STATISTICAL METHODS FOR THE PHYSICAL SCIENCES

Week 7: Monte Carlo methods

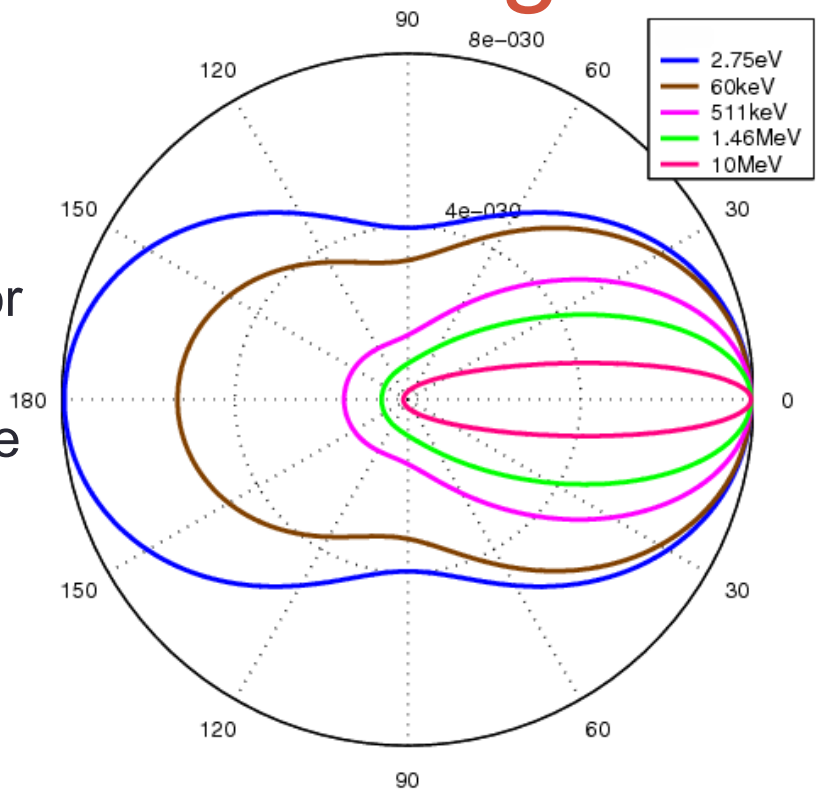
The power of random numbers…

Monte Carlo methods are any computational approach to problem solving which makes use of random numbers to solve problems which are extremely difficult or even impossible to solve analytically or with other numerical methods. There are a wide variety of applications:

- Physical simulations: radiation processes which follow single particles through multiple interactions, to build up the distribution of observed quantities (e.g. photon spectra from an emission process, background events in a detector).
- Population synthesis: observed populations are sampled from underlying populations which have undergone various evolutionary effects. What we observe may not be what we want to measure – how do we go from the underlying physics to our observations?
- Monte Carlo integration: using random numbers to map the volume of an integral.
- Markov Chain Monte Carlo: using a random process to map the pdf of parameters in complex multidimensional parameter spaces.

Monte Carlo Compton scattering

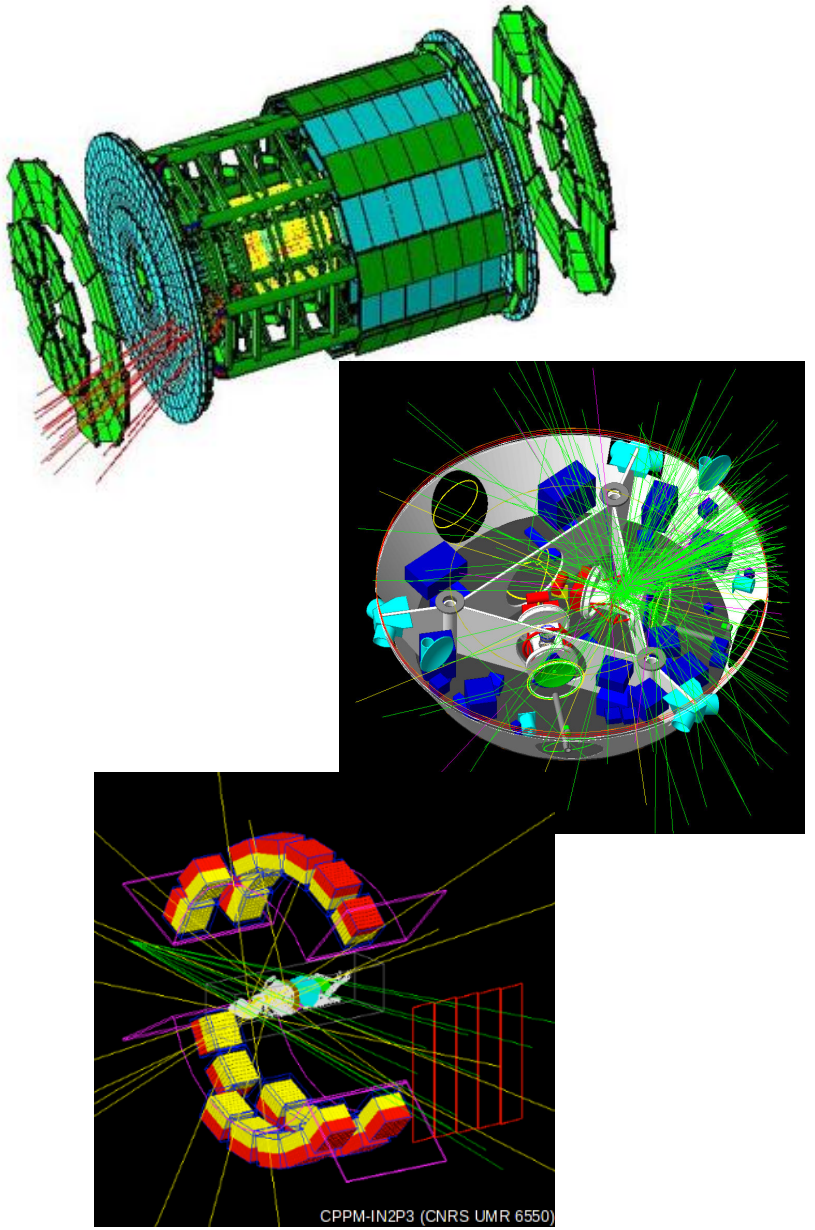
- Incident photon energies drawn from a suitable distribution (e.g. blackbody)
- Location of scattering depends probabilistically on total cross-section for that energy
- Scattering angle and new energy can be determined for the given energy using Klein-Nishina differential cross-section with accept-reject, with uniform distribution U(0,2π) for azimuthal scattering angle
- Can record the effects of multiple scatterings
- Additionally, could sample electron energy distribution and directions (for inverse Compton)



Klein-Nishina differential cross-section for scattering of different photon energies

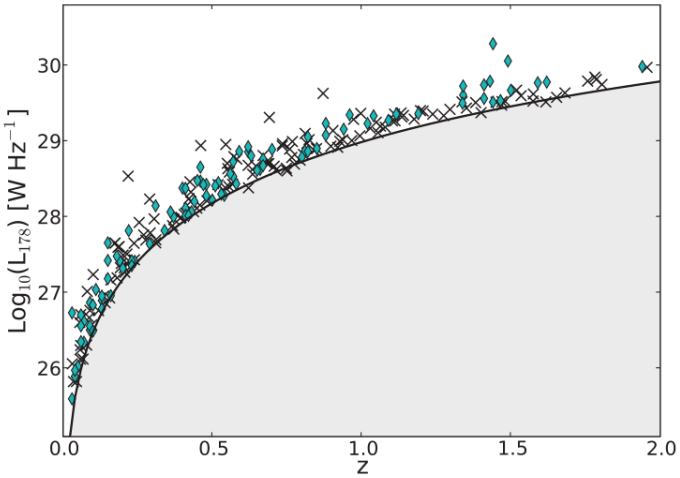
MC case studies: physical simulations: detectors

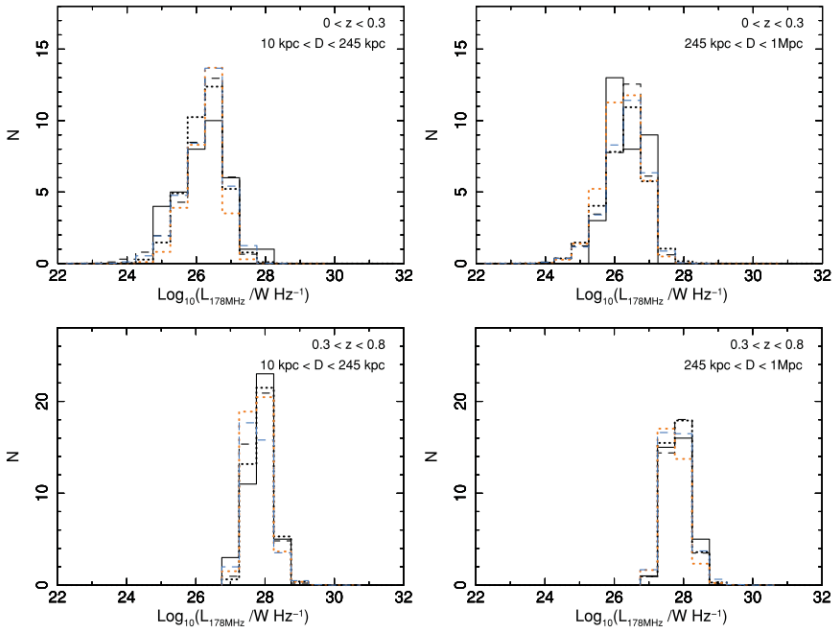
- Particle interactions/decays are random processes
- MC approach is essential to modelling the response of detectors to particles – the ones you want to detect as well as the unwanted background!
- GEANT4 software ‘simulates the passage of particles through matter’ http://geant4.cern.ch/
- Major applications in particle & nuclear physics, space-based detectors and medical imaging



The appearance of astrophysical populations

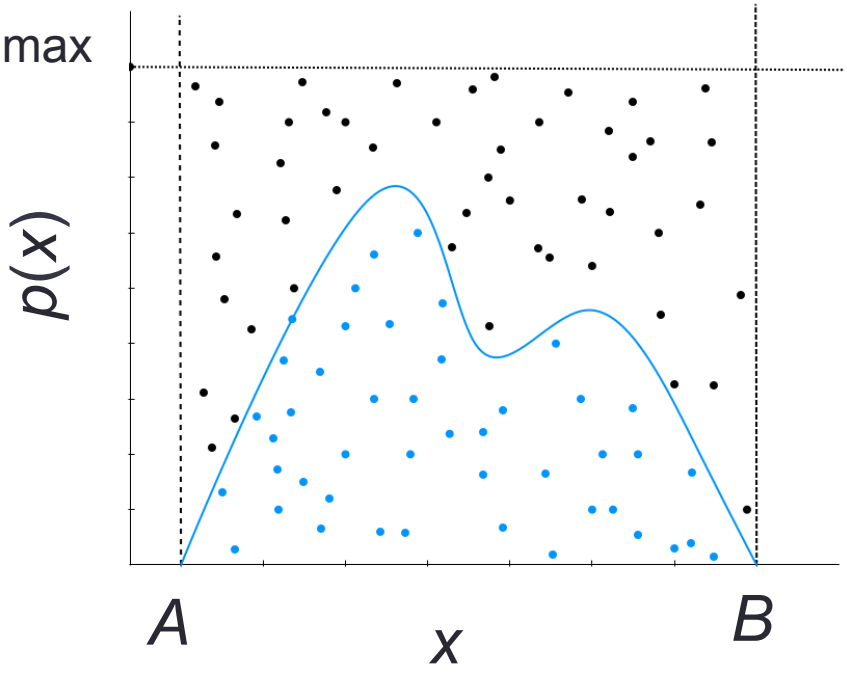
- Observed populations of astrophysical objects can be considered random samples from some underlying populations
- Biases also introduced by sensitivity, selection criteria etc.
- Problem even more complex if we want to compare observations with predictions of complex physical models
- E.g. Populations of FRII radio galaxies (Kapinska et al. 2012):
  - Model predicts radio luminosity and source size from jet power and particle Lorentz factor distribution, age, gas density of environment
  - Use MC to select from distributions of these parameters – observed flux depends on redshift z, may also evolve with cosmic time so randomly select z too!
  - Generate fake population, apply flux limit and compare with data (uses maximum likelihood, more next week…)
  - Do this for many assumed model parameter distributions, to find best-fitting model!





From *U(0,1)* to another distribution: the accept-reject method

- Less efficient than using the inverse cdf, but may be useful when the inverse cdf cannot be calculated.
- Consider pdf from range $x = A$ to B
- Generate two uniform random numbers drawn from $x = U(A,B)$ and $y = U(0,\max)$ where max denotes a maximum value searched (may correspond to maximum of pdf if known).

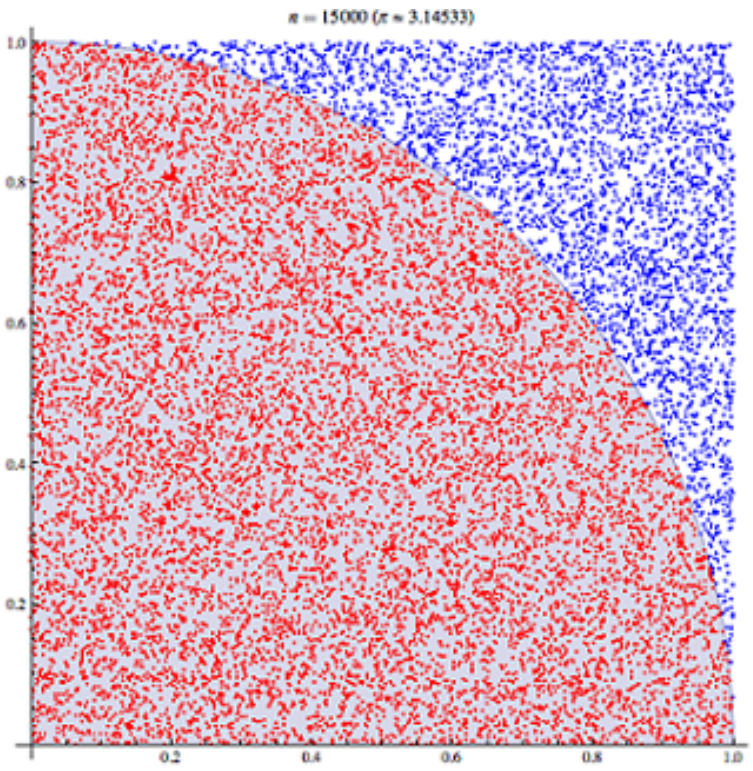


- Use only values $y < p(x)$ – discard larger values and draw again
- The resulting values drawn will have the pdf $p(x)$ in the range A to B
- Caveat: can be problems if most of pdf concentrated in narrow range (where to put A and B if distribution extends to large $|x|$?)

Monte Carlo integration

- We can use the accept-reject method to do integration!
- Simple example: what is the area of a circle?
  - Simulate n_{samp} data points using $U(0,1)$ to draw their x and y positions in a square of unit area.
  - Accept only $x^2 + y^2 \leq 1$
  - The Monte Carlo estimate of the area of the circle is equal to:

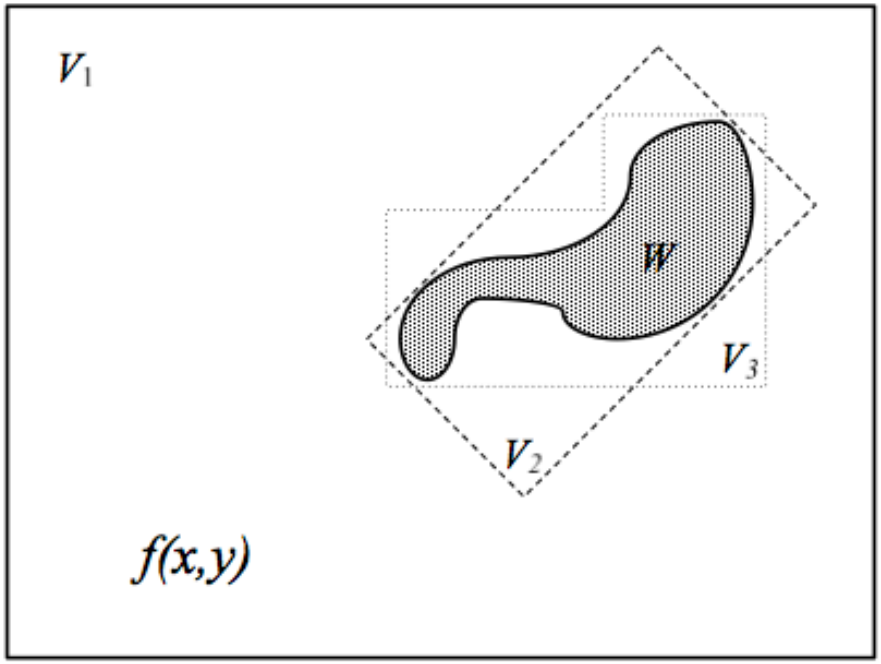
$$4 \times \frac{n_{\text{accept}}}{n_{\text{samp}}}$$



By choosing an appropriately-shaped sampling space and sufficient samples, Monte Carlo integration can be used to obtain much more difficult integrals, which may be impossible to solve analytically!

More general Monte Carlo integration

- Consider integrating over a more complex multi-dimensional volume *W*
- Sampling uniformly over the original axes, i.e. *x* and *y* in this case, is very inefficient.
- Instead, we can consider uniformly sampling a volume V_2 which is better matched to ours.
- Or for even more efficiency, we can build the sampled region from two or more uniformly sampled sub-regions, weighting the random selection of either according to their volume



More complex variants of these methods are called *importance sampling* and *stratified sampling* respectively – for more information see the Numerical Recipes 3rd Edition (available as a pdf online)

Mapping out a complex multi-dimensional probability density I

Consider a data set **D** which we want to fit using a model with multiple parameters given by the vector $\boldsymbol{\theta}$.

To determine the best-fitting parameters and confidence intervals, we would like to map out the probability density Pr($\boldsymbol{\theta}$|**D**).

Recapping Bayes theorem:

likelihood, prior, posterior, evidence

$$\Pr(\boldsymbol{\theta}|\mathbf{D}) = \frac{\Pr(\mathbf{D}|\boldsymbol{\theta})\Pr(\boldsymbol{\theta})}{\Pr(\mathbf{D})}$$

The evidence Pr(**D**) is the integral of the numerator over all possible $\boldsymbol{\theta}$, which can in practice be very difficult to compute. But the problem is simplified if we instead define:

$$\pi(\boldsymbol{\theta}) = \Pr(\mathbf{D}|\boldsymbol{\theta})\Pr(\boldsymbol{\theta})$$

where we now assume a given **D** so that for all $\boldsymbol{\theta}$, $\pi(\boldsymbol{\theta})$ scales with the true probability density.

Mapping out a complex multi-dimensional probability density II

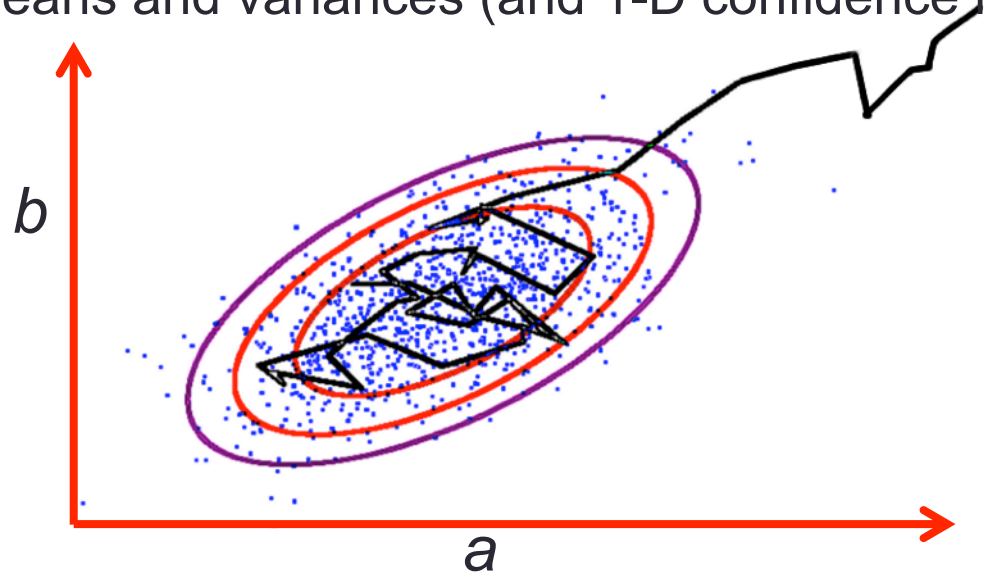
If we can map $\pi(\boldsymbol{\theta})$ we can still calculate the important quantities such as expectation values, variances and confidence intervals for our model parameters.

- In principle we can use a form of Monte Carlo integration by sampling points in the parameter space uniformly to sample the parameter space. This is very sub-optimal – can be very inefficient if $\pi(\boldsymbol{\theta})$ varies sharply in some places and much more smoothly in others.
- But what if we could find a method which automatically samples a region proportionately to the local $\pi(\boldsymbol{\theta})$? Sharper increases will also be sampled at a higher density, and the density of sampled points will also provide a measure of $\pi(\boldsymbol{\theta})$.

This latter approach is satisfied by *Markov Chain Monte Carlo (MCMC)*

MCMC visually

- Consider fitting to your data a model with two parameters, *a* and *b*.
- The confidence contours marking the exact probability distributions are shown but may not be easily determined.
- An MCMC approach (black line) can be used to map out the distribution: after the initial 'burn-in' phase (see later), the method starts to sample parameter values with density scaling with the local pdf.
- Sufficient samples may yield the blue points, which will enable contours to be mapped. However, only a small number are needed to estimate means and variances (and 1-D confidence intervals).



Markov Chains and random walks

Markov chain: stochastic (i.e. random) process where the probability of a future state is based only on the present state, not any previous ones.

E.g.: simple random walk ('Brownian motion'):

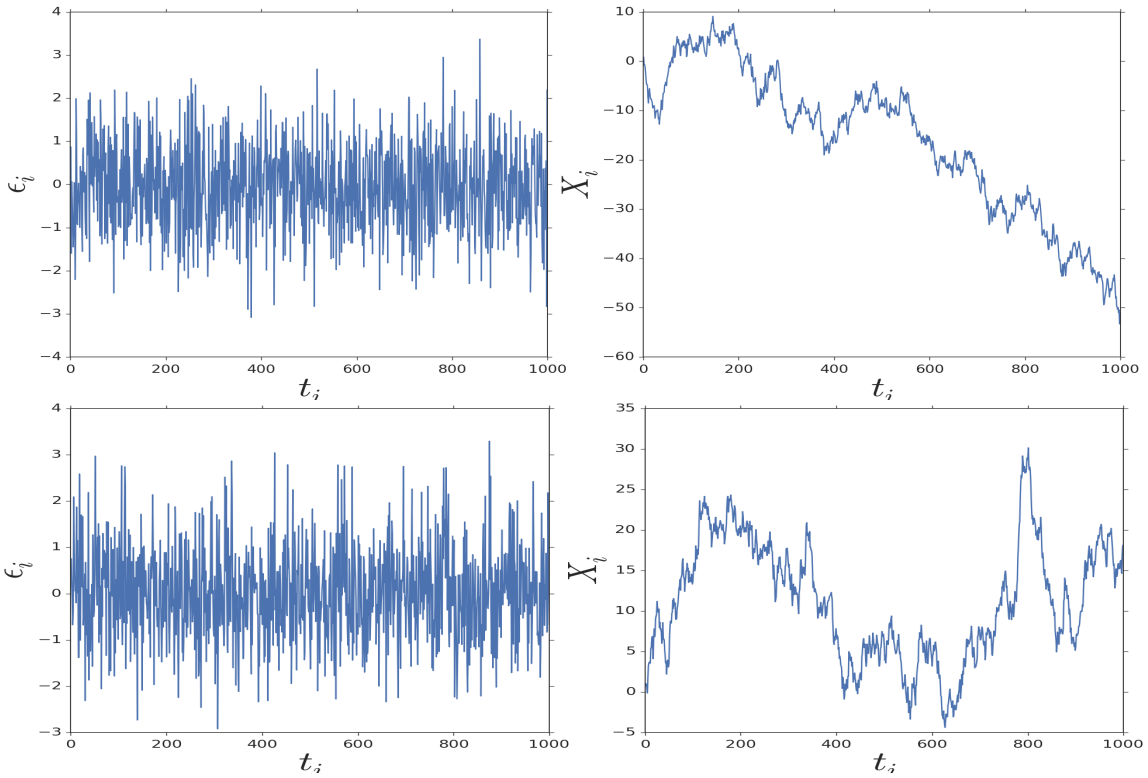
- Consider a sequence of ***independent identically distributed (i.i.d)*** random variables ϵ_i
- Now we build a new sequence X_i using the following simple rule:

$$X_i = X_{i-1} + \epsilon_i \qquad \text{i.e.} \qquad X_i = \sum_{j=1}^{i} \epsilon_j$$

- These are examples of one-dimensional stochastic processes. X_i is an example of a ***one-dimensional random walk***. It is easy to see in this case (week 3, linear functions of random variables) that for ϵ_i with mean μ_i and variance σ_i^2, after *n* steps:

$$\mathrm{E}[X] = \sum_{i=1}^{n} \mu_i \qquad \mathrm{V}[X] = \sum_{i=1}^{n} \sigma_i^2$$

1D random walk examples (same mean and variance of ε_i)



Our Markov chain should map out the parameter space. But for a simple random walk, different ***realisations*** of the same process can look very different even though they are produced by the same stochastic model! This is not what we want: we want our Markov chain be drawn from the same distribution (on average) each time.

Ergodicity and detailed balance

- The function $\pi(\boldsymbol{\theta})$ can be thought of as mapping a *phase space* of possible states of $\boldsymbol{\theta}$. For our Markov Chain process to successfully map out the distribution $\pi(\boldsymbol{\theta})$, it needs to be ***ergodic***.
- An ergodic process has the same behaviour averaged over time, as averaged over all of the states in its phase space.
- Now consider a Markov Chain where the current state (i.e. assumed set of parameter values) $\boldsymbol{\theta}_1$ transitions to a state $\boldsymbol{\theta}_2$ with probability $p(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)$, or vice versa.
- For a Markov Chain process to be ergodic, it simply needs to satisfy the ***detailed balance equation:***

$$\pi(\boldsymbol{\theta_1})p(\boldsymbol{\theta_2}|\boldsymbol{\theta_1}) = \pi(\boldsymbol{\theta_2})p(\boldsymbol{\theta_1}|\boldsymbol{\theta_2})$$

- This means that the transition rates from $\boldsymbol{\theta}_2$ to $\boldsymbol{\theta}_1$ are in balance (i.e. there is no overall time-evolution of the system and ergodicity is satisfied). Thermodynamically this is equivalent to a physical equilibrium so that $\boldsymbol{\theta_2} \longleftrightarrow \boldsymbol{\theta_1}$

More on detailed balance

We can get more insight by integrating the detailed balance equation with respect to $\boldsymbol{\theta}_1$:

$$\int p(\boldsymbol{\theta_2}|\boldsymbol{\theta_1})\pi(\boldsymbol{\theta_1})\mathrm{d}\boldsymbol{\theta_1} = \pi(\boldsymbol{\theta_2}) \int p(\boldsymbol{\theta_1}|\boldsymbol{\theta_2})\mathrm{d}\boldsymbol{\theta_1} = \pi(\boldsymbol{\theta_2})$$

The LHS gives the probability distribution for $\boldsymbol{\theta}_2$ after integrating over all transitions from $\boldsymbol{\theta}_1$. The RHS shows this is equal to $\pi(\boldsymbol{\theta}_2)$, implying that if $\boldsymbol{\theta}_1$ is drawn from π then so is its successor in the chain, $\boldsymbol{\theta}_2$.

In other words, if detailed balance applies, the probability distribution of sampled points becomes 'stationary' at the scaled version of the pdf, π, i.e. the process is ergodic and samples the correct distribution.

Metropolis-Hastings algorithm

Starting at $\boldsymbol{\theta}_1$, repeat the following sequence:

1. Pick a *proposal distribution* for the transition $\boldsymbol{\theta}_1 \rightarrow \boldsymbol{\theta}_2$: $q(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)$
   (e.g. q could be a multivariate normal centred on $\boldsymbol{\theta}_1$)

2. Generate a *candidate point* $\boldsymbol{\theta}_{2c}$ to transition to from $\boldsymbol{\theta}_1$

3. Calculate an *acceptance probability* that the candidate point will be used as the next step in the chain:

$$\alpha(\boldsymbol{\theta_1}, \boldsymbol{\theta_{2c}}) = \min\left(1, \frac{\pi(\boldsymbol{\theta_{2c}})q(\boldsymbol{\theta_1}|\boldsymbol{\theta_{2c}})}{\pi(\boldsymbol{\theta_1})q(\boldsymbol{\theta_{2c}}|\boldsymbol{\theta_1})}\right)$$

4. Accept the candidate point with probability $\alpha(\boldsymbol{\theta}_1, \boldsymbol{\theta}_{2c})$, otherwise reject and set $\boldsymbol{\theta}_2 = \boldsymbol{\theta}_1$

Metropolis-Hastings: satisfying detailed balance

The Metropolis-Hastings *transition probability* is given by:

$$p(\boldsymbol{\theta_2}|\boldsymbol{\theta_1}) = q(\boldsymbol{\theta_2}|\boldsymbol{\theta_1})\alpha(\boldsymbol{\theta_1}, \boldsymbol{\theta_2})$$

We can further show that M-H satisfied the detailed balance equation, by multiplying both sides of the acceptance probability equation by $\pi(\boldsymbol{\theta}_1)q(\boldsymbol{\theta_2}|\boldsymbol{\theta}_1)$:

$$\pi(\boldsymbol{\theta_1})q(\boldsymbol{\theta_2}|\boldsymbol{\theta_1})\alpha(\boldsymbol{\theta_1}, \boldsymbol{\theta_2}) = \min\left[\pi(\boldsymbol{\theta_1})q(\boldsymbol{\theta_2}|\boldsymbol{\theta_1}),\ \pi(\boldsymbol{\theta_2})q(\boldsymbol{\theta_1}|\boldsymbol{\theta_2})\right]$$

We can interchange the subscripts in the minimum term, and then rewrite in the same way as the LHS:

$$\begin{aligned} &= \min\left[\pi(\boldsymbol{\theta_2})q(\boldsymbol{\theta_1}|\boldsymbol{\theta_2}),\ \pi(\boldsymbol{\theta_1})q(\boldsymbol{\theta_2}|\boldsymbol{\theta_1})\right] \\ &= \pi(\boldsymbol{\theta_2})q(\boldsymbol{\theta_1}|\boldsymbol{\theta_2})\alpha(\boldsymbol{\theta_2}, \boldsymbol{\theta_1}) \end{aligned}$$

which, if we substitute in the transition probability equation, is the required detailed balance equation, QED.

The proposal distribution

How should one choose the proposal distribution, $q(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)$?

- Often, $q(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)$ only depends on the absolute difference $|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2|$, which simplifies the equation for the acceptance probability so that it only depends on the ratio $\pi(\boldsymbol{\theta}_{2c})/\pi(\boldsymbol{\theta}_1)$.
- A typical example is to assume $q(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)$ is a multivariate normal distribution (with a dimension for each parameter) centred on $\boldsymbol{\theta}_1$
- How then to choose the standard deviation in each parameter dimension?

The *acceptance fraction* a_f is the fraction of proposed steps which are accepted.

- If $a_f \sim 0$, very few steps are accepted and the chain will have few independent samples and will be very inefficient sampler of the target density $\pi(\boldsymbol{\theta})$.
- If $a_f \sim 1$, nearly all steps are accepted and the chain is effectively a random walk with no regard to target density (no ergodicity).
- Rule-of-thumb: a_f should be between 0.2 and 0.5 (Gelman, Rberts & Gilks 1996). You may need to experiment with your q in order to satisfy this.

Autocorrelation time and 'burn-in'

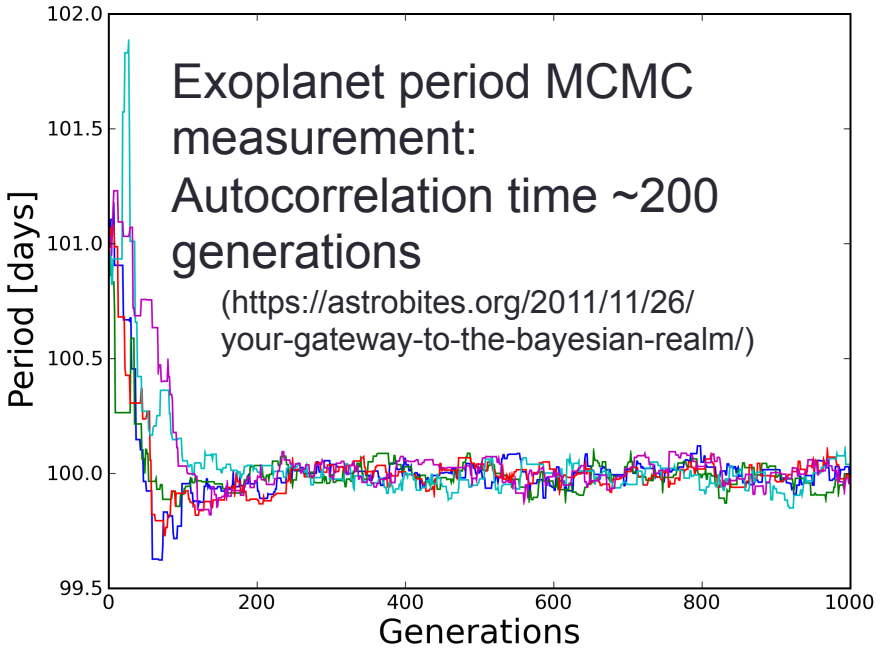
It may take some number of steps (or 'generations') for the MCMC process to reach equilibrium and so map out the probability density.

$$C(T) = \lim_{t \to \infty} \mathrm{cov}\left[X(t+T), X(t)\right]$$

The autocovariance of a time-series (e.g. parameter values stepped through) with itself *T* steps later, measures the degree of *autocorrelation*.

For samples to be independent (and therefore truly ergodic and the process to reach equilibrium), the covariance must go to zero.

The time it takes this to happen is called the *autocorrelation time*



To ensure that the probability density is correctly mapped, it is necessary to throw away a *burn-in* set of initial MCMC samples which exceeds the autocorrelation time. In the above example, the first 400 generations were thrown out.

More advanced MCMC methods

Metropolis-Hastings offers a basic start, but many more advanced approaches have been developed, which converge much more rapidly on the true distribution (e.g. using multiple 'walkers' to map the likelihood landscape more quickly).

A good starting point to using MCMC in your work is to install one of the common Python packages to do MCMC, e.g. now commonly used in astronomy, *emcee*: http://dan.iel.fm/emcee/current/

As always, be sure to understand the methods you use, and the caveats, assumptions and limitations to their use!

The *Advanced Statistics* course will cover the theory and application of MCMC in more detail….