumulative distribution function (CDF)

$$P(a) = \int_{-\infty}^{a} p(x)dx = \text{Prob}(x < a)$$



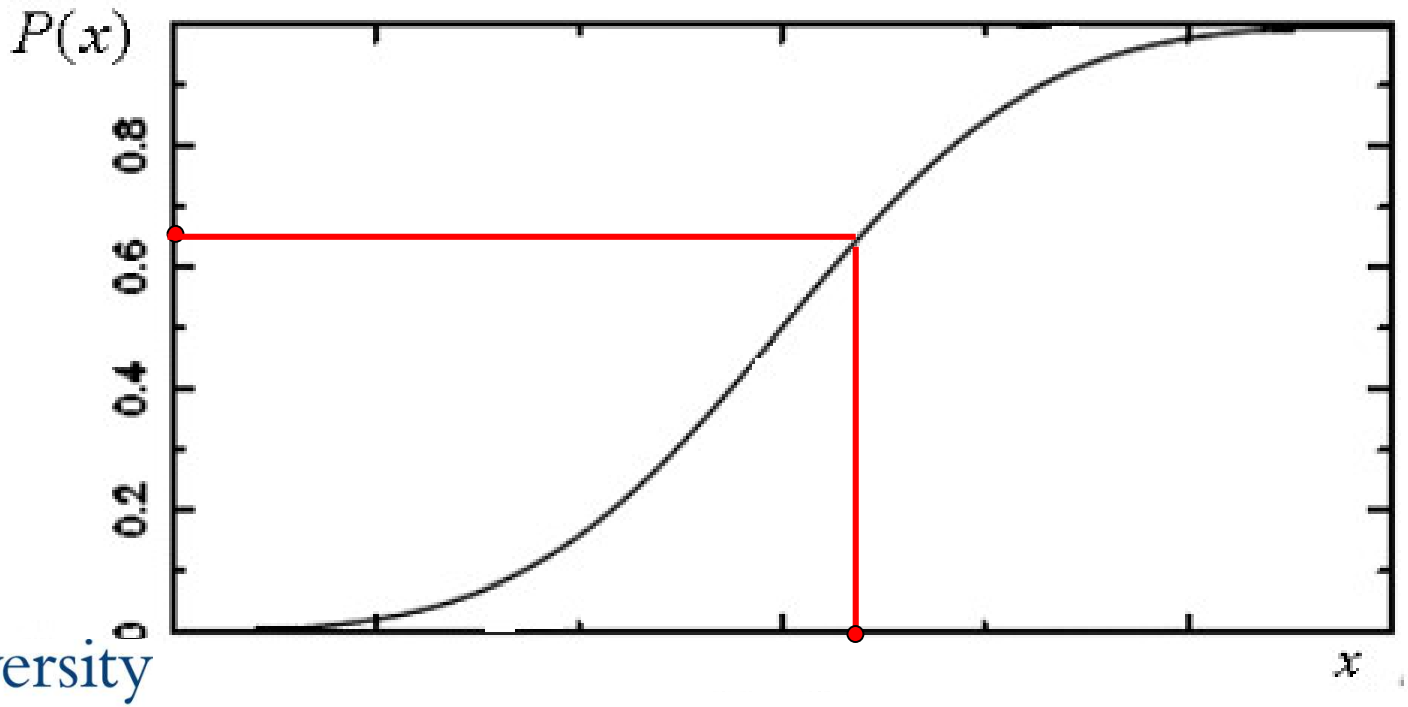
1) Sample a random variable $y \sim U[0,1]$

2) Compute x such that $y = P(x)$ i.e. $x = P^{-1}(y)$



1) Sample a random variable $y \sim U[0,1]$

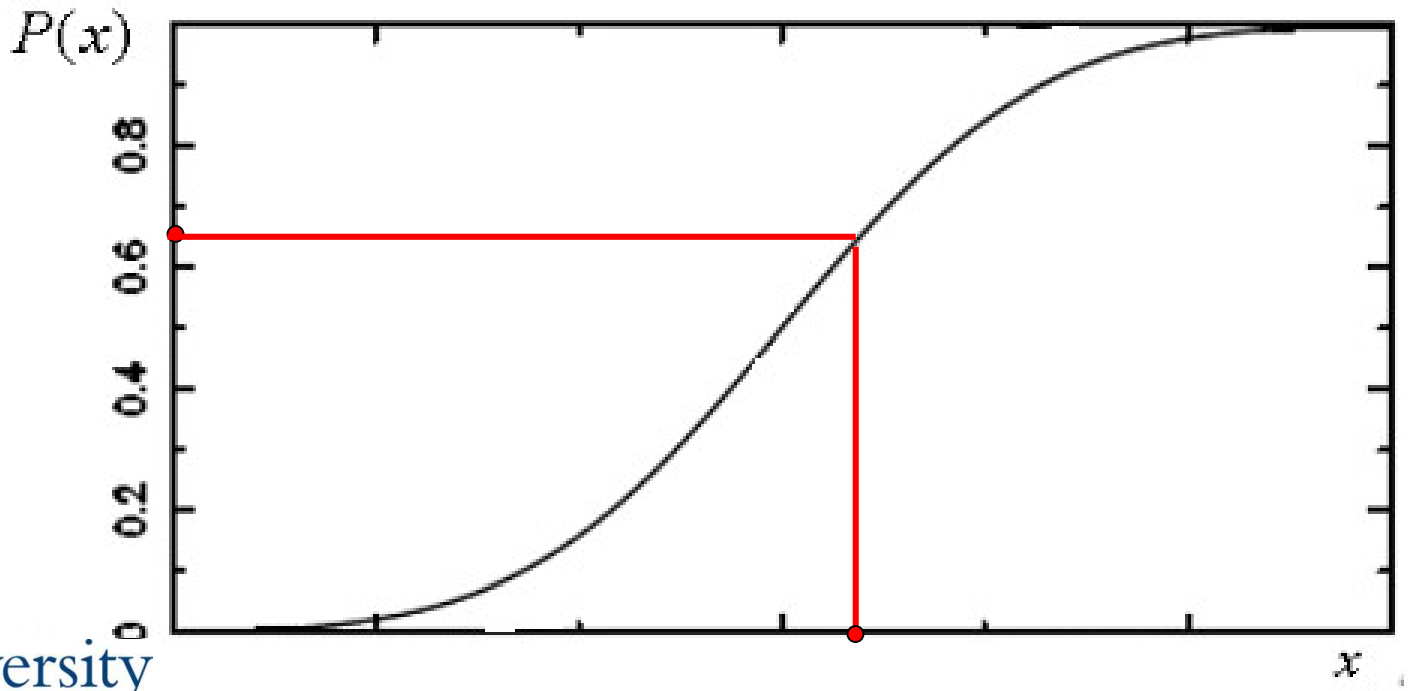
2) Compute x such that $y = P(x)$ i.e. $x = P^{-1}(y)$



1) Sample a random variable $y \sim U[0,1]$

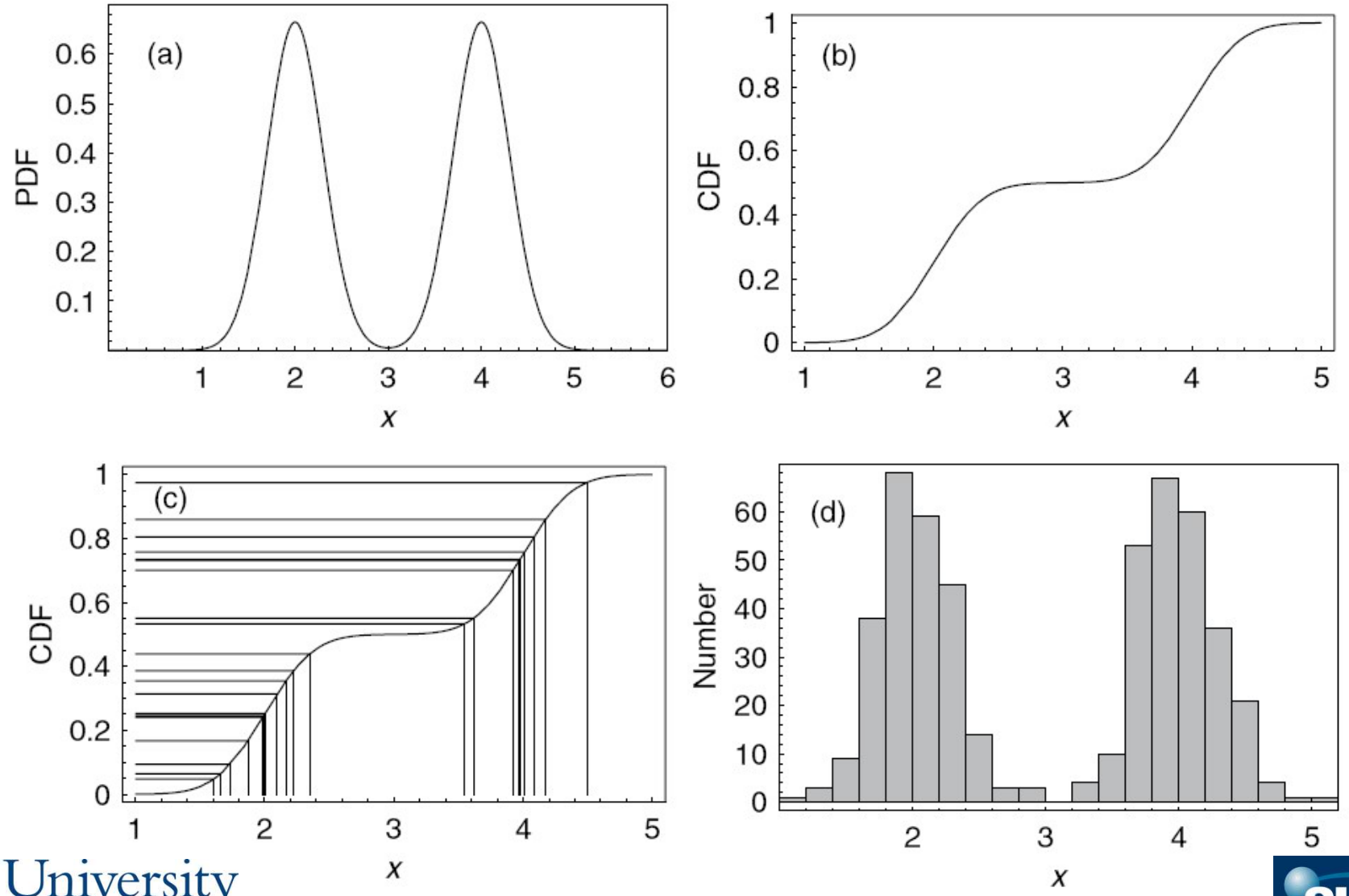
2) Compute x such that $y = P(x)$ i.e. $x = P^{-1}(y)$

3) Then $x \sim p(x)$ i.e. x is drawn from the pdf corresponding to the cdf $P(x)$



Example (from Gregory)

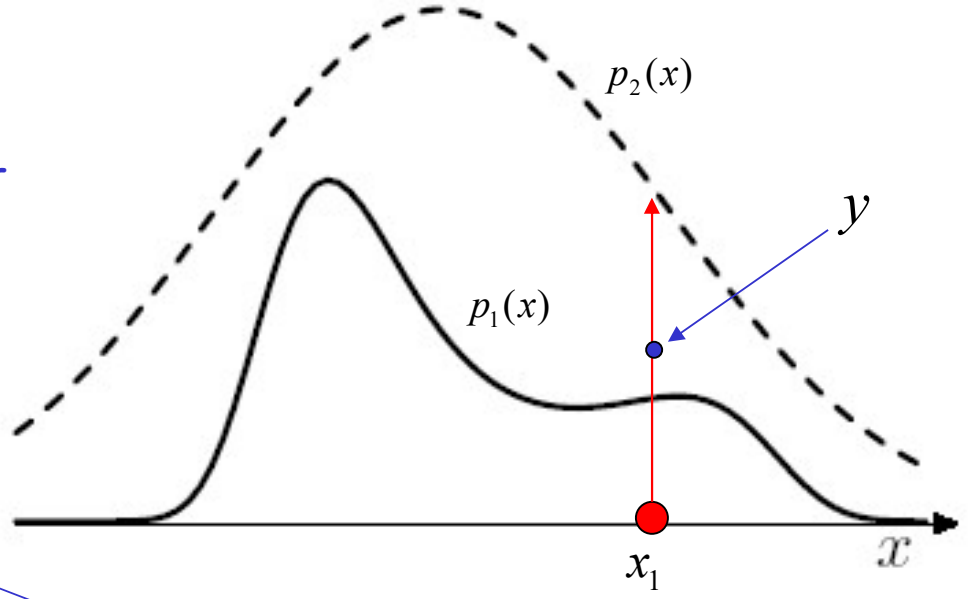
[See also Mathworld applet]



9.4 Rejection sampling

Suppose we want to sample from some pdf $p_1(x)$ and we know that

$$p_1(x) < p_2(x) \quad \forall x$$



1) Sample x_1 from $p_2(x)$

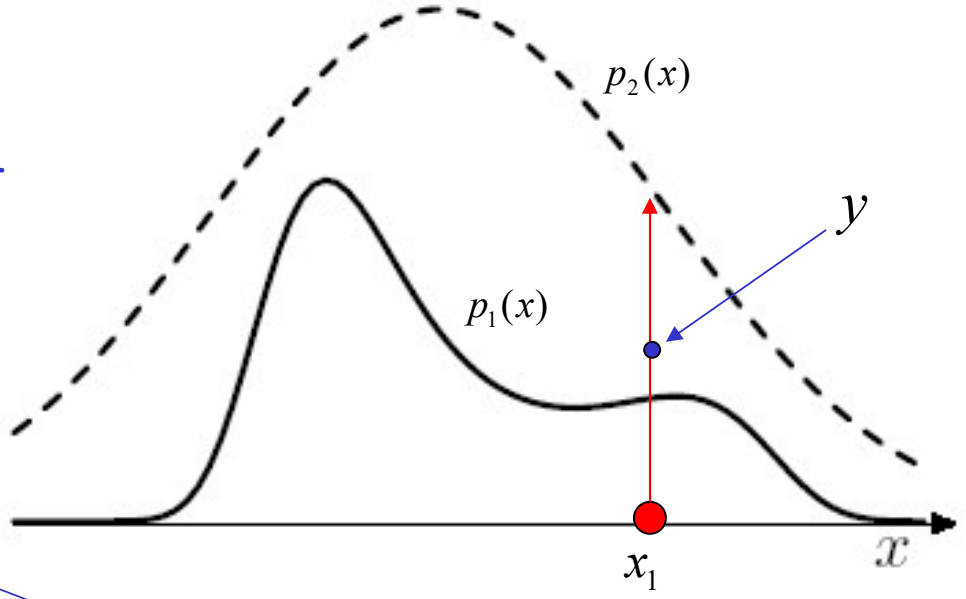
2) Sample $y \sim U[0, p_2(x_1)]$

(Suppose we have an 'easy' way to do this)

9.4 Rejection sampling

Suppose we want to sample from some pdf $p_1(x)$ and we know that

$$p_1(x) < p_2(x) \quad \forall x$$



1) Sample x_1 from $p_2(x)$

2) Sample $y \sim U[0, p_2(x_1)]$

(Suppose we have an 'easy' way to do this)

3) If $y < p_1(x)$ **ACCEPT**

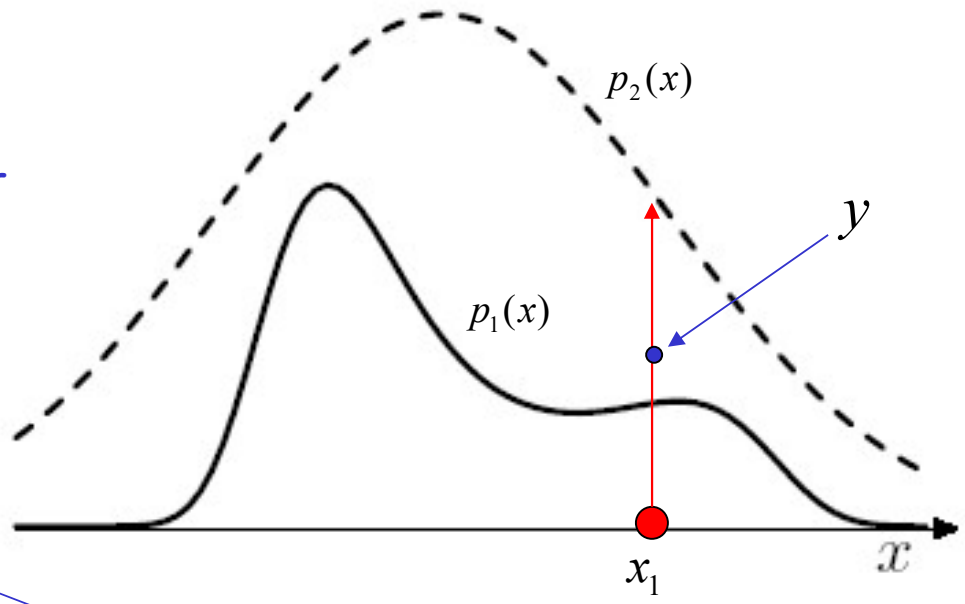
otherwise **REJECT**

9.4 Rejection sampling

(following Mackay)

Suppose we want to sample from some pdf $p_1(x)$ and we know that

$$p_1(x) < p_2(x) \quad \forall x$$



1) Sample x_1 from $p_2(x)$

2) Sample $y \sim U[0, p_2(x_1)]$

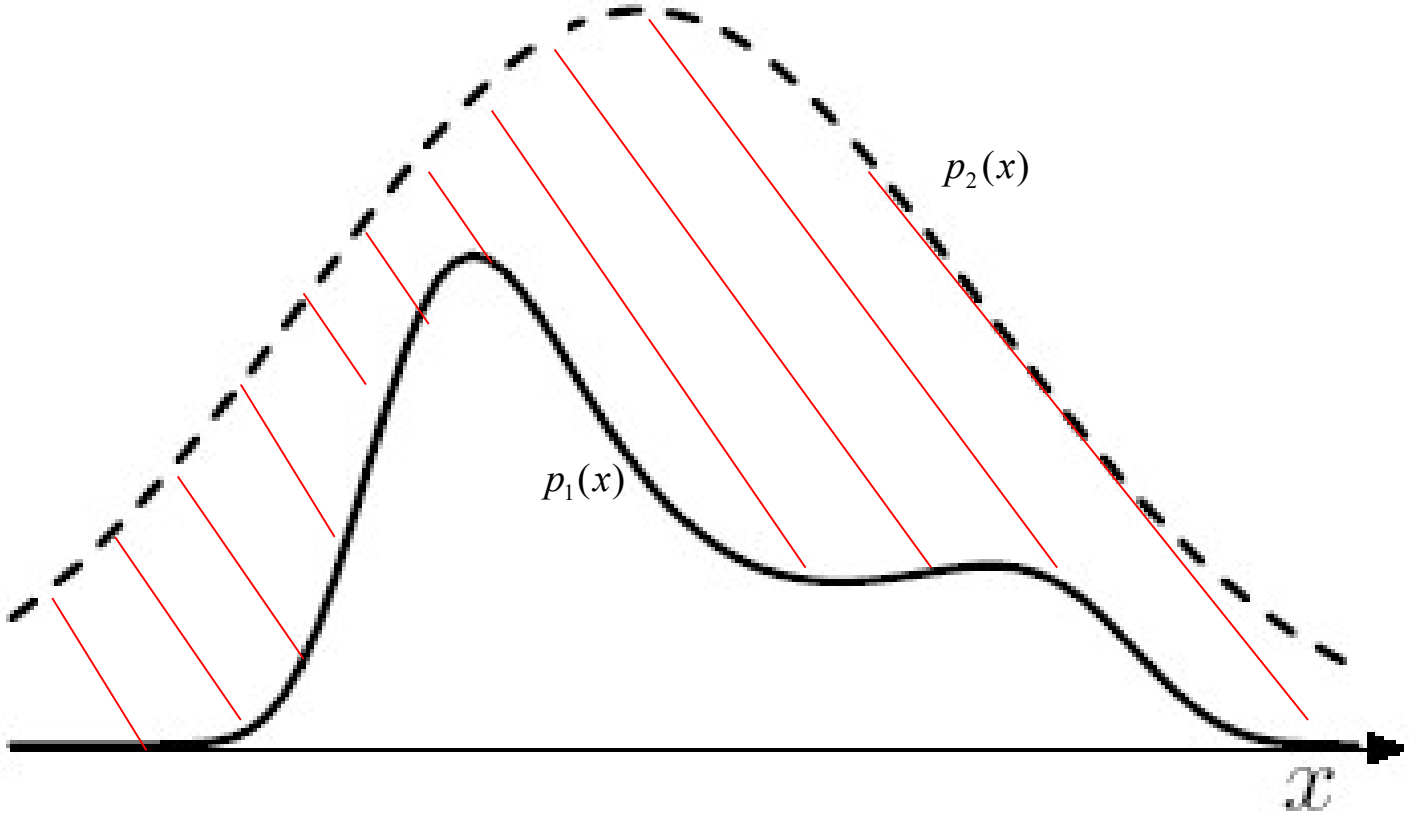
(Suppose we have an 'easy' way to do this)

3) If $y < p_1(x)$ **ACCEPT**

otherwise **REJECT**

Set of accepted values $\{x_i\}$ are a sample from $p_1(x)$

9.4 Rejection sampling



Method can be very slow if the shaded region is too large.

Ideally we want to find a pdf $p_2(x)$ that is: (a) easy to sample from

(b) close to $p_1(x)$

9.5 Genetic Algorithms

1995ApJS..101..309C

THE ASTROPHYSICAL JOURNAL SUPPLEMENT SERIES, 101:309–334, 1995 December

GENETIC ALGORITHMS IN ASTRONOMY AND ASTROPHYSICS

P. CHARBONNEAU
High Altitude Observatory, National Center for Atmospheric Research,[1] P.O. Box 3000, Boulder, CO 80307-3000;
paulchar@hao.ucar.edu
Received 1994 December 30; accepted 1995 June 9

ABSTRACT

This paper aims at demonstrating, through examples, the applicability of *genetic algorithms* to wide classes of problems encountered in astronomy and astrophysics. Genetic algorithms are heuristic search techniques that incorporate, in a computational setting, the biological notion of evolution by means of natural selection. While increasingly in use in the fields of computer science, artificial intelligence, and computed-aided engineering design, genetic algorithms seem to have attracted comparatively little attention in the physical sciences thus far.

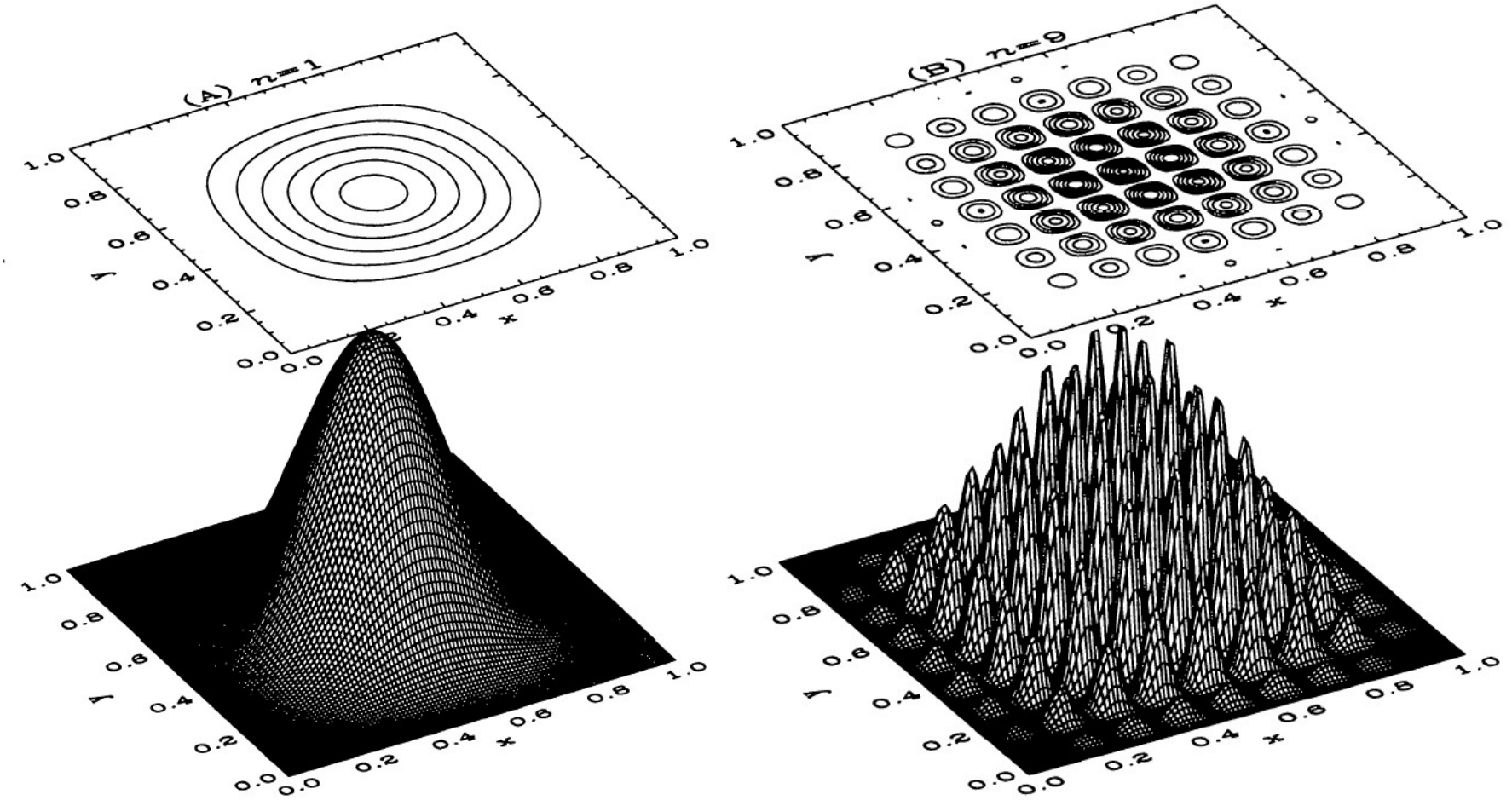
The following three problems are treated: (1) modeling the rotation curve of galaxies, (2) extracting pulsation periods from Doppler velocities measurements in spectral lines of δ Scuti stars, and (3) constructing spherically symmetric wind models for rotating, magnetized solar-type stars. A listing of the genetic algorithm-based general purpose optimization subroutine `PIKAIA`, used to solve these problems, is given in the Appendix.

Subject headings: galaxies: kinematics and dynamics — methods: numerical — stars: mass loss — stars: oscillations

9.5 Genetic Algorithms

$$f(x, y) = [16x(1-x)y(1-y)\sin(n\pi x)\sin(n\pi y)]^2,$$
$$x, y \in [0, 1], \quad n = 1, 2, \cdots$$

(Charbonneau 1995)



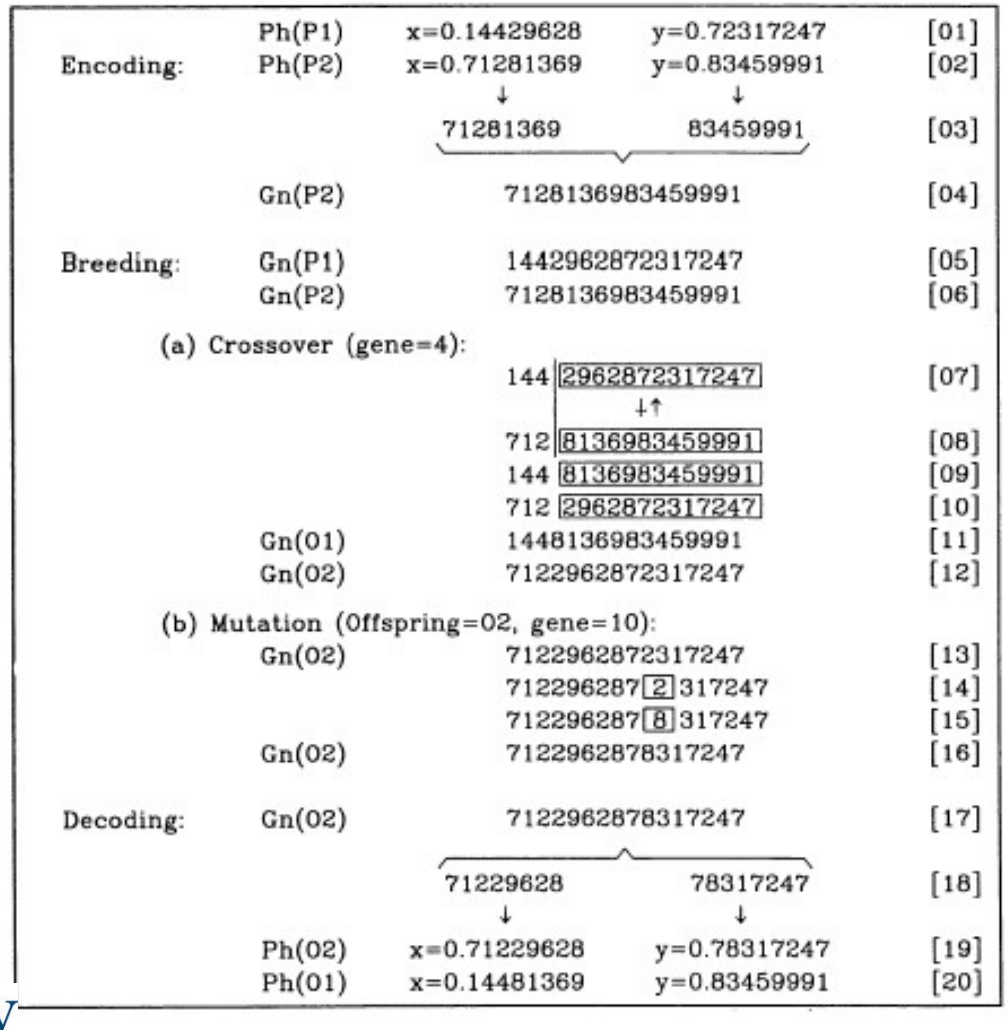
9.5 Genetic Algorithms

(Charbonneau 1995)

1. Construct a random initial population and evaluate the fitness of its members.
2. Construct a new population by breeding selected individuals from the old population.
3. Evaluate the fitness of each member of the new population.
4. Replace the old population by the new population.
5. Test convergence; unless fittest phenotype matches target phenotype within tolerance, `goto` step 2.

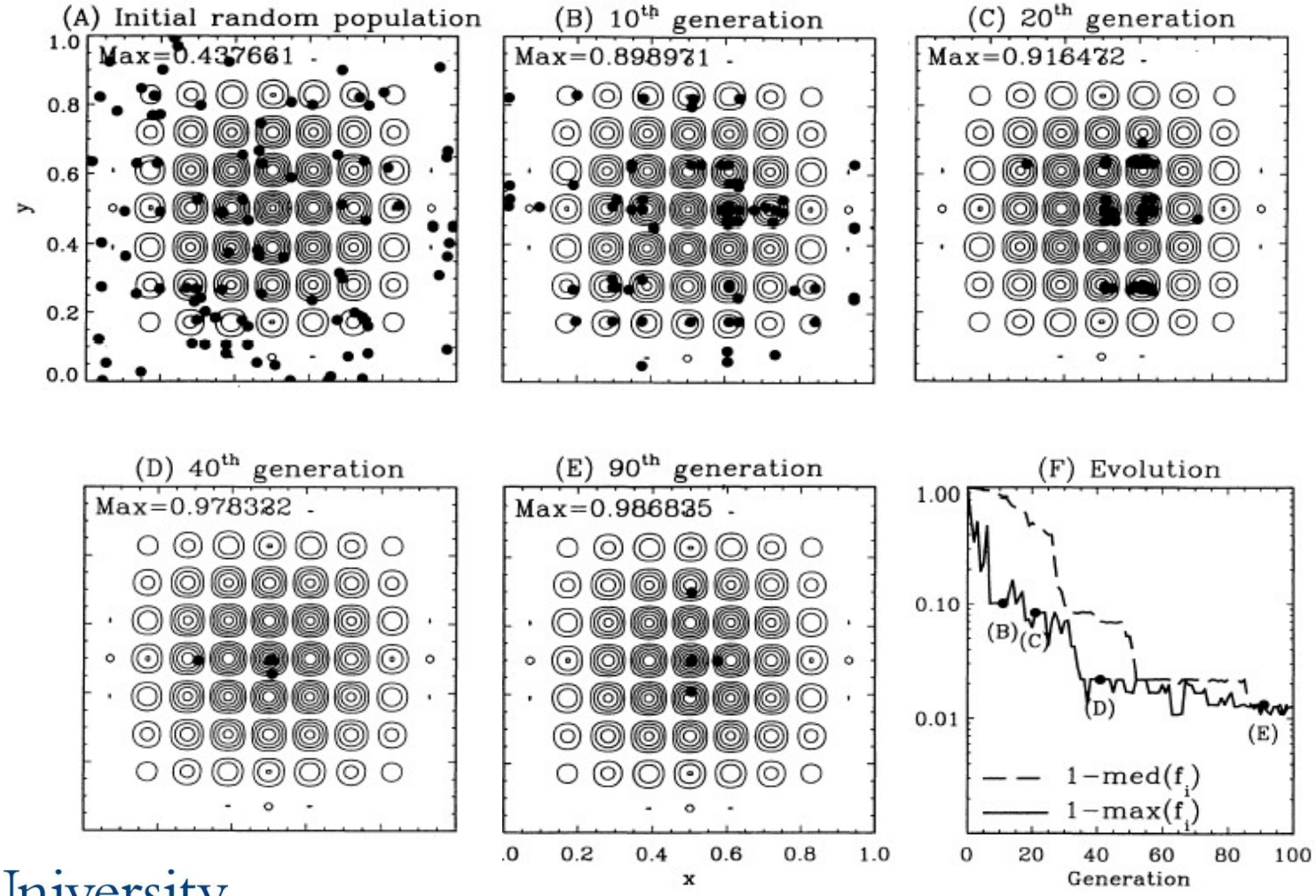
9.5 Genetic Algorithms

see http://www.hao.ucar.edu/Public/models/pikaia/pikaia.html



9.5 Genetic Algorithms

see http://www.hao.ucar.edu/Public/models/pikaia/pikaia.html

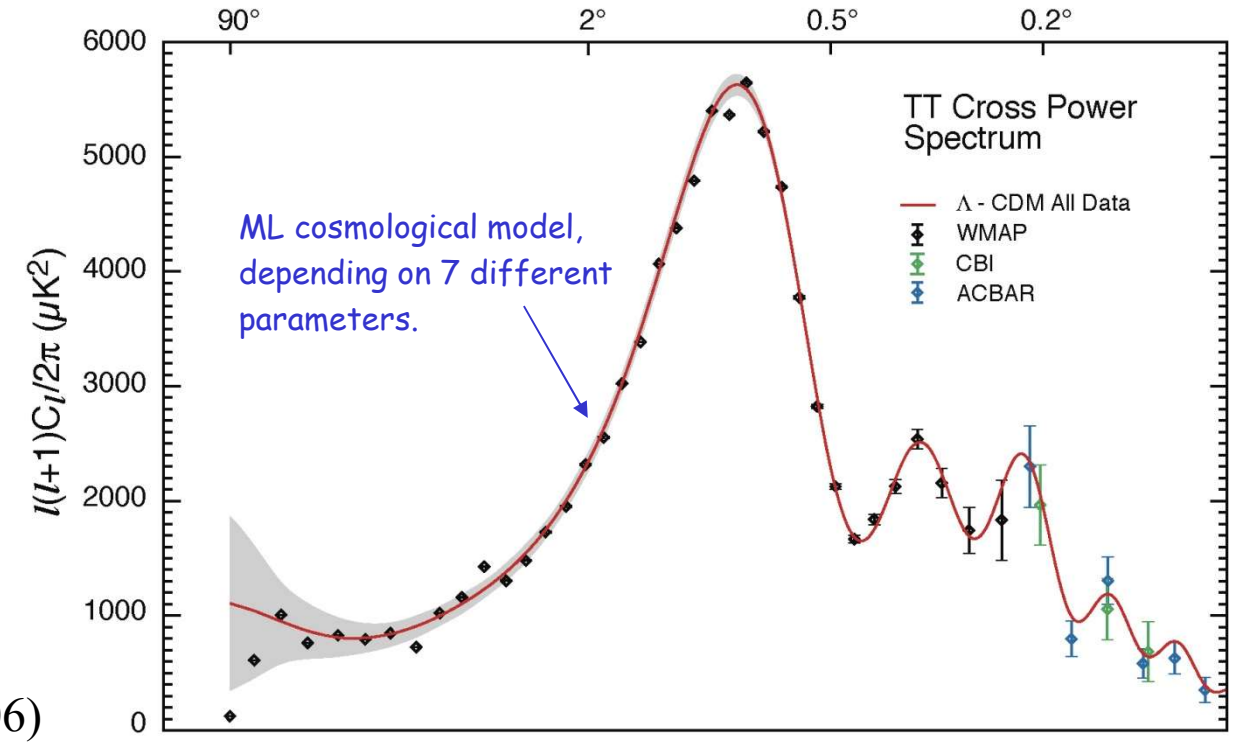


9.6 Markov Chain Monte Carlo

This is a very powerful and (fairly) new method for sampling from pdfs. (These can be complicated and/or of high dimension).

MCMC widely used e.g. in cosmology to determine 'maximum likelihood' model to CMBR data.

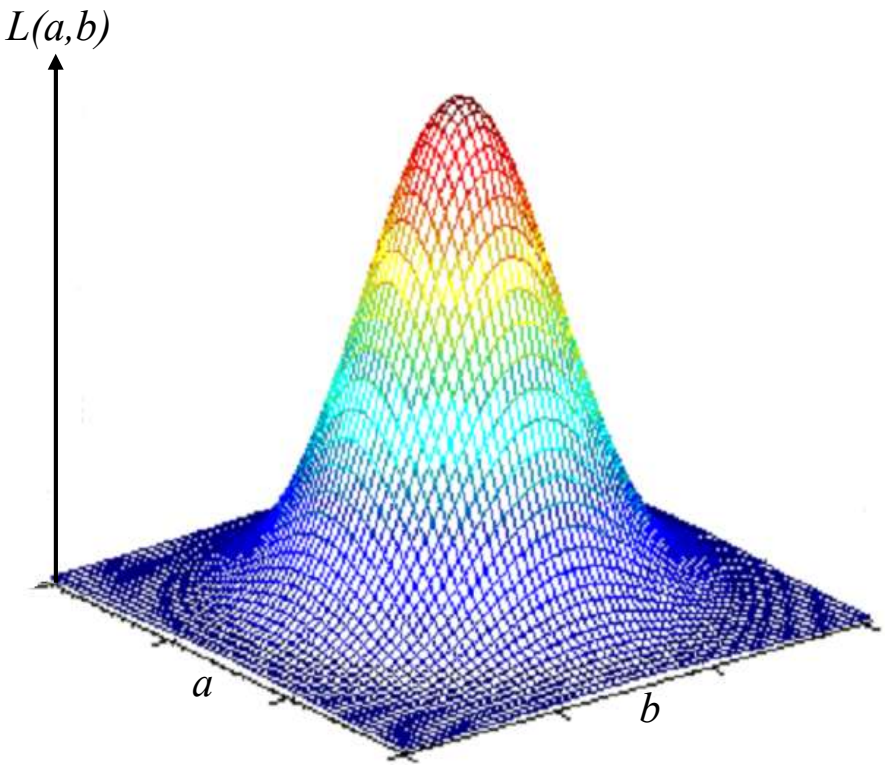
Angular power spectrum of CMBR temperature fluctuations



(Hinshaw et al 2006)

Consider a 2-D example (e.g. bivariate normal distribution);
Likelihood function depends on parameters a and b.

Suppose we are trying to find the maximum of $L(a,b)$



1) Start off at some randomly chosen value (a_1, b_1)

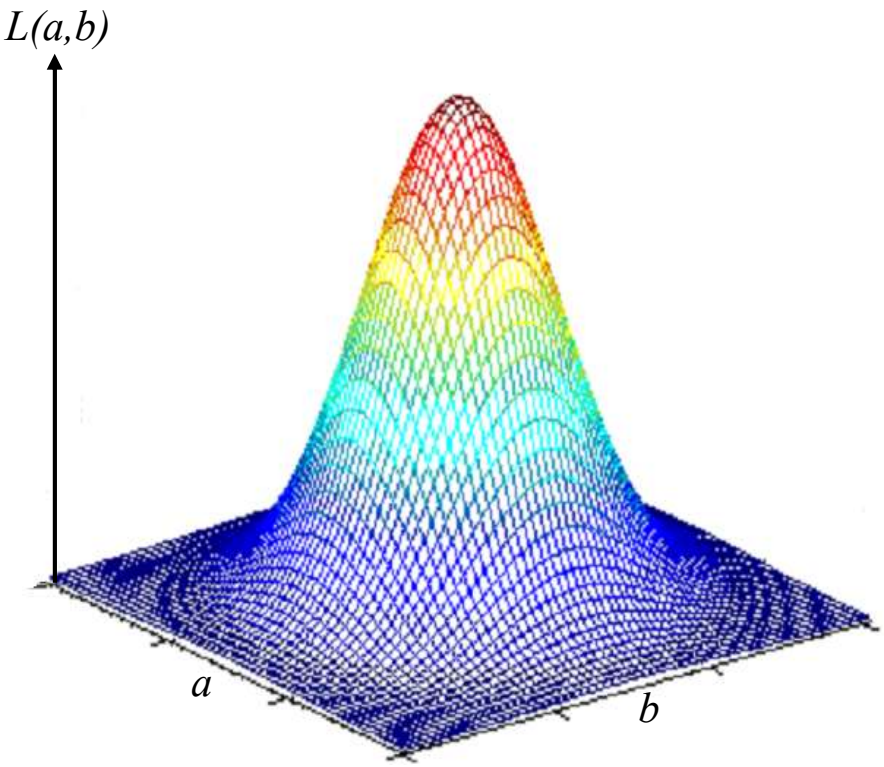
2) Compute $L(a_1, b_1)$ and gradient $\left(\frac{\partial L}{\partial a}, \frac{\partial L}{\partial b}\right)_{(a_1,b_1)}$

3) Move in direction of steepest +ve gradient – i.e. $L(a_1, b_1)$ is increasing fastest

4) Repeat from step 2 until (a_n, b_n) converges on maximum of likelihood

Consider a 2-D example (e.g. bivariate normal distribution);
Likelihood function depends on parameters a and b.

Suppose we are trying to find the maximum of $L(a,b)$



1) Start off at some randomly chosen value (a_1, b_1)

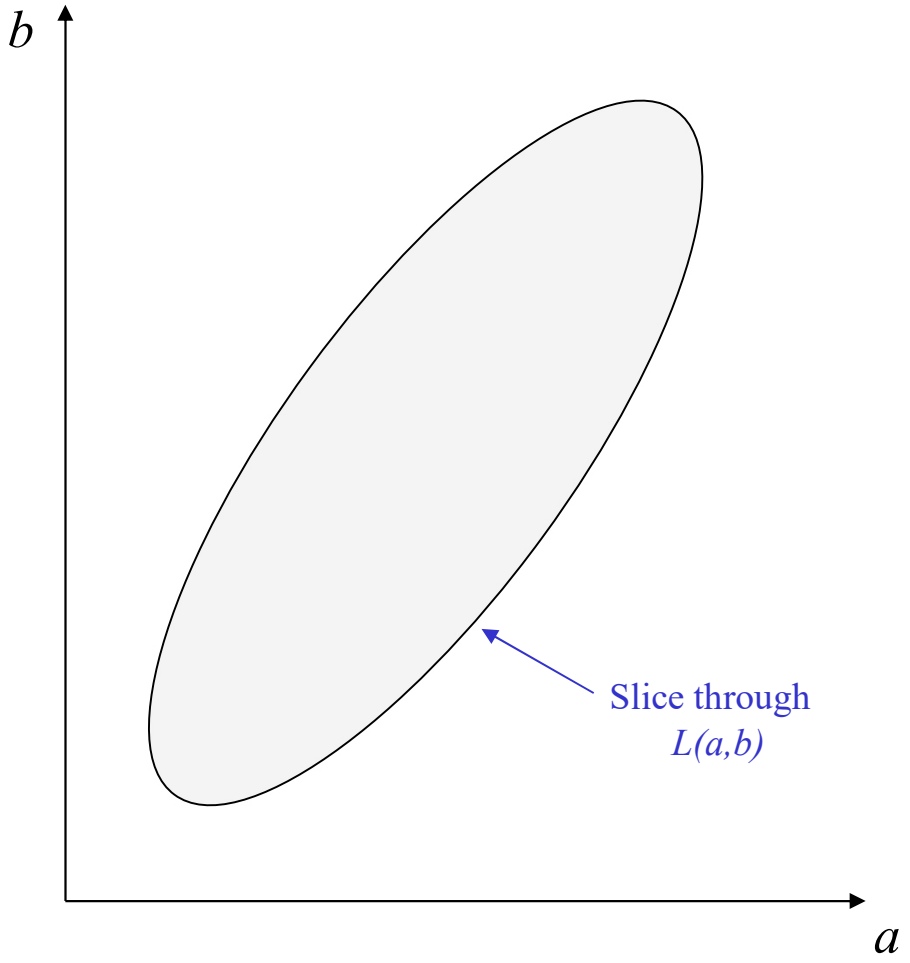
2) Compute $L(a_1, b_1)$ and gradient $\left(\frac{\partial L}{\partial a}, \frac{\partial L}{\partial b}\right)_{(a_1,b_1)}$

3) Move in direction of steepest +ve gradient – i.e. $L(a_1, b_1)$ is increasing fastest

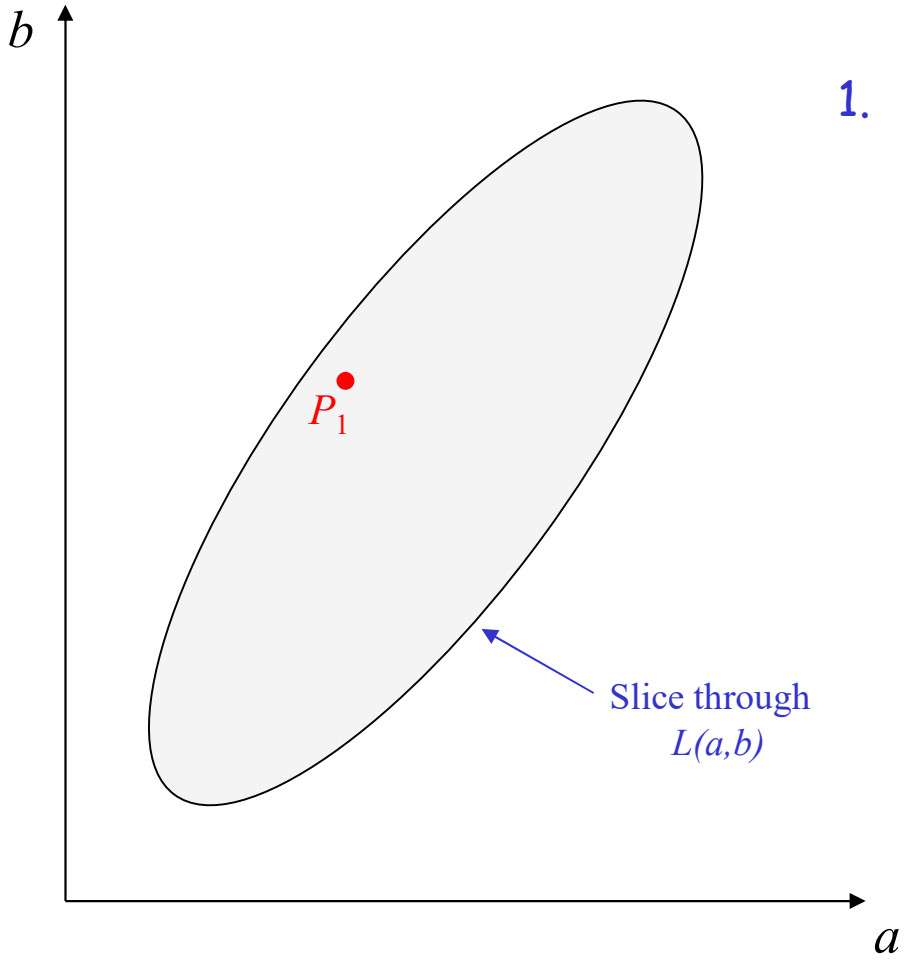
4) Repeat from step 2 until (a_n, b_n) converges on maximum of likelihood

OK for finding maximum, but not for generating a sample from $L(a,b)$ or for determining errors on the the ML parameter estimates.

MCMC provides a simple **Metropolis algorithm** for generating random samples of points from $L(a,b)$

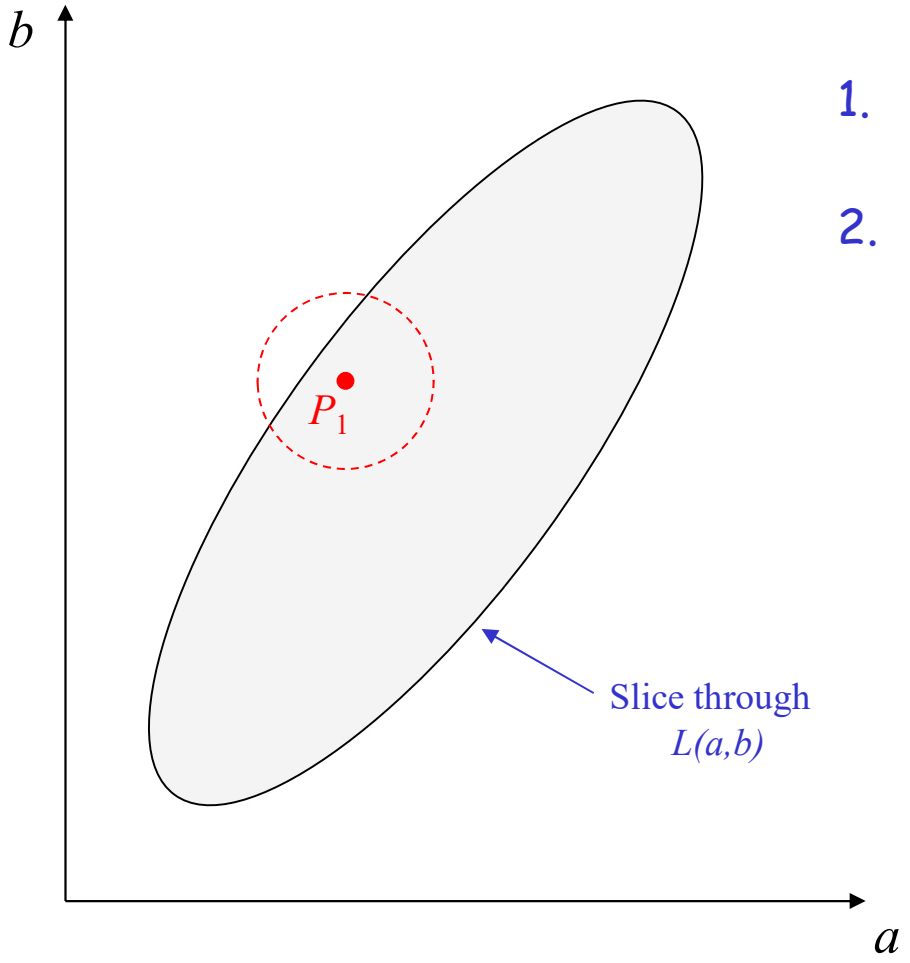


MCMC provides a simple **Metropolis algorithm** for generating random samples of points from $L(a,b)$



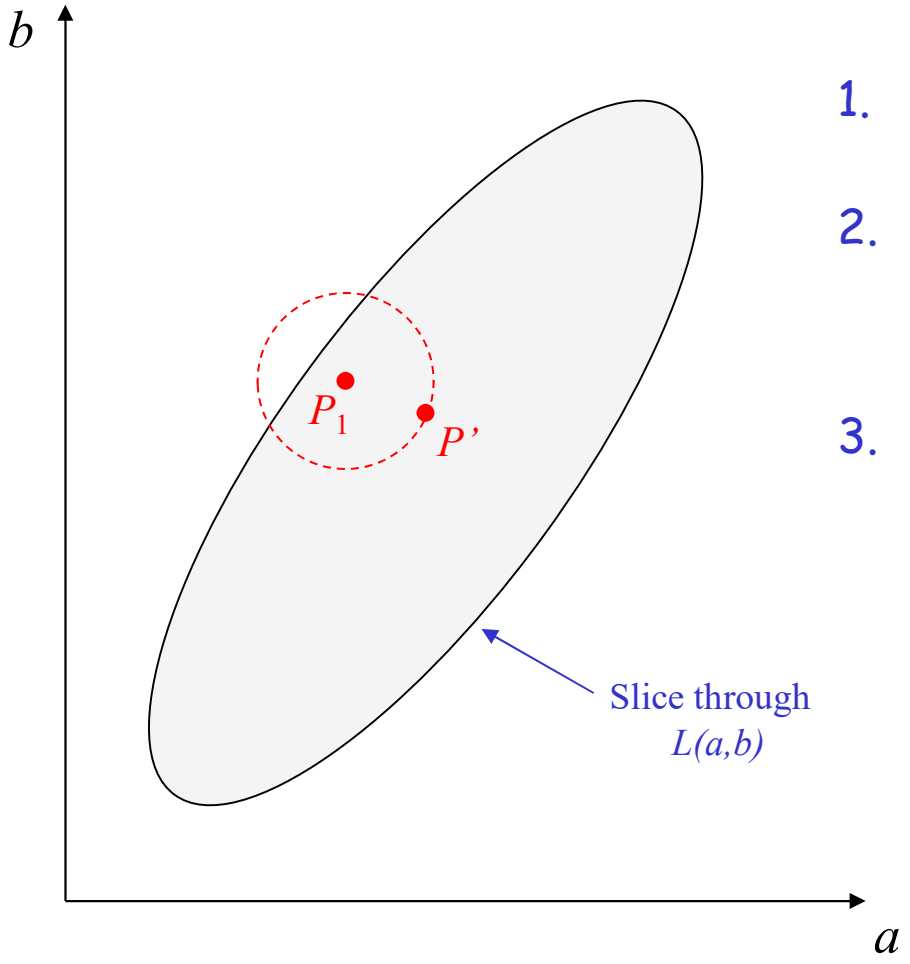
1. Sample random initial point $P_1 = (a_1, b_1)$

MCMC provides a simple **Metropolis algorithm** for generating random samples of points from $L(a,b)$



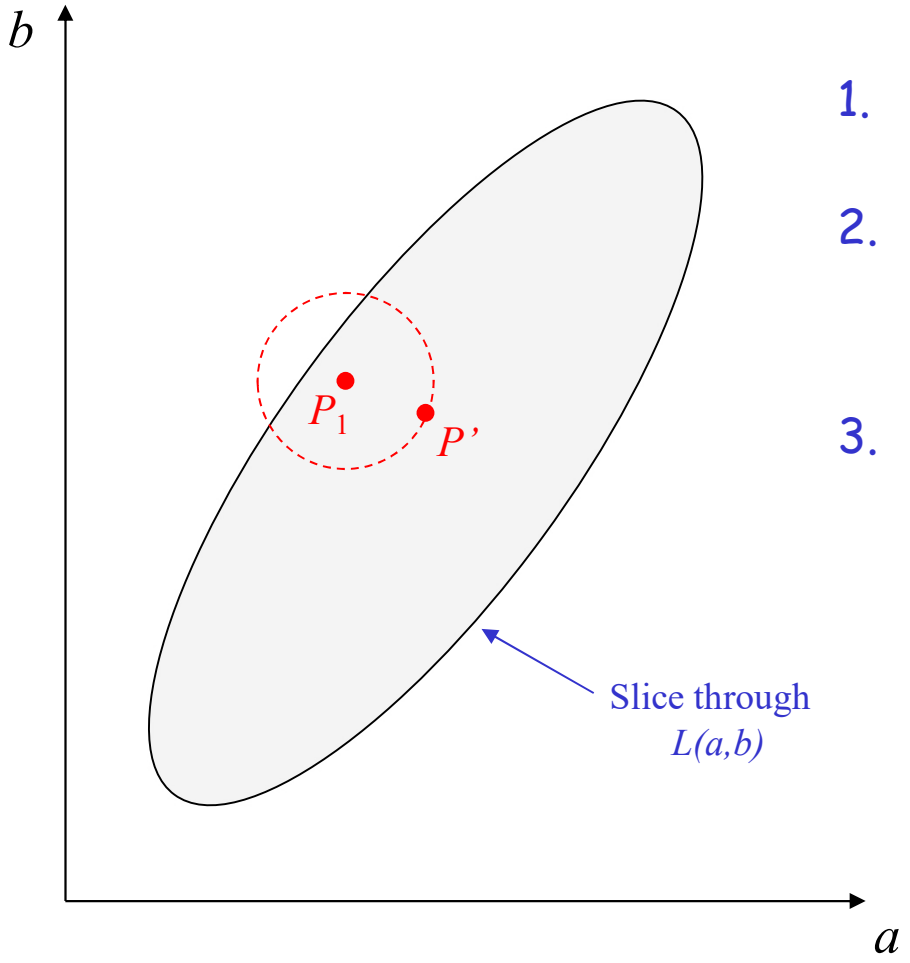
1. Sample random initial point $P_1 = (a_1, b_1)$
2. Centre a new pdf, Q, called the **proposal density**, on P_1

MCMC provides a simple **Metropolis algorithm** for generating random samples of points from $L(a,b)$



1. Sample random initial point $P_1 = (a_1, b_1)$
2. Centre a new pdf, Q, called the **proposal density**, on P_1
3. Sample tentative new point $P' = (a', b')$ from Q

MCMC provides a simple **Metropolis algorithm** for generating random samples of points from $L(a,b)$



1. Sample random initial point $P_1 = (a_1, b_1)$
2. Centre a new pdf, Q, called the **proposal density**, on P_1
3. Sample tentative new point $P' = (a', b')$ from Q
4. Compute $$R = \frac{L(a',b')}{L(a_1,b_1)}$$

5. If $R > 1$ this means P' is **uphill** from P_1.

We **accept** P' as the next point in our chain, i.e. $P_2 = P'$

6. If $R < 1$ this means P' is **downhill** from P_1.

In this case we ***may*** reject P' as our next point.

In fact, we accept P' with probability R.

How do we do this?...

(a) Generate a random number $x \sim \mathrm{U}[0,1]$

(b) If $x < R$ then accept P' and set $P_2 = P'$

(c) If $x > R$ then reject P' and set $P_2 = P_1$

5. If $R > 1$ this means P' is **uphill** from P_1.

We **accept** P' as the next point in our chain, i.e. $P_2 = P'$

6. If $R < 1$ this means P' is **downhill** from P_1.

In this case we ***may*** reject P' as our next point.

In fact, we accept P' with probability R.

How do we do this?...

(a) Generate a random number $x \sim \mathrm{U}[0,1]$

(b) If $x < R$ then accept P' and set $P_2 = P'$

(c) If $x > R$ then reject P' and set $P_2 = P_1$

Acceptance probability depends only on the previous point - ***Markov Chain***

So the Metropolis Algorithm generally (but not always) moves uphill, towards the peak of the Likelihood Function.

Remarkable facts

- Sequence of points $\{ P_1, P_2, P_3, P_4, P_5, \ldots \}$ represents a sample from the LF $L(a,b)$ (see notes on website)
- Sequence for each coordinate, e.g. $\{ a_1, a_2, a_3, a_4, a_5, \ldots \}$ samples the marginalised likelihood of a
- We can make a histogram of $\{ a_1, a_2, a_3, a_4, a_5, \ldots, a_n \}$ and use it to compute the mean and variance of a (i.e. to attach an error bar to a)

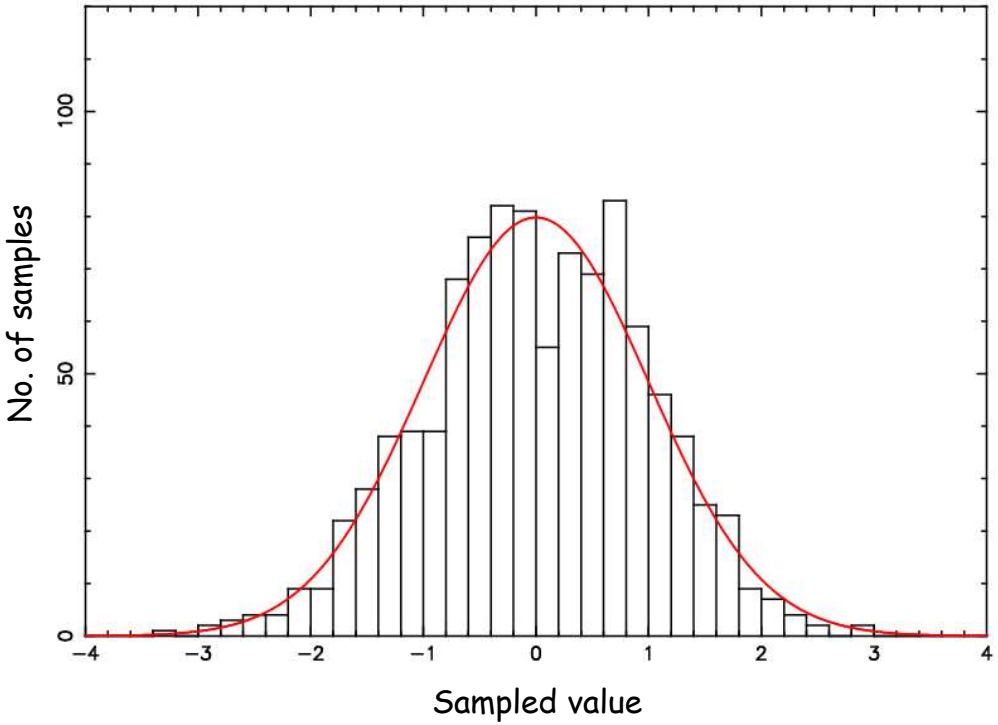
Why is this so useful?…

Suppose our LF was a 1-D Gaussian. We could estimate the mean and variance quite well from a histogram of e.g. 1000 samples.

But what if our problem is, e.g. 7 dimensional?

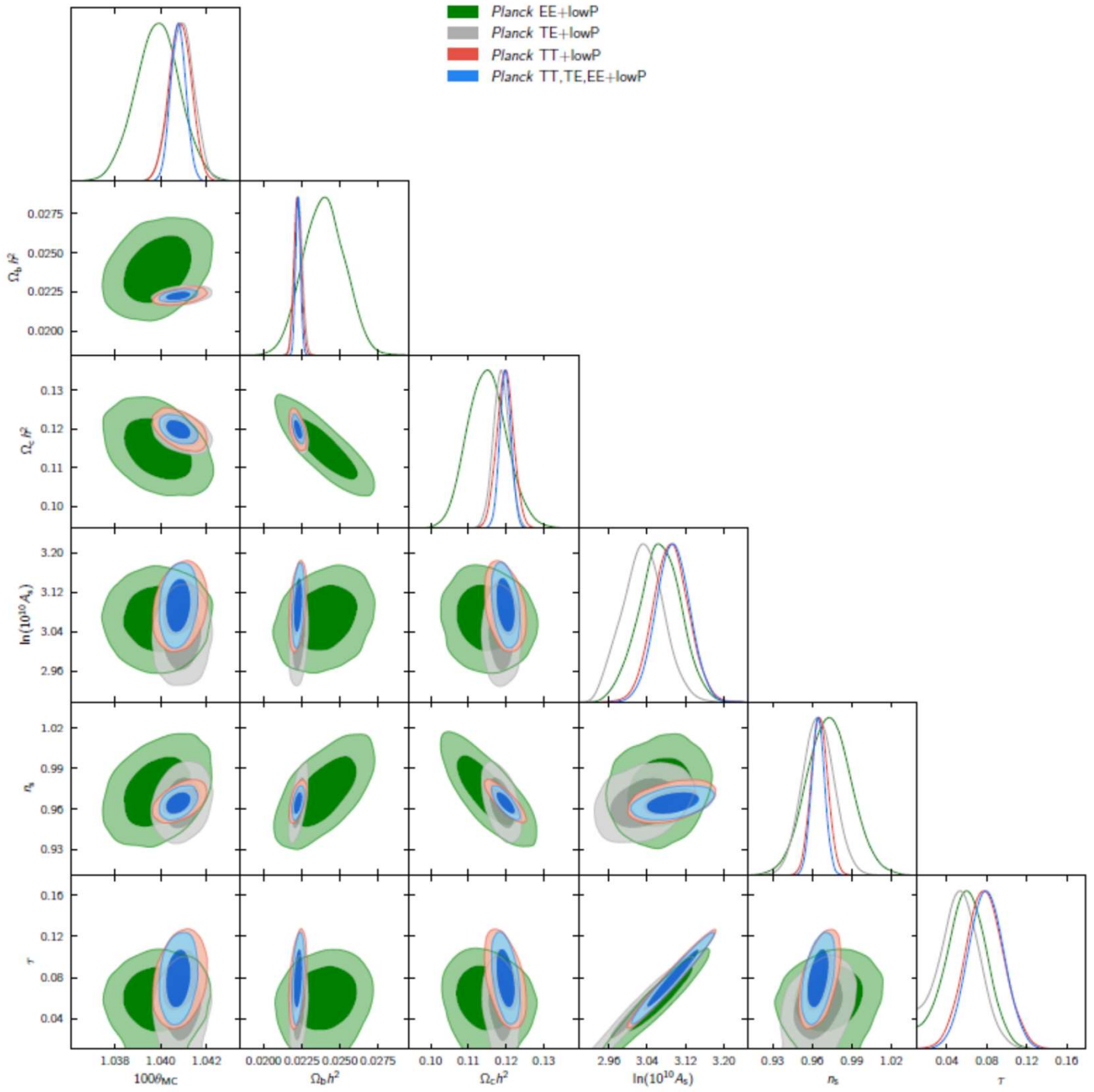
'Exhaustive' sampling could require $(1000)^7$ samples!

MCMC provides a short-cut.



To compute a new point in our Markov Chain we need to compute the LF. But the computational cost does **not** grow so dramatically as we increase the number of dimensions of our problem.

This lets us tackle problems that would be impossible by 'normal' sampling.


Planck EE+lowP
Planck TE+lowP
Planck TT+lowP
Planck TT,TE,EE+lowP
$100\theta_{MC}$
$\Omega_b h^2$
$\Omega_c h^2$
$\ln(10^{10}A_s)$
n_s
τ

Question 19: When applying the Metropolis algorithm, if the width of the proposal density is very small

A the Markov Chain will move around the parameter space very slowly

B the Markov Chain will converge very quickly to the true pdf

C the acceptance rate of proposed steps in the Markov Chain will be very small

D most steps in the Markov Chain will explore regions of very low probability

A number of factors can improve the performance of the Metropolis algorithm, including:

- using parameters in the likelihood function which are (close to) independent (i.e. their Fisher matrix is approx. diagonal).

- adopting a judicious choice of proposal density, well matched to the shape of the likelihood function. (See Mathworld example here).

- using a **simulated annealing** approach – i.e. sampling from a modified posterior likelihood function of the form

$$p_T(\theta \mid D, I) = \exp\left\{\frac{\ln\left[p(\theta \mid D, I)\right]}{T}\right\}$$

for large T the modified likelihood is a flatter version of the true likelihood

Temperature parameter T starts out large, so that the acceptance rate for 'downhill' steps is high – search is essentially random.

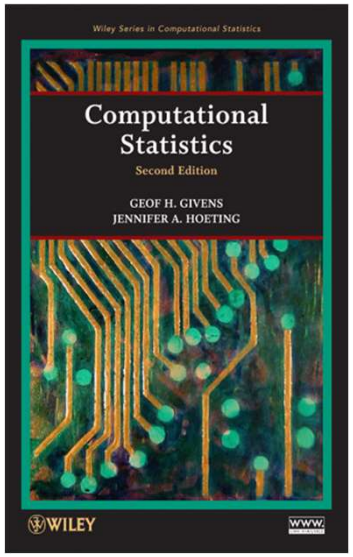
(This helps to avoid getting stuck in local maxima)

T is gradually reduced as the chain evolves, so that 'downhill' steps become increasingly disfavoured.

In some versions, the evolution of T is carried out automatically – known as adaptive simulated annealing.

See, for example, Numerical Recipes Section 10.9, or Gregory Chapter 11, for more details.

Excellent advanced textbook



http://www.stat.colostate.edu/computationalstatistics/

A related idea is **parallel tempering** (see e.g. Gregory, Chap 12)

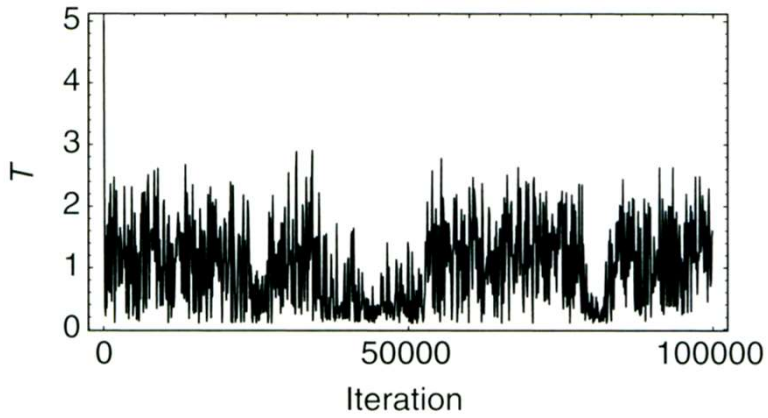
Series of MCMC chains, with different $\beta = 1/T,$ set off in parallel, with a certain probability of swapping parameter states between chains.

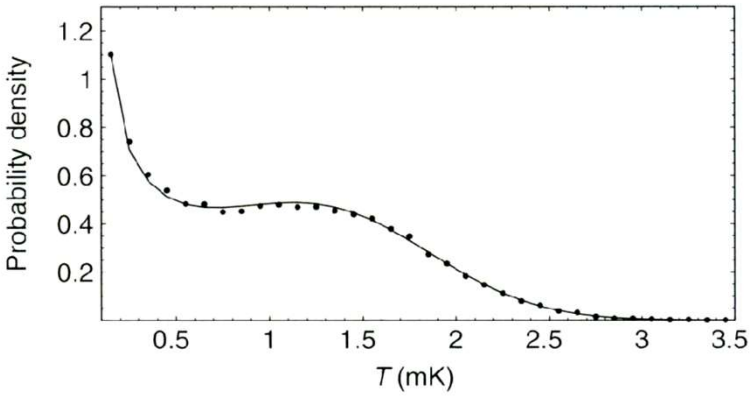
High temperature chains are effective at mapping out the global structure of the likelihood surface.

Low temperature chains are effective at mapping out the shape of local likelihood maxima.

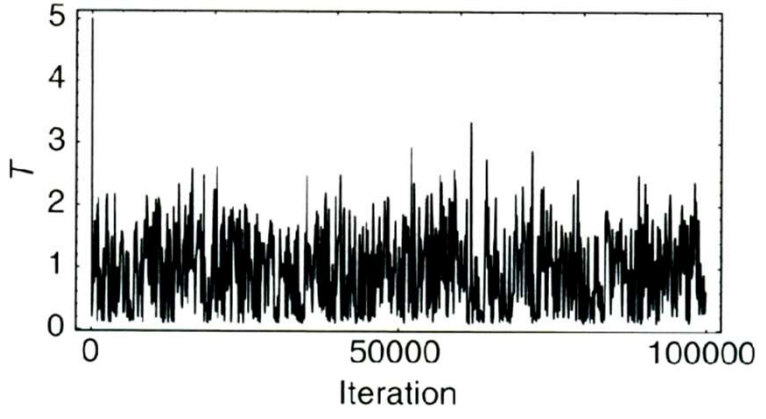
Example: spectral line fitting, from earlier.

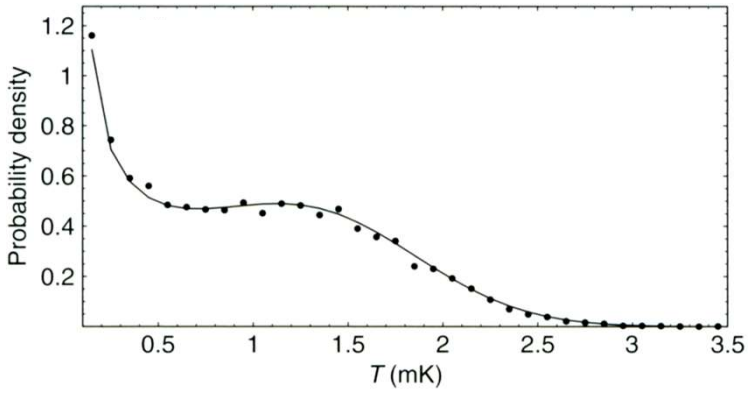
Conventional MCMC





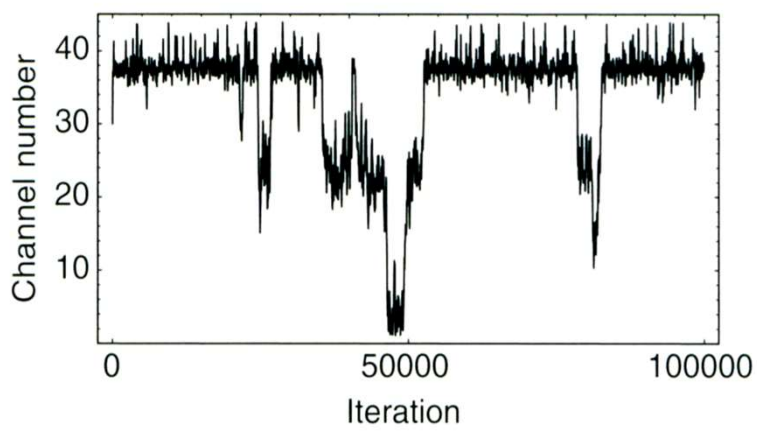
MCMC with parallel tempering

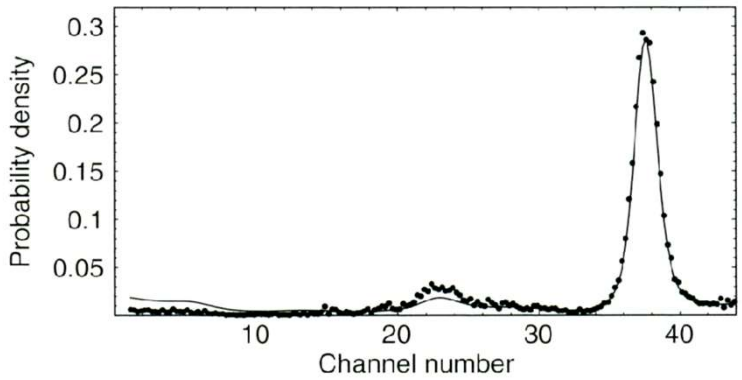




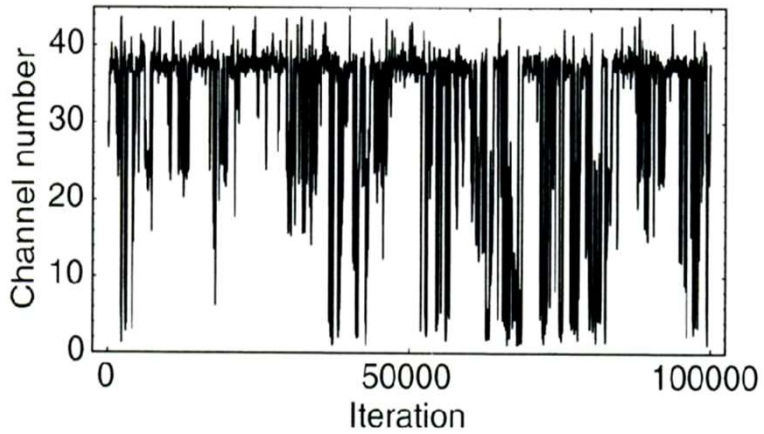
Example: spectral line fitting, from earlier.

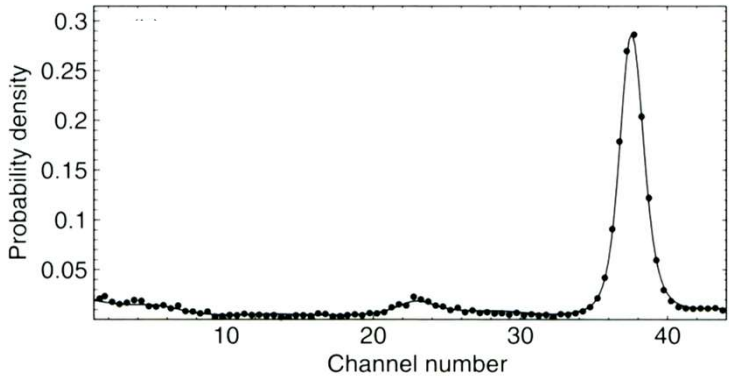
Conventional MCMC





MCMC with parallel tempering





Approximating the Evidence

$$\text{Evidence} = \int \mathrm{p}(\text{data}\,|\,\theta, M)\, p(\theta\,|\,M)\, d\theta$$

Average likelihood, weighted by prior

- Calculating the evidence can be computationally very costly (e.g. CMBR C_ℓ spectrum in cosmology)
- How to proceed?...

1. Information criteria (see e.g. Liddle 2004, 2007)

1. Laplace and Savage-Dickey approximations (see e.g. Trotta 2005)

3. Nested sampling (Skilling 2004, 2006; http://www.inference.phy.cam.ac.uk/bayesys/)

Nested Sampling (Skilling 2004, 2006; Mukherjee et al 2005, 2007)

$$\text{Evidence} = \int \text{p}(\text{data}\,|\,\theta, M)\, p(\theta\,|\,M)\, d\theta$$

Key idea:

We can rewrite the Evidence as

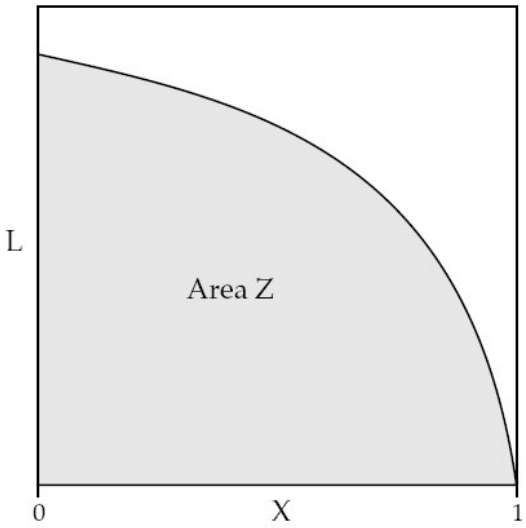
$$\text{Evidence} = \int \text{p}(\text{data}\,|\,\theta, M)\, \text{dX}$$

where X is a 1-D variable known as the ***prior mass*** uniformly distributed on [0,1]

$$\text{Evidence} = Z = \int_0^1 L(X)\, \text{dX}$$

Skilling (2006)

$$\text{Evidence} \quad = \quad Z \quad = \quad \int_0^1 L(X)\,\mathrm{dX}$$



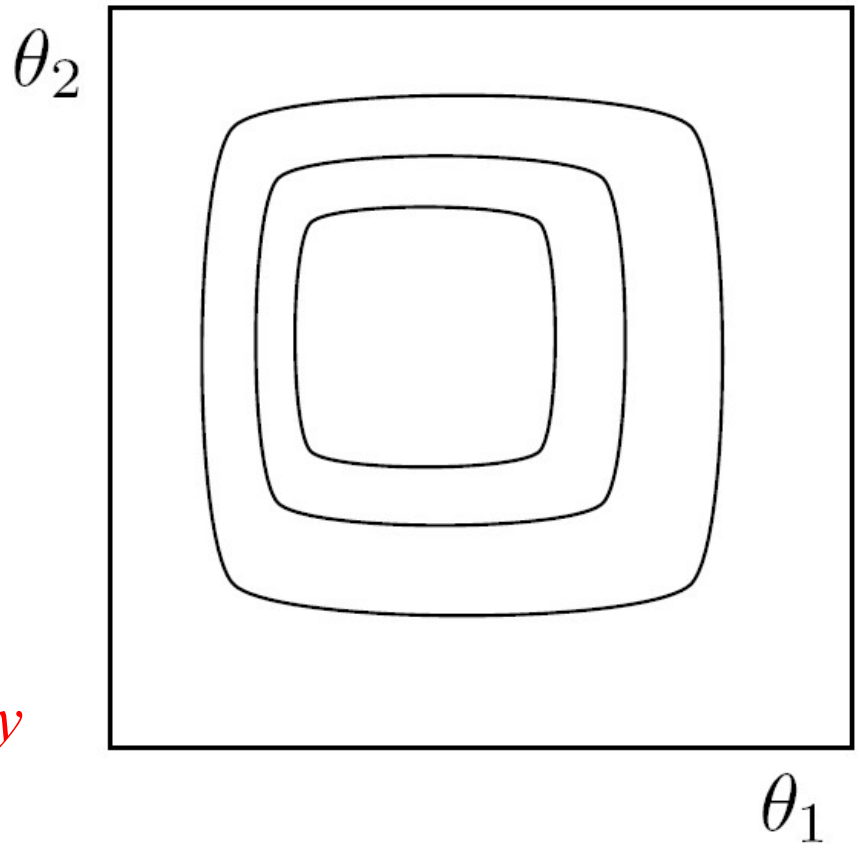
Example: 2-D Likelihood function

$$L \;=\; \begin{array}{|r|r|r|r|} \hline 0 & 8 & 15 & 3 \\ \hline 11 & 24 & 22 & 10 \\ \hline 19 & 30 & 26 & 16 \\ \hline 9 & 23 & 18 & 6 \\ \hline \end{array}$$

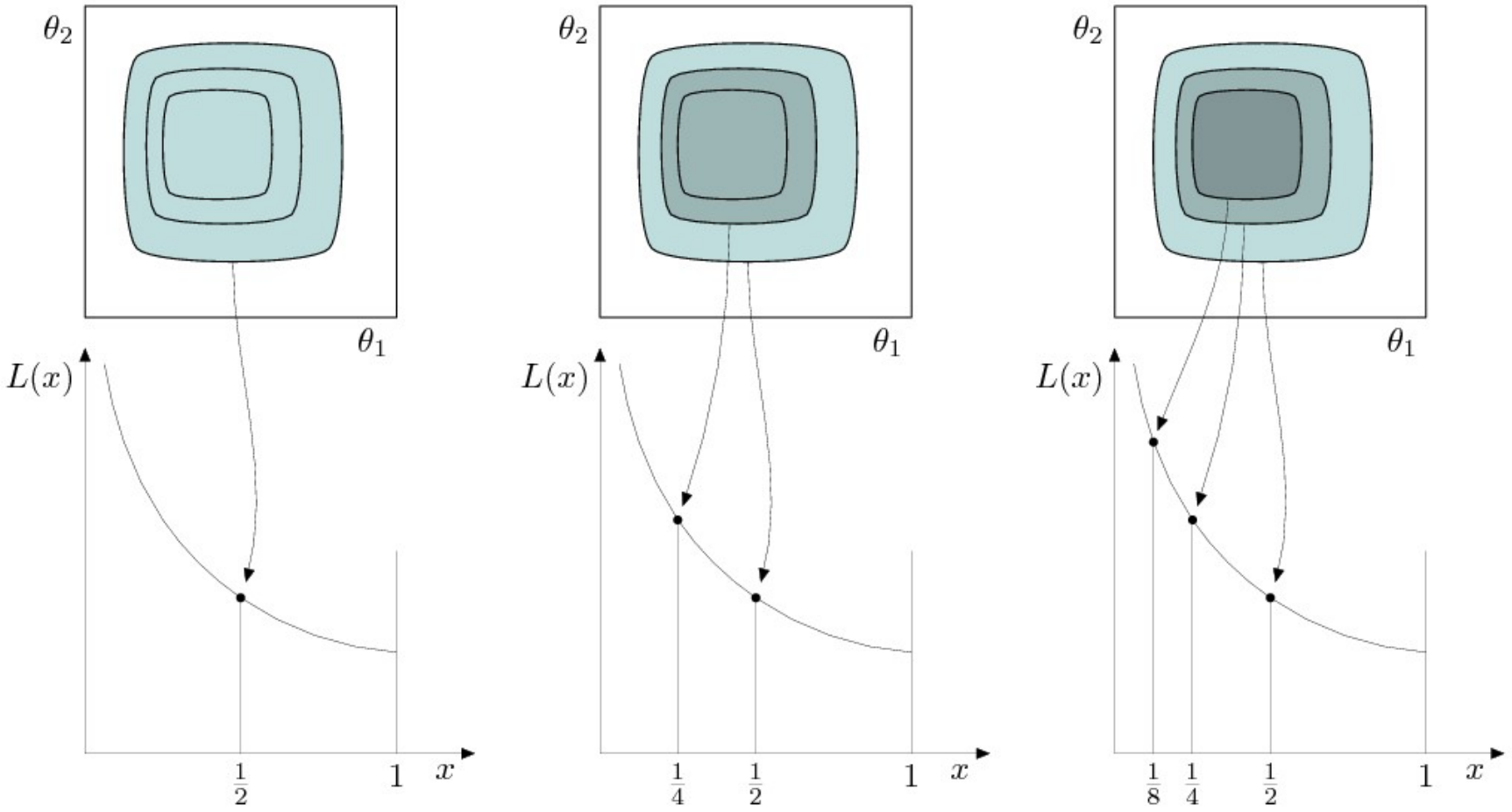
Our plan is to proceed as if we could sort these elements by likelihood, in the above example to $L = (30,26,24,23,22,19,18,16,15,11,10,9,8,6,3,0)$, whence $Z = \frac{30}{16}+\frac{26}{16}+\frac{24}{16}+\frac{23}{16}+\frac{22}{16}+\frac{19}{16}+\frac{18}{16}+\frac{16}{16}+\frac{15}{16}+\frac{11}{16}+\frac{10}{16}+\frac{9}{16}+\frac{8}{16}+\frac{6}{16}+\frac{3}{16}+\frac{0}{16} = 15$, to be evaluated right-to-left into domains of progressively greater likelihood. The likelihood corresponding to (say) $X = \frac{1}{5}$, being one fifth of the way along the sequence so falling into the fourth cell out of sixteen, is $L(X=0.2) = 23$.

Example: 2-D Likelihood function (from Mackay 2005)

- Contours of constant likelihood, L
- Each contour encloses a different fraction, X, of the area of the square
- Each point in the plane has an associated value of L and X



However, mapping systematically the relationship between L *and* X **everywhere** *may be computationally very costly*

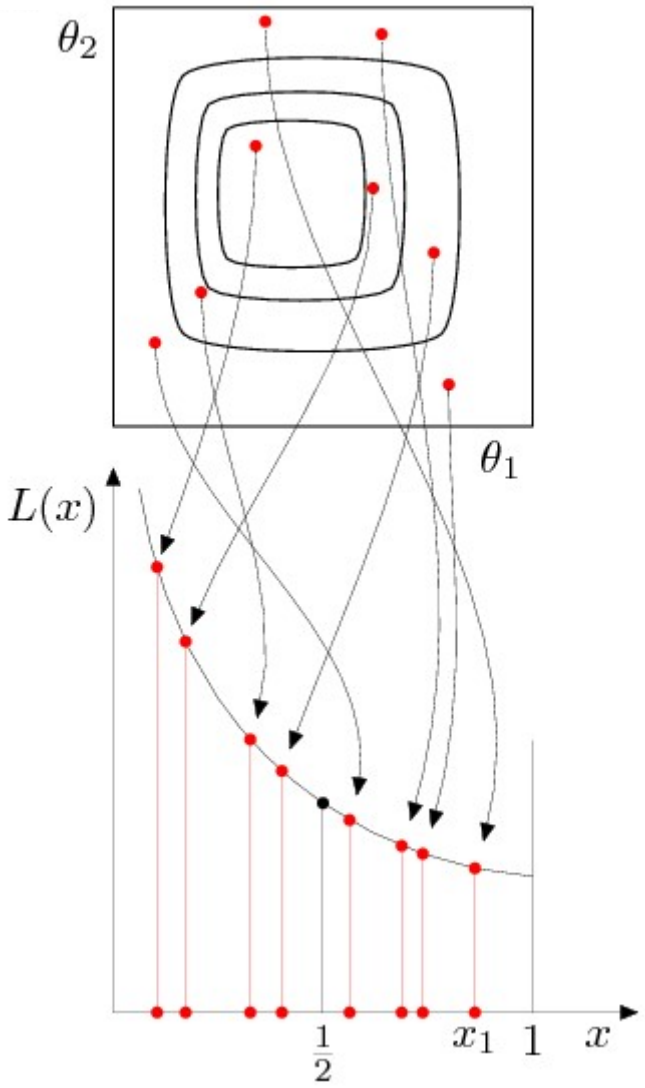


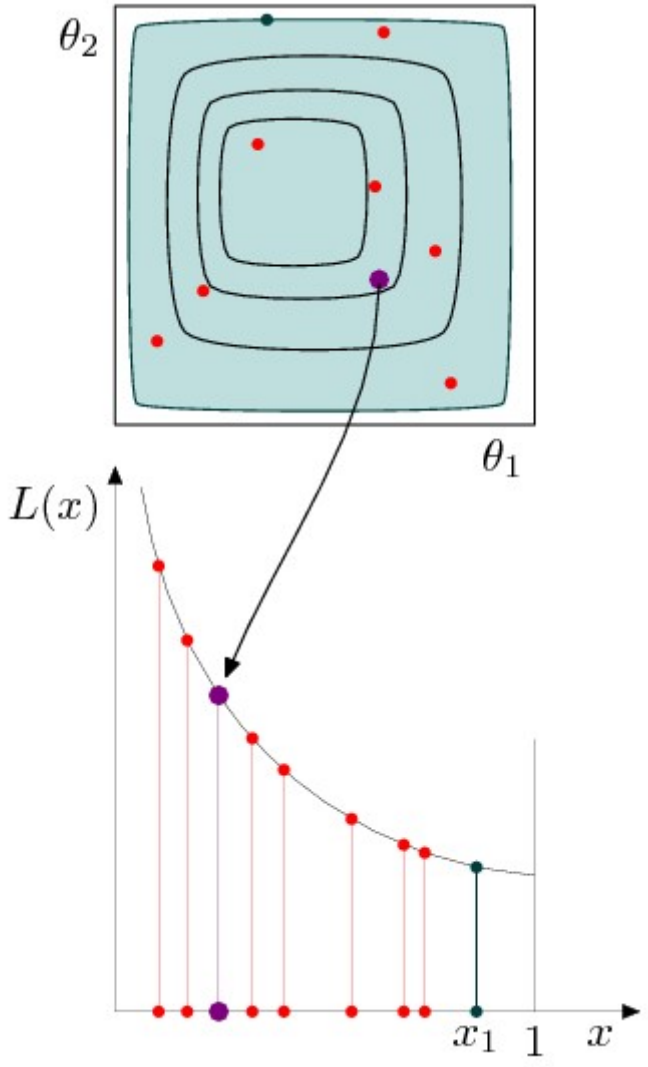
However, mapping systematically the relationship between L *and* X **everywhere** *may be computationally very costly*

Skilling (2006)

Approximation procedure

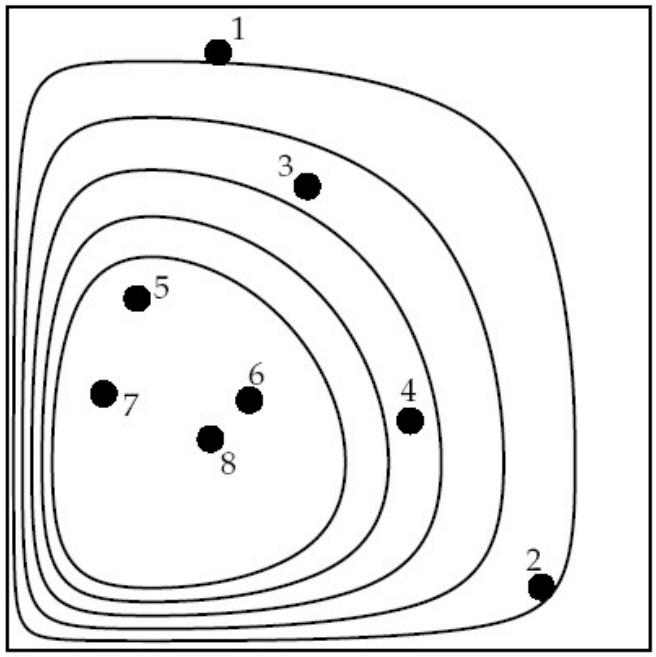
Start with N points $\theta_1, \ldots, \theta_N$ from prior;
initialise $Z = 0$, $X_0 = 1$.
Repeat for $i = 1, 2, \ldots, j$;
record the lowest of the current likelihood values as L_i,
set $X_i = \exp(-i/N)$ (crude) or sample it to get uncertainty,
set $w_i = X_{i-1} - X_i$ (simple) or $(X_{i-1} - X_{i+1})/2$ (trapezoidal),
increment Z by $L_i w_i$,
then replace point of lowest likelihood by new one drawn from within $L(\theta) > L_i$, in proportion to the prior $\pi(\theta)$.
Increment Z by $N^{-1}(L(\theta_1) + \ldots + L(\theta_N))\, X_j$.



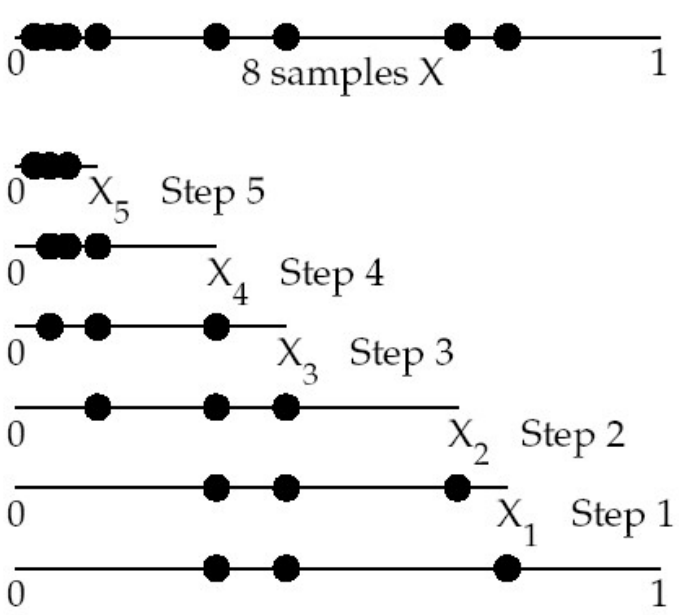


Let x_1 be the largest x-value. The typical value of x_1 is something like $N/(N+1)$ or $e^{-1/N}$. (The former is its arithmetic expected value, the latter its geometric mean.) We introduce a contour associated with this point.

Skilling (2006)



Parameter space



Enclosed prior mass X

NESTED SAMPLING TERMINATION

Termination of the main loop could simply be after a pre-set number of steps, or could be when even the largest current likelihood, taken over the full current box, would not increase the current evidence by more than some small fraction f;

$$\max(L_1, \ldots, L_N) X_j < f Z_j \implies \text{termination.} \qquad (16)$$

Plausibly, the accumulation of Z is then tailing off, so the sum is nearly complete.