# Lecture 3: an introduction to MCMC

Ben Lambert[1]
ben.c.lambert@gmail.com

[1]Imperial College London

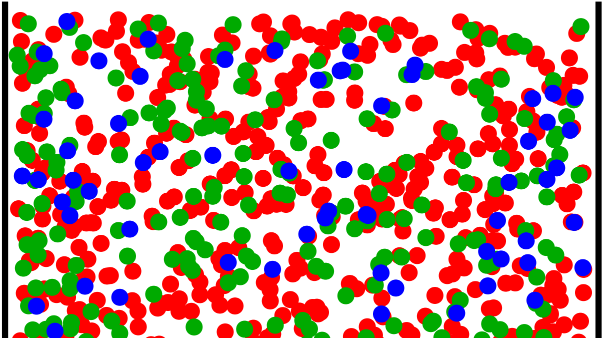
Tuesday 5th March, 2019

# Lecture outcomes

1. Appreciate how sampling can be used to gain insight into a distribution.
2. Grasp how **dependent sampling** via **MCMC** allows sampling from the posterior.
3. Understand the mechanics of Random Walk Metropolis and how it works intuitively.
4. Know that judging convergence of chains to the posterior is *hard*.
5. Be able to formulate the statistical inverse problem for ODEs.
6. Know what meant by ABC and when to use it.

# What is (independent) sampling and how can it give insight to distributions?

- Suppose we have a large (infinite) urn filled with coloured balls.
- The number of colours and the frequencies of each are **unknown**.
- **Question:** how can we determine the underlying probability distribution of ball colour?

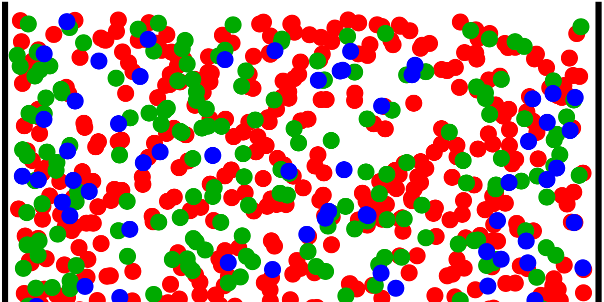


# What is (independent) sampling and how can it give insight to distributions?

**Answer:** we draw lots of balls from the urn and count the **sampled** frequencies!

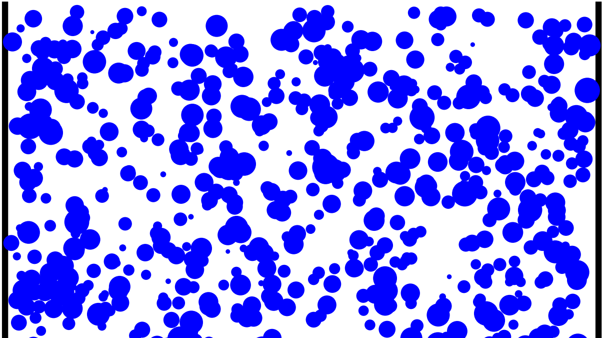
# What is (independent) sampling and how can it give insight to distributions?

- Drawing one ball from the urn is the act of taking a single **sample**.
- If the balls in the urn are swishing about then the colour of the next ball does not depend on the current ball's colour.
- Here the samples are (conditionally-) **independent**.
- Independent sampling gives us a very **efficient** way of gaining insight into a distribution.



# Sampling from a continuous distribution

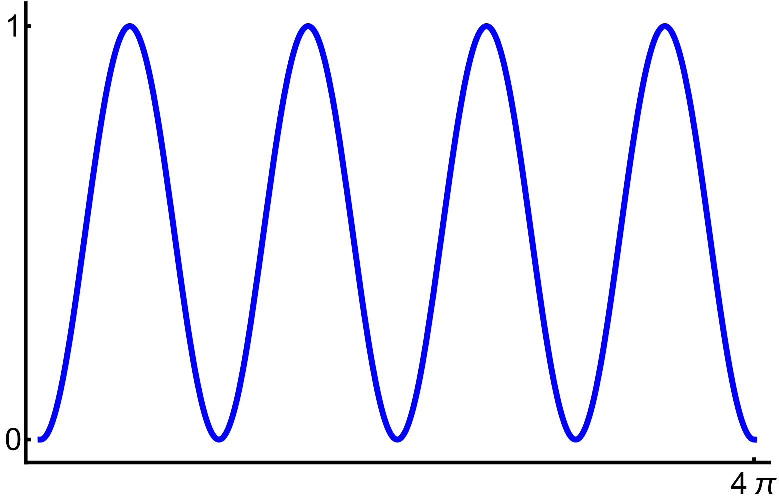
- Suppose we have a large (infinite) urn filled with balls of differing sizes.
- The distribution of sizes is **unknown**.
- **Question:** can we use same method to determine the underlying probability distribution of ball size? **Answer:** yes!



# Sampling from a continuous distribution

# Generating independent samples: sine curve

**Question:** how can we generate independent samples from the following (un-normalised) PDF?



# Generating independent samples: sine curve

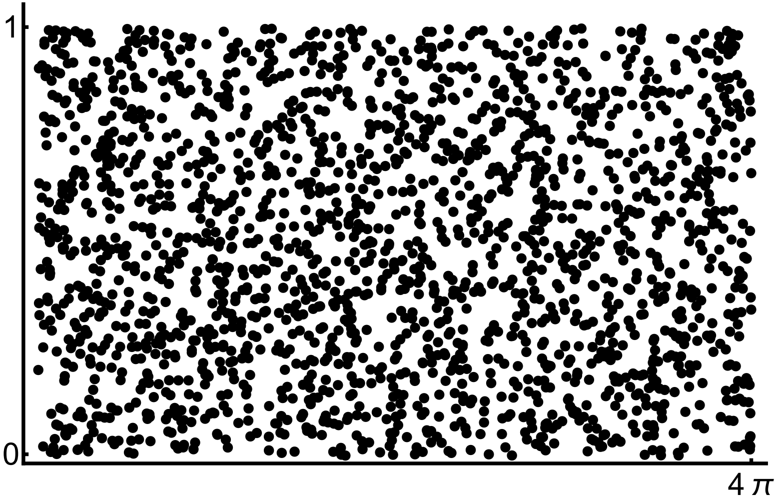
**Answer:** do the following a large number of times:

1. Generate **x** coordinates: uniformly-distributed points from $(0, 4\pi)$; where $4\pi$ is the domain of the function.
2. Generate **y** coordinates: uniformly-distributed points from $(0, 1)$; where 1 is the maximum value of the function.
3. If $y < p(x)$, then **accept** x coordinate as a sample.
4. If $y \geq p(x)$, then **reject** x coordinate as a sample.

Known as **Rejection** sampling.

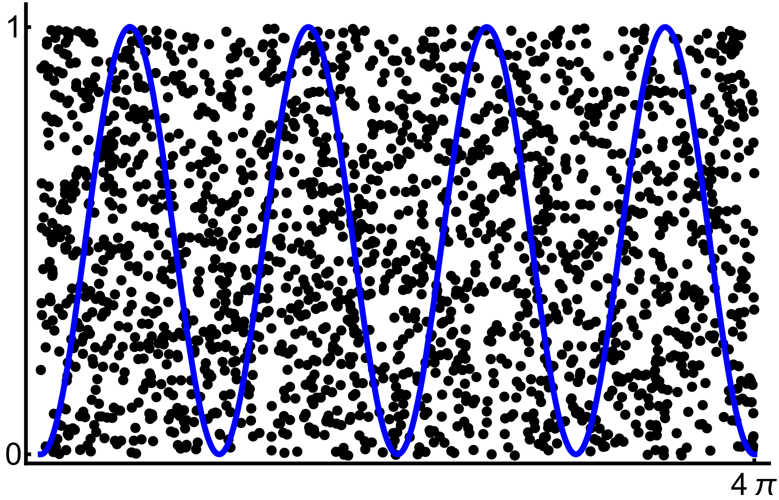
# Generating independent samples: sine curve

Generate **x** and **y** coordinates.



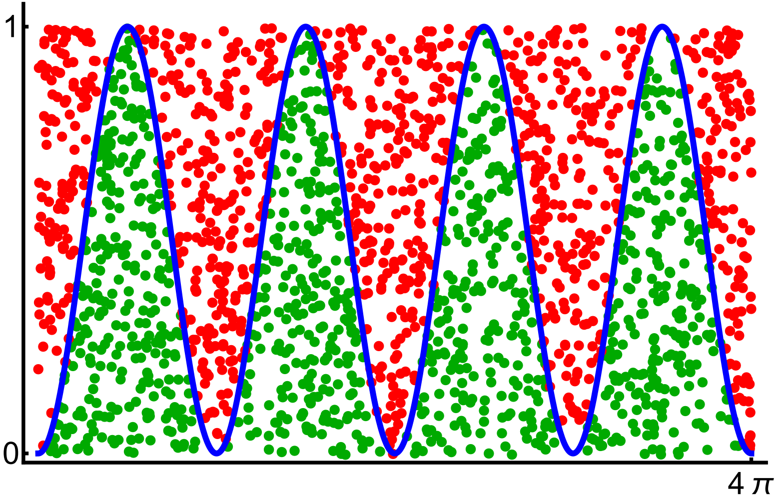
# Generating independent samples: sine curve

Overlay pdf.



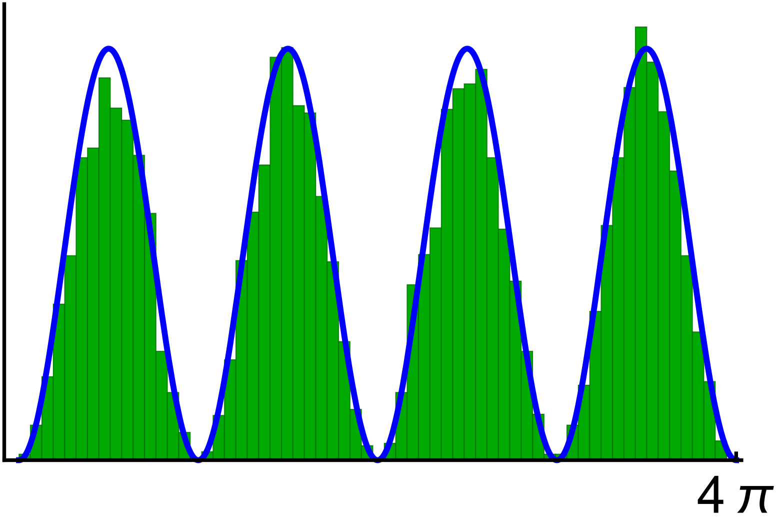
# Generating independent samples: sine curve

Accept **x** coordinates as samples if $y < p(x)$.



# Generating independent samples: sine curve

The resultant samples.



# Why do sampling in the first place?

Typically we want to calculate the posterior mean of some parameter, $\theta_1$:

$$
\begin{aligned}
\mathrm{E}(\theta_1|X) &= \int\limits_{\Theta_1}\int\limits_{\Theta_{-1}} \theta_1 \times p(\theta_1, \boldsymbol{\theta_{-1}}|X)\mathrm{d}\theta_{-1}\mathrm{d}\theta_1 \\
&= \int\limits_{\Theta_1} \theta_1 \times p(\theta_1|X)\mathrm{d}\theta_1
\end{aligned}
$$

where $\boldsymbol{\theta_{-1}}$ corresponds to the $d-1$ other parameters of the model.

This integral (the top line) is just too difficult to calculate exactly for all but the simplest models $\implies$ we instead use sampling to approximate it!

# Why is generating independent samples difficult?

- **Rejection sampling** requires generation of a large number of random points to produce relatively few samples.
- This inefficiency increases (exponentially) with the dimensionality of the distribution; i.e. for posteriors with more parameters.
- Other methods exist (inverse-transform sampling and importance sampling, for example) but they suffer from complexity and/or inefficiency issues.
- We cannot calculate the denominator so are unable to use some of these methods.
- Even if we had the denominator the complexity of most models means that independent sampling isn't possible.

# Is sampling finished?

# What is dependent sampling?

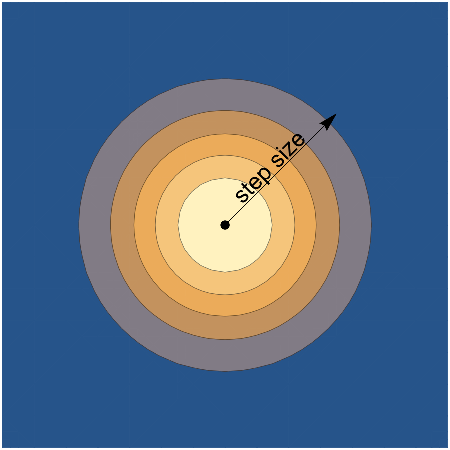
**Definition:**

"A sampling algorithm where the next sample **depends** on the current value."

**And** the list of all (accepted) positions of the sampler form the sample.

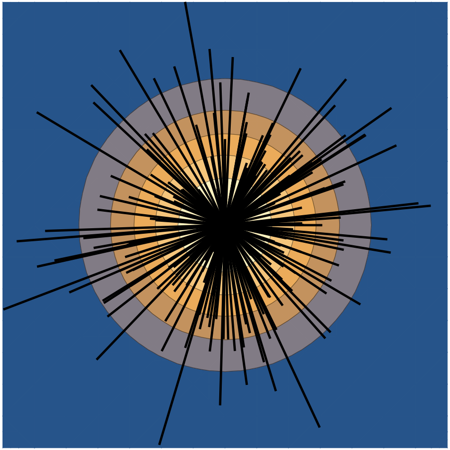
# Example dependent sampler: choose a new position based on a local “jumping” distribution

Suppose the next value of the sampler is drawn from a 2d normal distribution centred on our current position.

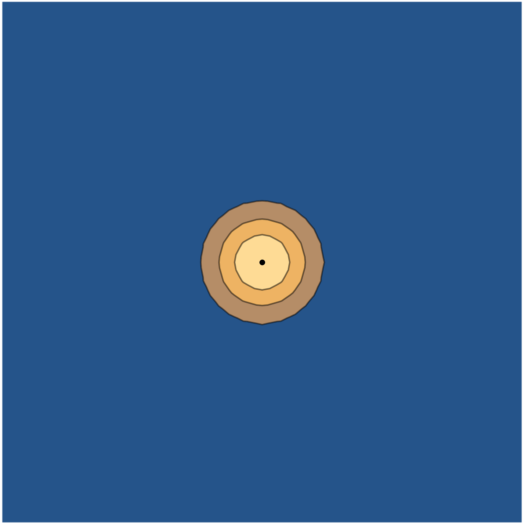


# Example dependent sampler: next steps

Showing 200 example steps.



# Visualising the path of a dependent sampling



# Dependent samplers as Markov Chains (Monte Carlo)

- Where to step next is determined via a distribution *conditional* on the current parameter value.
- This stepping is probabilistic $\implies$ *Monte Carlo*.
- The conditional distribution only depends on the current value of the sampler meaning it is memoryless about past path.
- This memoryless means that the path of the sampler is a *1st order Markov Chain*.

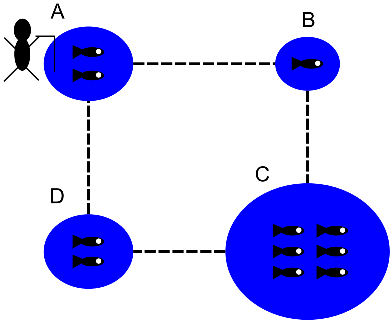
# Open questions

How can we decide on the:

1. Starting position.
2. Jumping distribution's shape.

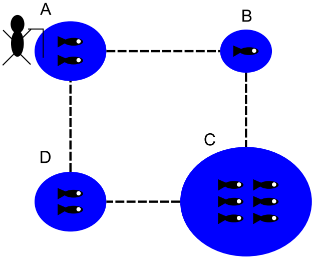
To ensure convergence to the posterior distribution? Especially because we cannot compute the posterior itself!

# David Robinson's fishing




- David Robinson (a more fortunate cousin of Robinson Crusoe) is marooned on an island.
- Access to four freshwater lakes of different sizes; each with a supply of fish.

# David Robinson's fishing



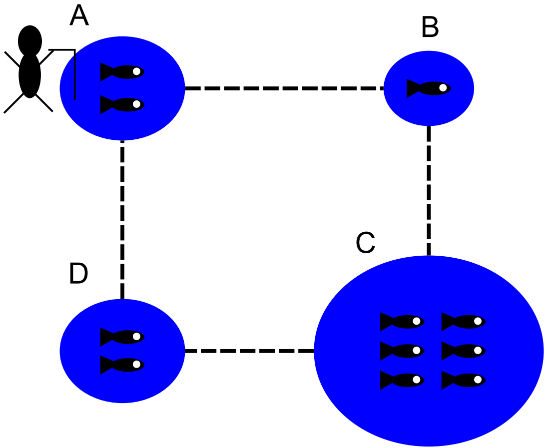

- Robinson does not know the amount of fish in each lake.
- He also does not know the number of lakes!
- However, the amount of fish in each lake is proportionate to its area.
- From a particular lake he can see the two adjoining lakes, and can estimate their area.

# David Robinson's fishing

- He has a terrible memory (too much coconut toddy), and each day forgets any estimates of lake size he made previously.
- He wants to fish (at maximum) one new lake per day.
- He possesses a coin and a solar-powered calculator that can generate (pseudo-)random numbers uniformly distributed between 0 and 1.
- He is initially "washed up" next to lake A.

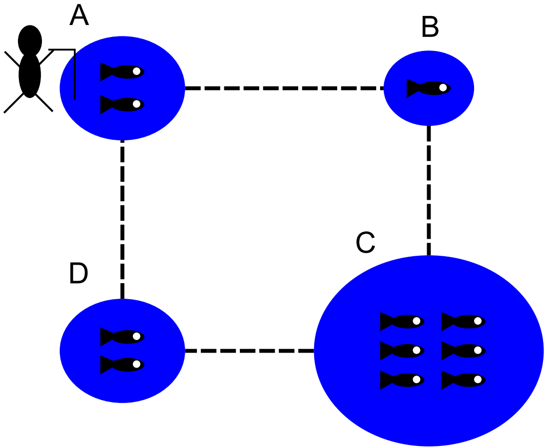
# David Robinson's fishing

- **Question:** What strategy should he use to fish as sustainably as possible?



# David Robinson's fishing

- **Remember:** Robinson doesn't know the # of lakes, nor the amount of fish in each!

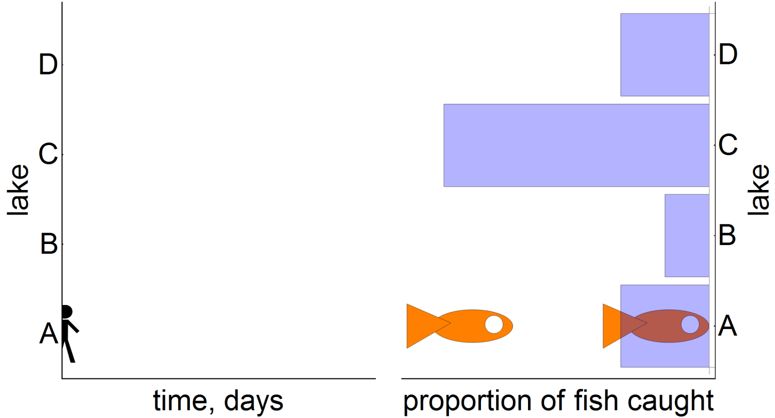


# David Robinson's fishing

**Answer:** visit each lake in proportion to the fish it contains, by doing the following:

1. Each night he flips the coin.
2. If it's heads (tails) he proposes a move to the neighbouring lake in the clockwise (anticlockwise) direction.
3. Calculates the ratio of the size of the proposed lake to the current one.
4. Compares the ratio with a (pseudo-)random number from the calculator.
5. If the ratio exceeds the generated number, he moves. If not, he stays put and fishes the same lake tomorrow.

# David Robinson's fishing: does it work?



# David Robinson's fishing: summary

- Robinson lacked knowledge of *numbers* of fish in each lake and the number of lakes.
- Knows that the number of fish in each lake is proportionate to its size.
- His memory stops him remembering the exact sizes.
- Each night he flips a coin; heads (tails) $\implies$ consider clockwise (anticlockwise) neighbouring lake.
- Estimates ratio of size of selected lake to current one.
- If ratio exceeds a uniform random number he moves. If not he stays where he is.
- After about 100 days his "random" strategy is quite similar from an "omniscient" one.

# Defining Random Walk Metropolis

Robinson's strategy is an example of the "Random Walk Metropolis" algorithm. This has the following form:

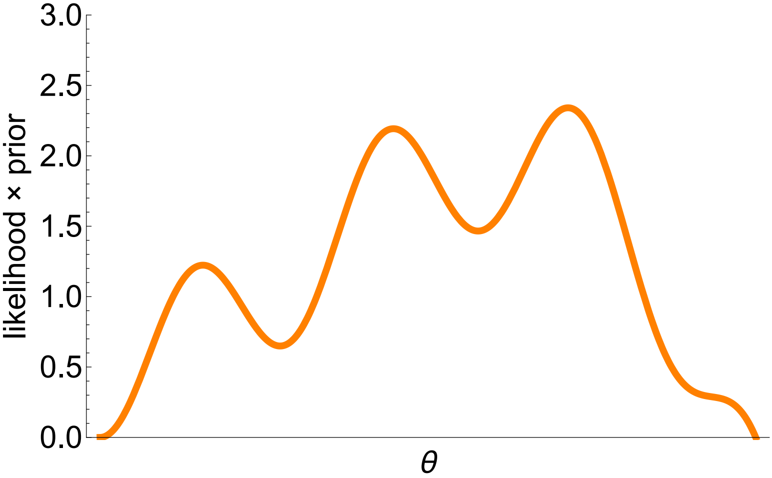
- Generate a random starting location $\theta_0$.
- Iterate the following for $t = 1, .., T$:
  - Propose a new location from a jumping distribution: $\theta_{t+1} \sim J(\theta_{t+1}|\theta_t)$.
  - Calculate the ratio:

$$r = \frac{\text{likelihood}(\theta_{t+1}) \times \text{prior}(\theta_{t+1})}{\text{likelihood}(\theta_t) \times \text{prior}(\theta_t)} \tag{1}$$

  - Compare $r$ with a uniformly-distributed number $u$ between 0 and 1.
  - If $r \geq u \implies$ we move.
  - Otherwise, we remain at our current position.

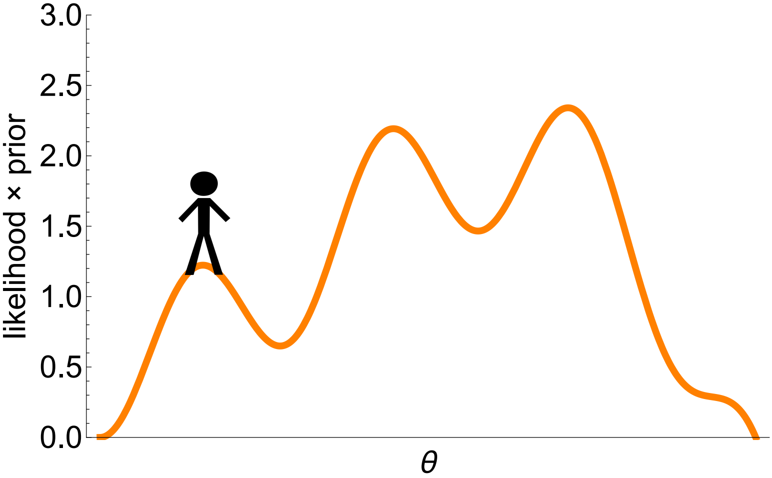
# Defining Random Walk Metropolis

Start with the un-normalised density.



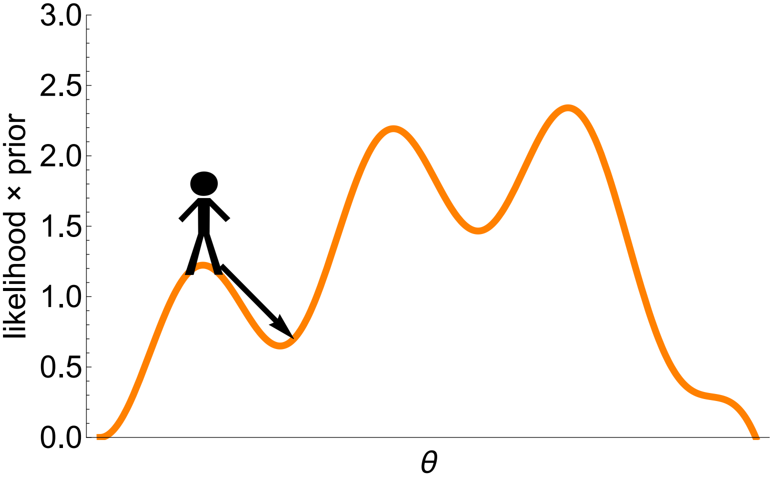
# Defining Random Walk Metropolis

Select a random starting location.



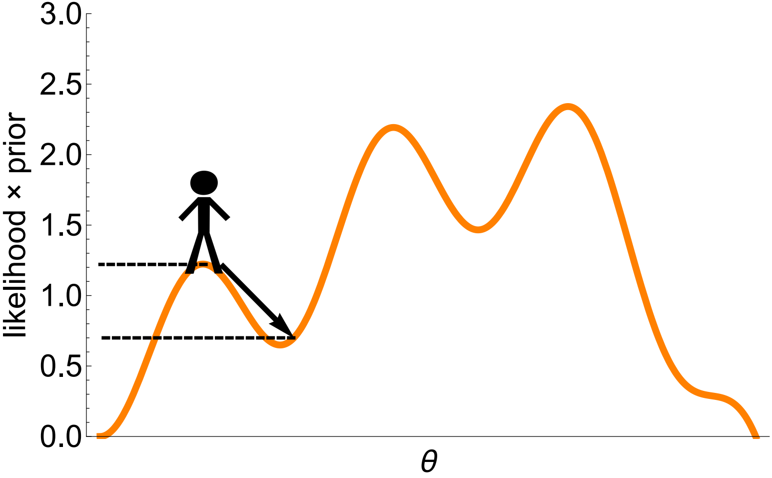
# Defining Random Walk Metropolis

Propose a new location using jumping distribution.



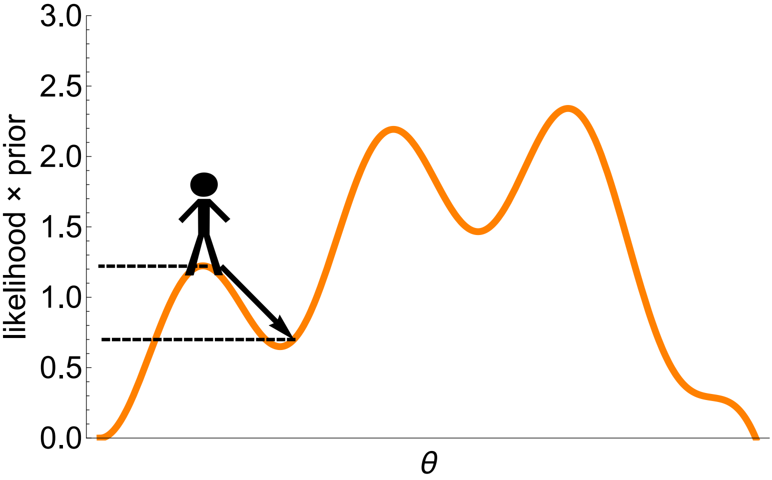
# Defining Random Walk Metropolis

Calculate ratio of likelihood $\times$ prior at proposed to current location, and find $r \approx 0.58$.



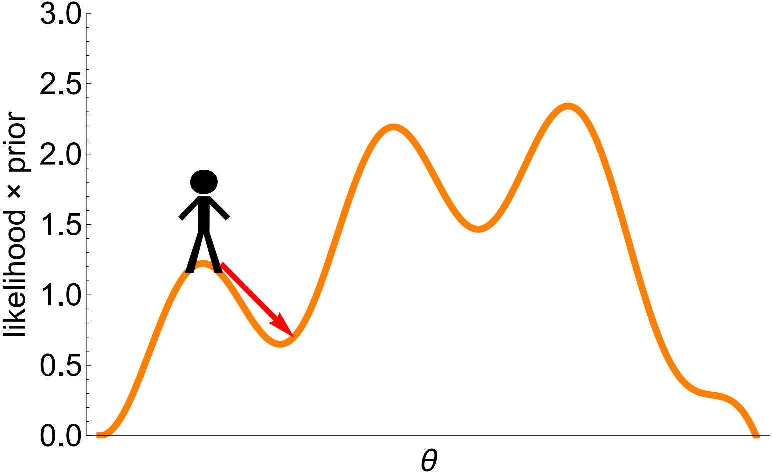
# Defining Random Walk Metropolis

Compare $r \approx 0.58$ with random real between 0 and 1. For example suppose we obtain $u = 0.823$.



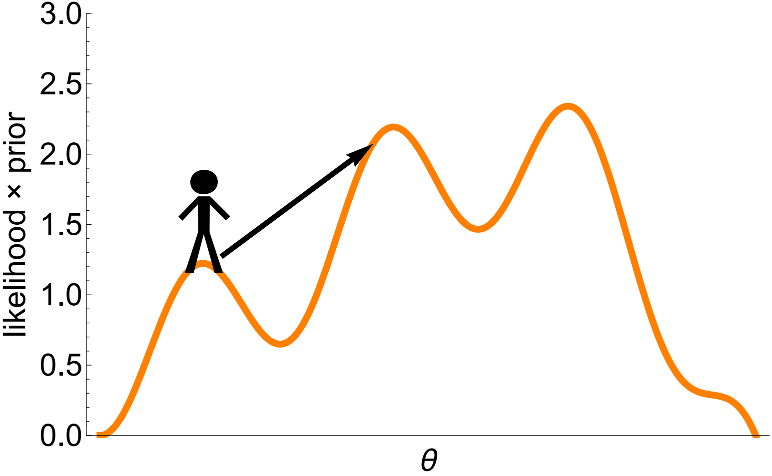
# Defining Random Walk Metropolis

Since $r < u$ we remain at our original location.



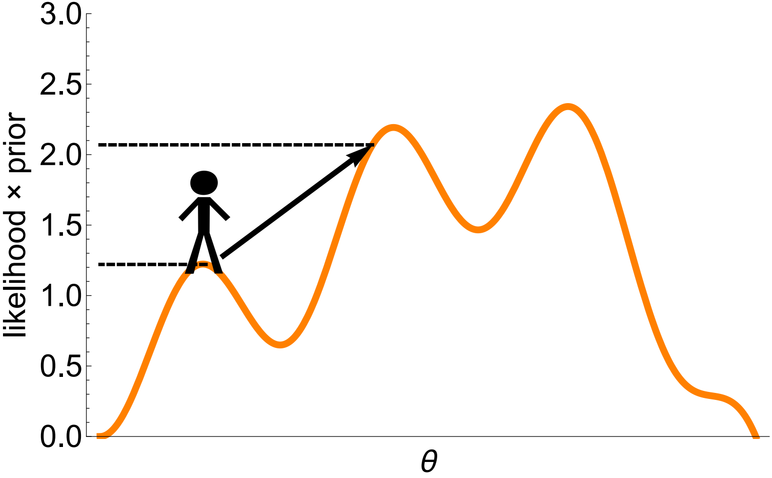
# Defining Random Walk Metropolis

Generate a new proposed step using jumping distribution.



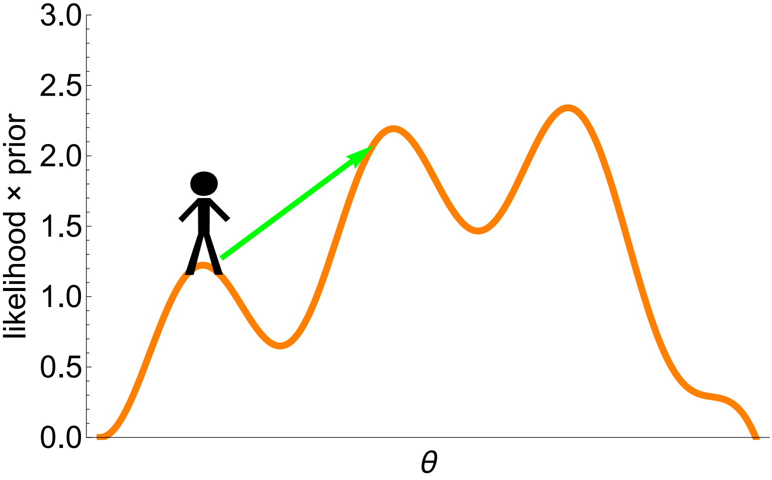
# Defining Random Walk Metropolis

Calculate ratio of likelihood $\times$ prior at proposed to current location, and find $r \approx 1.75$.



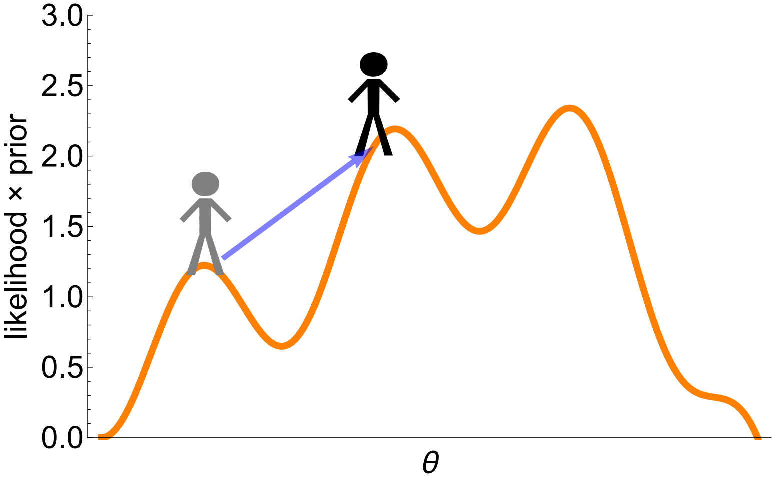
# Defining Random Walk Metropolis

Since $r > 1$ (maximum possible $u$) $\implies$ we move to new location.



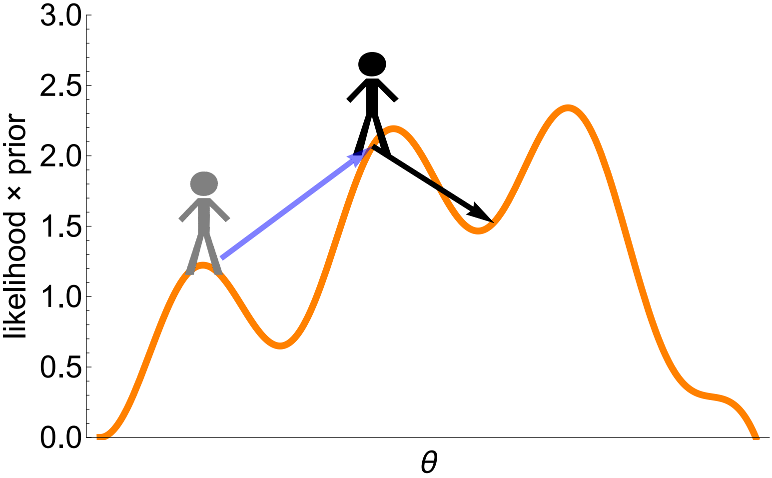
# Defining Random Walk Metropolis

Since $r > 1$ (maximum possible $u$) $\implies$ we move to new location.



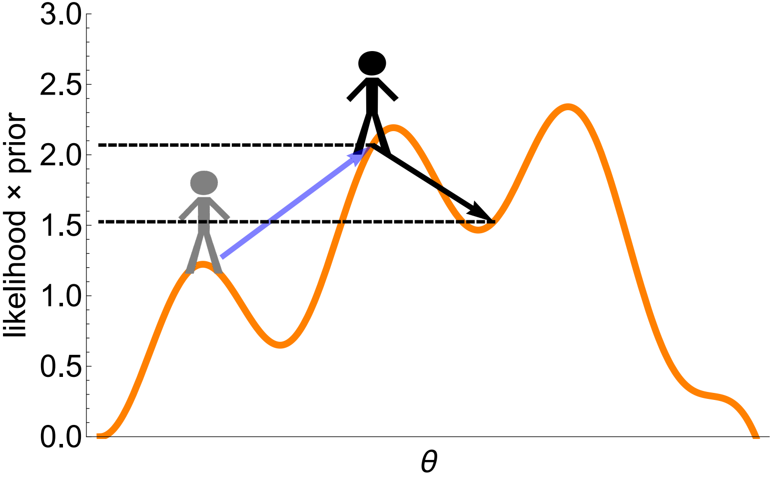
# Defining Random Walk Metropolis

Propose a new step using jumping distribution.



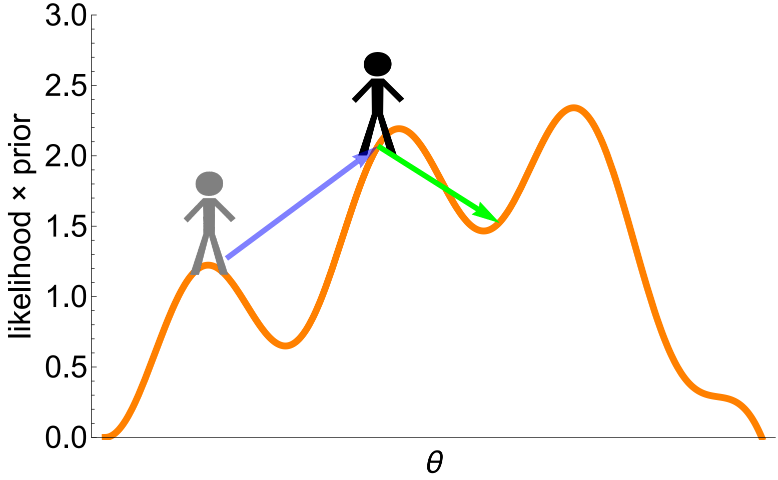
# Defining Random Walk Metropolis

Calculate $r \approx 0.75$.



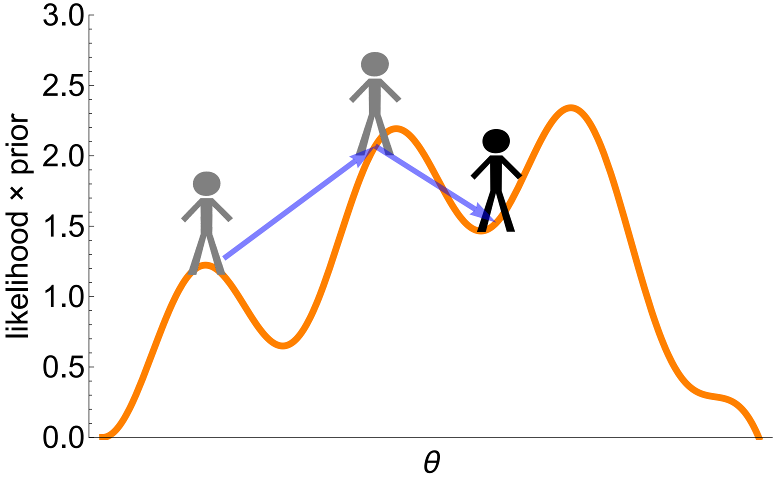
# Defining Random Walk Metropolis

Generate $u = 0.278 < r \implies$ we move!



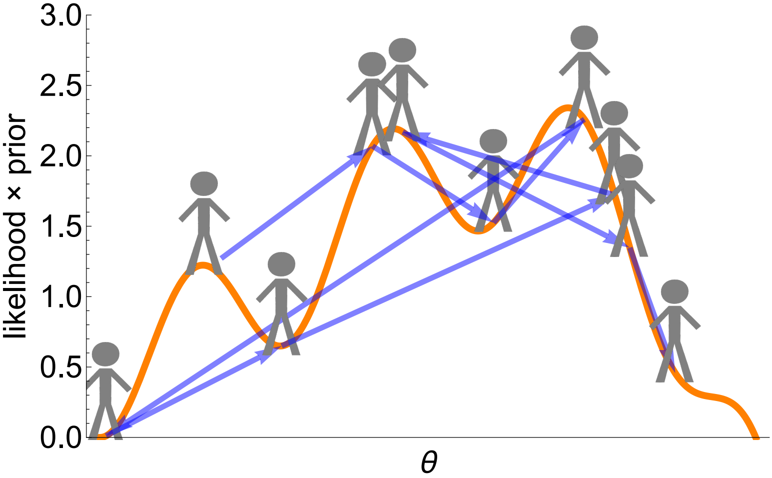
# Defining Random Walk Metropolis

Generate $u = 0.278 < r \implies$ we move!



# Defining Random Walk Metropolis

Repeat a large number of times.



# Random Walk Metropolis: benefits

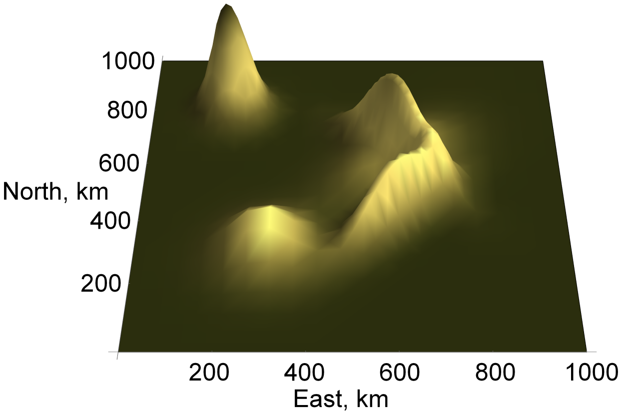
- Under quite general conditions the Random Walk Metropolis sampler converges **asymptotically** to the posterior.
- However for a sufficiently large sample size the sampling distribution may be practically indistinguishable from the true posterior.
- The algorithm requires us to be able to calculate the ratio:

$$r = \frac{\text{likelihood}(\theta_{t+1}) \times \text{prior}(\theta_{t+1})}{\text{likelihood}(\theta_t) \times \text{prior}(\theta_t)} \tag{2}$$

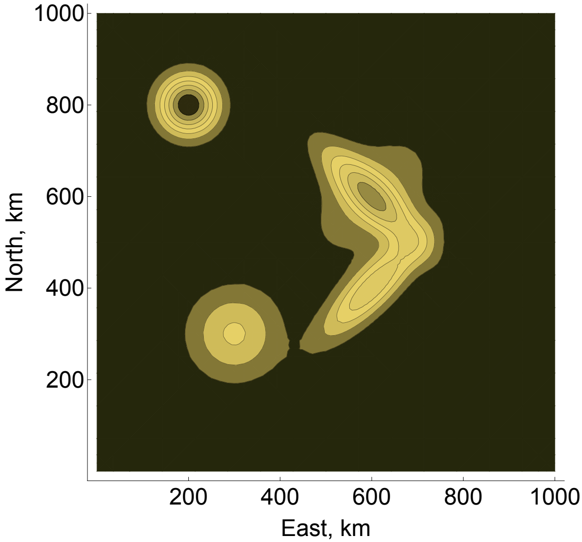
- The ratio uses **only** the numerator of Bayes' rule $\implies$ we side-step calculating the denominator!

# Random Walk Metropolis in action

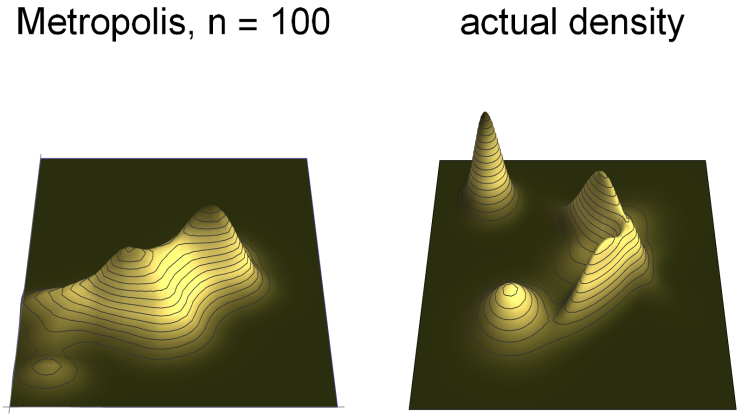
Can we use Random Walk Metropolis to sample from the continuous distribution below?



# Random Walk Metropolis in action



# Random Walk Metropolis in action



## Random Walk Metropolis: short summary

- Algorithm works by starting in a randomly-determined position in parameter space.
- In each iteration we generate a proposed (local) step from our current position.
- We then move based the ratio of the proposed **un-normalised** posterior to our current location $\implies$ no need to calculate troublesome denominator.
- The path of our positions over time forms our **sample**.
- If we repeat the above for a (large) number of steps $\implies$ sampling distribution $\approx$ posterior.
- **Question:** what's the function of the accept/reject rule we use in the Metropolis algorithm?

# The importance of the accept/reject rule

Let's try out three different accept/reject rules to see how they fare.

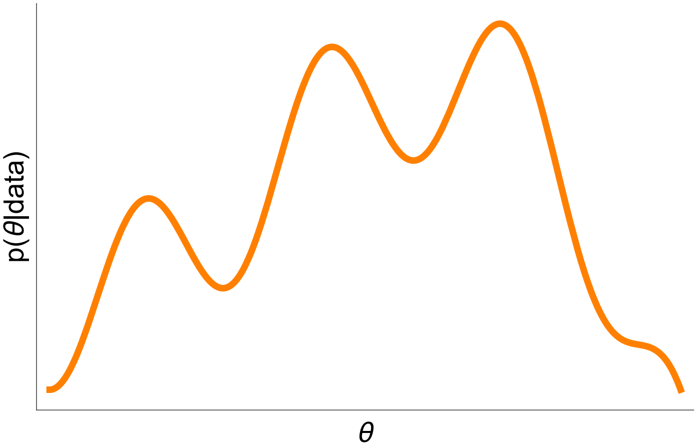
- **Drunkard's rule:** we always move irrespective of value of un-normalised density at new location versus current position.
- **Edmund Hilary's rule:** calculate

$$r = \frac{p(X|\theta_{t+1})p(\theta_{t+1})}{p(X|\theta_t)p(\theta_t)} \tag{3}$$

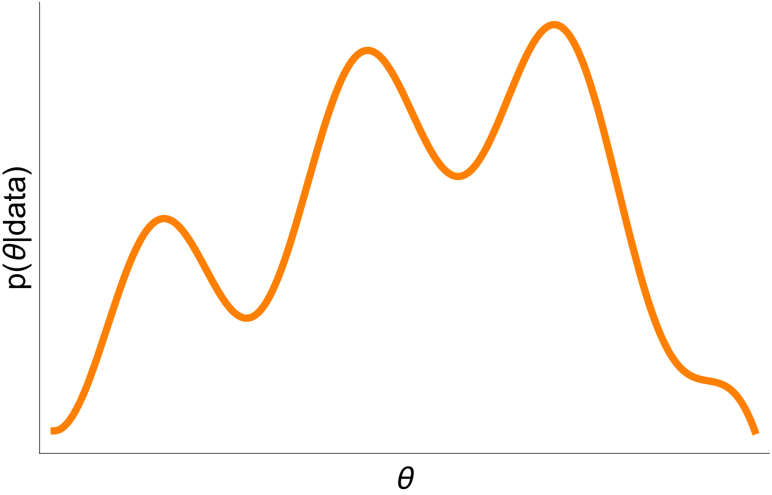
  - If $r > 1$ we move; otherwise don't.
- **Metropolis rule:**
  - If $r > u \sim Unif(0, 1)$, then move to new location.
  - Otherwise stay in current position.

# An example un-normalised posterior

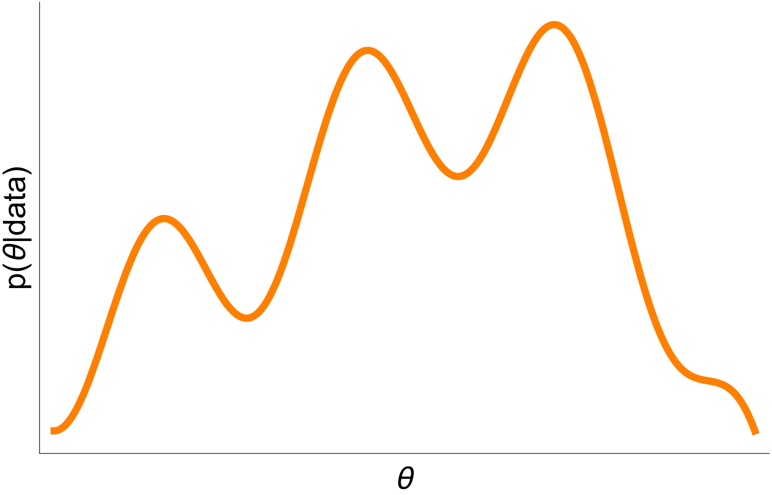
Start with the below distribution and try each different stepping rule.



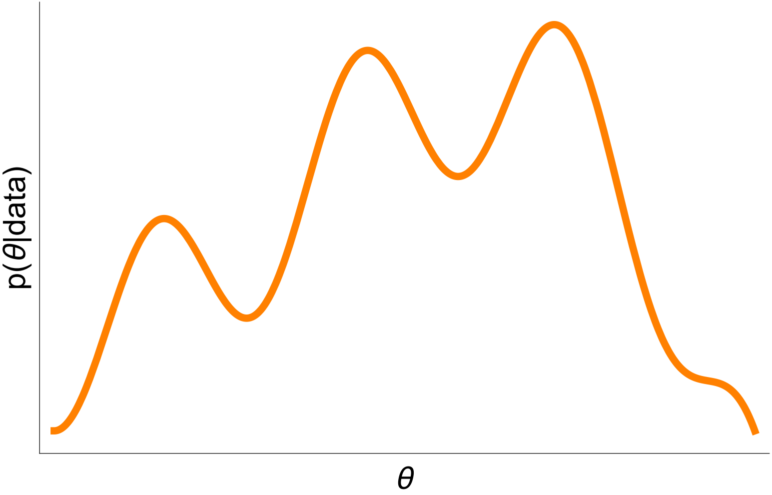
# Drunkard's walk



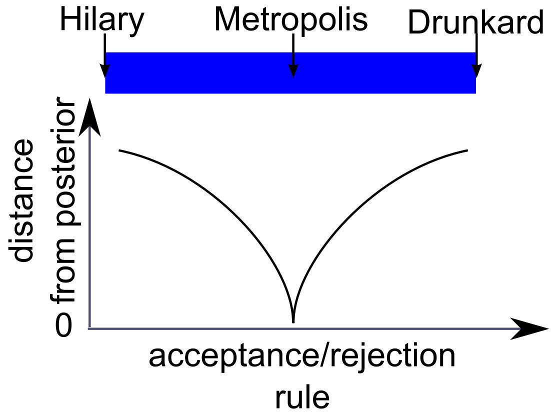
# Hilary's ascent



# Random Walk Metropolis



# Accept reject rule: summary




Only the Metropolis accept/reject rule allows sampling from each point in exact proportion to the posterior height.

## The intuition behind Random Walk Metropolis

Consider the ratio of the posterior density at point $\theta_{t+1}$ to $\theta_t$:

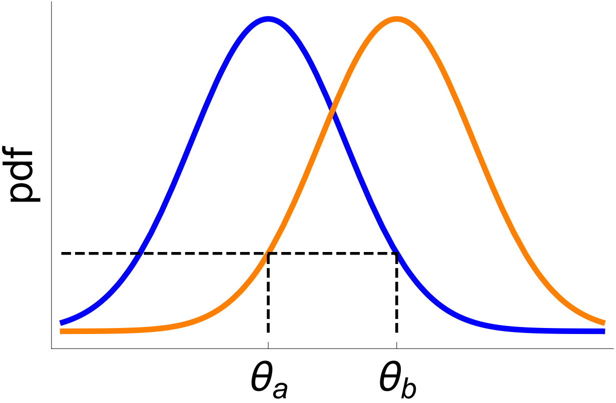
$$
\begin{aligned}
r &= \frac{p(\theta_{t+1}|X)}{p(\theta_t|X)} \\
&= \frac{\frac{p(X|\theta_{t+1})p(\theta_{t+1})}{p(X)}}{\frac{p(X|\theta_t)p(\theta_t)}{p(X)}} \\
&= \frac{p(X|\theta_{t+1})p(\theta_{t+1})}{p(X|\theta_t)p(\theta_t)}
\end{aligned}
$$

So the ratio of the numerators of Bayes' rule is **identical** to the ratio of the actual posteriors.

$\implies$ if we use $r$ to guide our stepping we will be sampling (eventually) from the posterior.

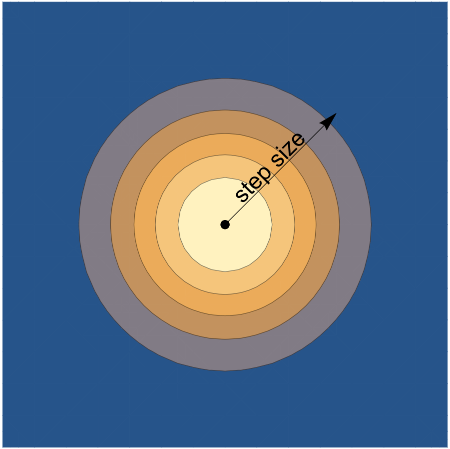
# How do we choose the jumping distribution?

- Sometimes called the "proposal distribution".
- In Random Walk Metropolis we use a symmetric distribution (relaxed in Metropolis-Hastings):
  $\implies J(\theta_a|\theta_b) = J(\theta_b|\theta_a)$



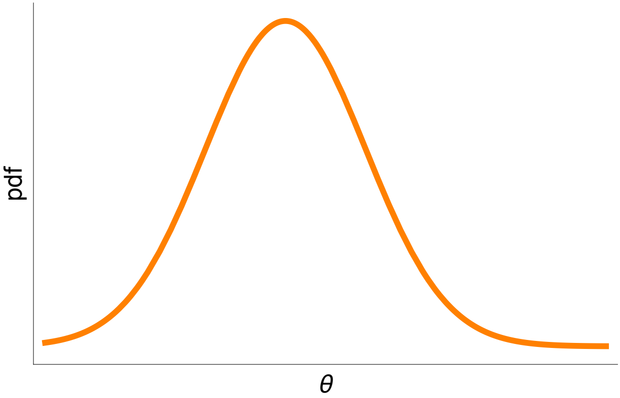
# The importance of step size

**Question:** how should we decide on the jumping kernel's step size?



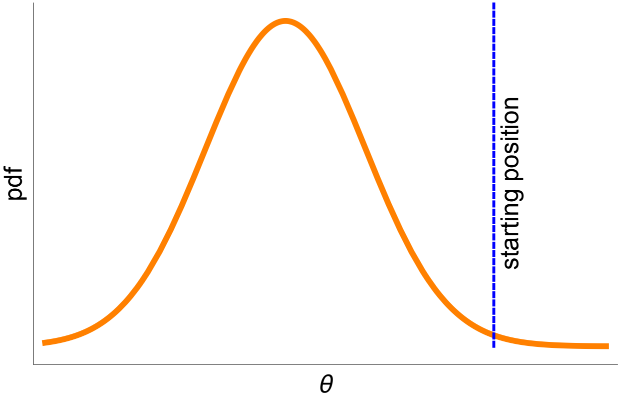
# Another example posterior distribution

Assume a unimodal distribution from which we want to sample.

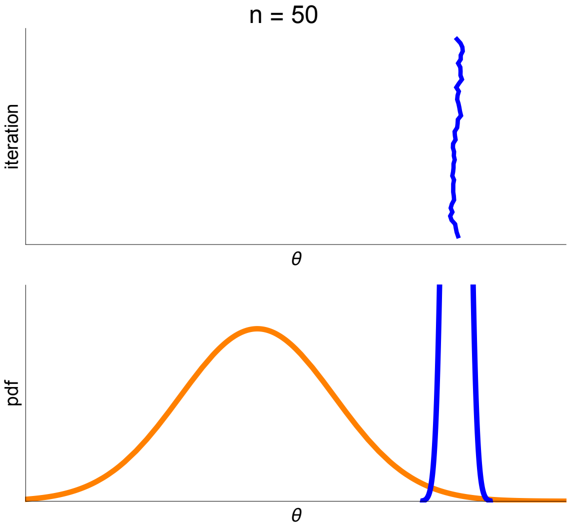


# Another example posterior distribution

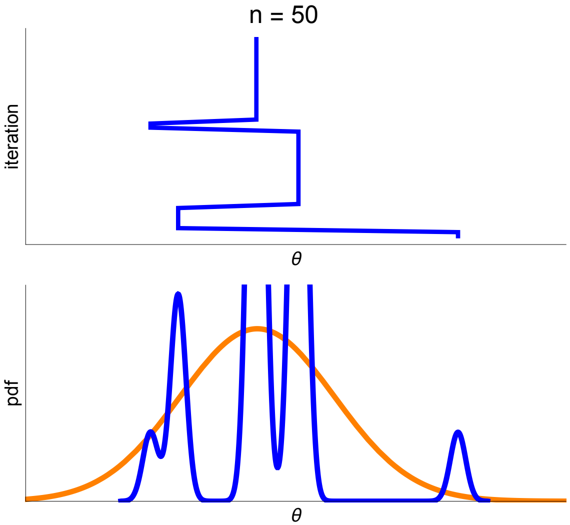
Start three algorithms with different step sizes at same point.



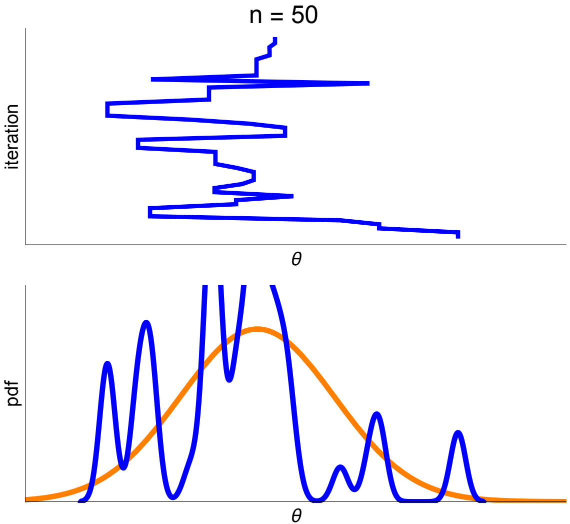
# The importance of step size: too small



# The importance of step size: too large



# The importance of step size: just right



## Step size: summary

- Whilst step size does not affect asymptotic convergence, it does affect finite sample performance.
- If step size is too small we do not find the typical set (area of high probability mass).
- If step size is too large we find the typical set, but do not explore it efficiently.
- Therefore do an initial run of sampler to find optimal step size before starting proper.

# What do we mean by convergence?

Recap the steps of Metropolis:

1. Propose an initial position $\theta_0$ using a initial proposal distribution $\pi(\theta) \neq p(\theta|X)$.
2. For $t = 1, ..., T$ do:
    - Propose a new location: $\theta_{t+1} \sim J(\theta_{t+1}|\theta_t)$.
    - Accept/reject move based on

$$r = \frac{p(X|\theta_{t+1})p(\theta_{t+1})}{p(X|\theta_t)p(\theta_t)} > u \sim Unif(0,1) \tag{4}$$

# What do we mean by convergence?

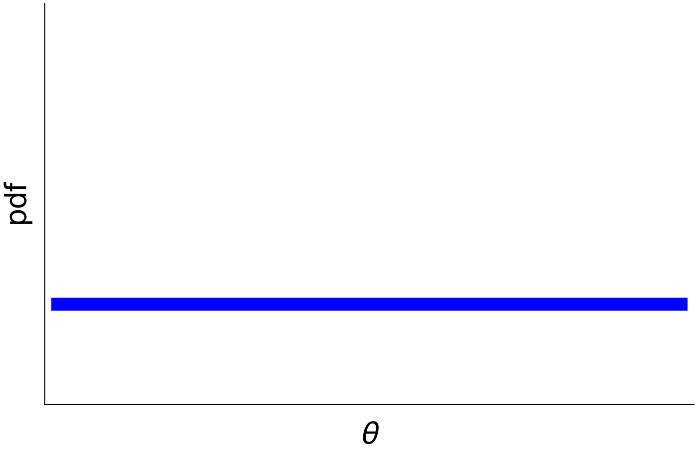


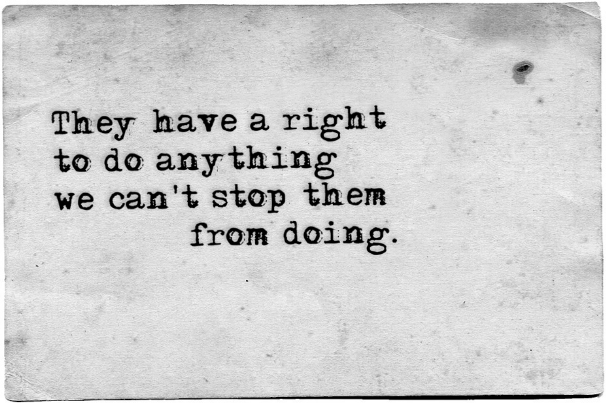

Figure: Adapted from Betancourt lecture: `www.youtube.com/watch?v=pHsuIaPbNbY`.

# Why do we need to monitor convergence?

- Start with an initial proposal distribution $\pi(\theta) \neq p(\theta|X)$.
- Repeatedly take steps and use the Metropolis accept/reject rule $\implies \pi(\theta_t)$; the sampling distribution at time $t$.
- Under a set of quite general assumptions we are guaranteed that asymptotically: $\pi(\theta_t) \rightarrow p(\theta|X)$.
- However, when practically can we assume: $\pi(\theta_t) \approx p(\theta|X)$?

# How to measure convergence?

- To monitor convergence to the posterior $\Longrightarrow$ need the posterior.
- But we don't have the posterior $\Longleftarrow$ the reason we are doing the sampling in the first place!



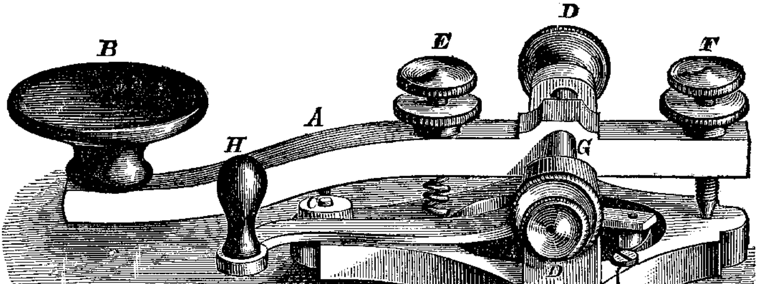
# Two strategies for monitoring convergence

**Strategy 1:** measure distributional separation.

- For example Kullback-Leibler:

$$KL = \int p(\theta|X) log \left( \frac{p(\theta|X)}{\pi(\theta_t)} \right) \mathrm{d}\theta \tag{5}$$

- Motivated by information theory.
- Can use un-normalised posterior to do this.
- Again integral is too difficult to do.



# Two strategies for monitoring convergence

**Strategy 2:** monitor the approach to a stationary distribution.

- We know asymptotically this will happen.
- By design of Metropolis stepping and accept/reject rules, we know the stationary distribution is the posterior.

# Monitoring convergence of a single chain

Initial idea:

- Compare summaries (mean, variance, etc.) of sampling distribution for a chain at time $t$ with itself at time $t + T$.
- If their rate of change is below a threshold $\implies$ convergence.

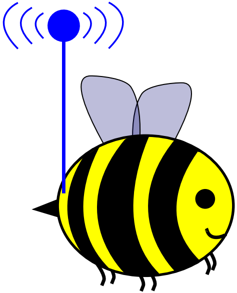
## Monitoring convergence of a single chain

**Question:** What is the problem with this idea?

# Convergence monitoring: Bob's bees

Thought experiment:

- Imagine a house of unknown shape.
- We have an unlimited supply of bees, each equipped with a GPS tracker allowing us to accurately monitor their position.
- **Question:** How can we use these to estimate the shape of the house?



# Convergence monitoring: Bob's bees

**Answer**:

- Release one (at a random location in the house) and monitor its path over time.
- Stop/collect bee after summary measures of its path stop changing.

# Convergence monitoring: single bee

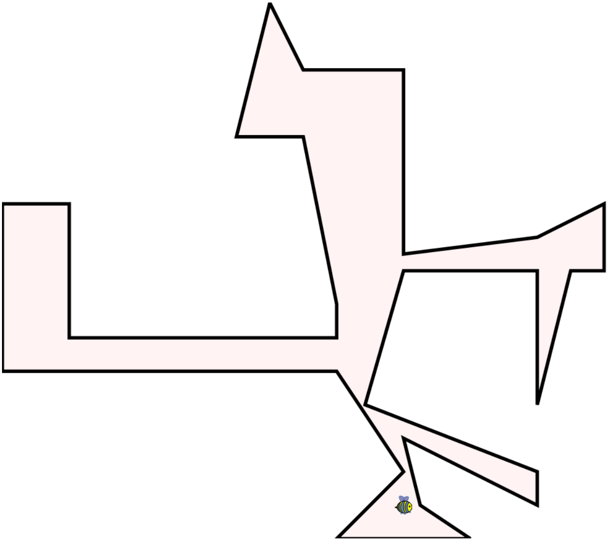
# Convergence monitoring: single bee, a bit later

# Convergence monitoring: single bee, a bit bit later

# Convergence monitoring: single bee

**Question:** what's the actual shape of the house?

# Convergence monitoring: single bee

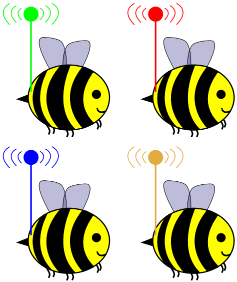


## Single chain problems: summary

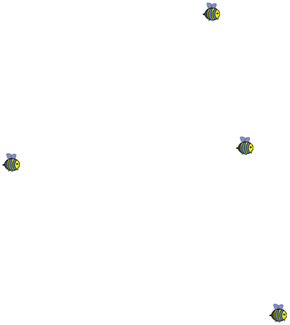
- One way to monitor convergence is to look for convergence in a single chain's summary statistics.
- This method is very susceptible to the curse of hindsight problem ("Now we've definitely converged on the posterior. We hadn't a minute ago.")
- Particularly because chains often get stuck in subregions of $\theta$ space.

# The solutions: lots of bees

- Release lots of bees starting at dispersed locations in parameter space.
- Stop recording when an individual bee's path is indistinguishable from all others'.



# Convergence monitoring: multiple bees



# Convergence monitoring: multiple bees (a lot later)

# Multiple chain convergence monitoring: summary

- Start a number of chains in random dispersed locations in $\theta$ space.
- Chains do *not* interact with one another (in Metropolis).
- Run each sampler until it is hard to distinguish one chain's path from all others'.
- Less susceptible to "curse of hindsight", since we can see if chains aren't mixing.
- Not foolproof! There still may be an area of high probability mass that we miss. However, less likely to fail compared to a single chain.
- The more chains, the better!

# Judging convergence

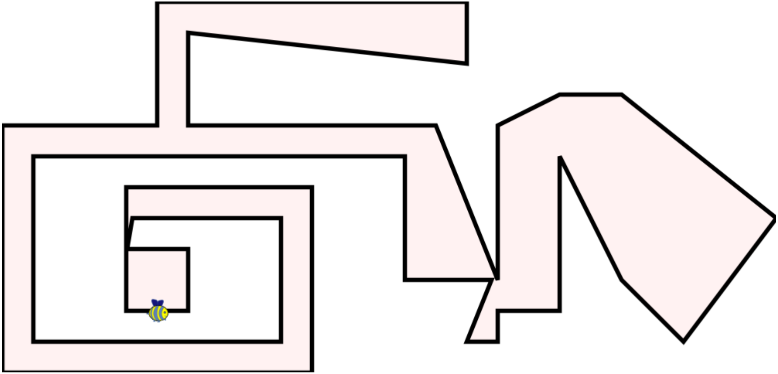
Single bee in a house.

# Judging convergence

Multiple bees in a house released in a single room.

# Judging convergence

**Question:** have we converged?



# Judging convergence

Multiple bees in new house released in highly dispersed rooms.

# Judging convergence

Multiple bees in new house released in highly dispersed rooms...much later.

# Multiple chain convergence monitoring: open questions

1. How to determine “random dispersed locations” at which to start the chains?
   - Ideally use an initial proposal distribution similar to posterior shape.
   - Otherwise a good rule of thumb is “Any point you don’t mind having in a sample is a good starting point”, Charles Geyer.
2. Which summary statistics to monitor to determine convergence?
3. At what threshold are “between chain” statistics sufficiently similar?

# Gelman and Rubin's $\hat{R}$

- Gelman and Rubin (1992) had the idea of comparing within-chain to between-chain variability.
- They quantified this comparison using:

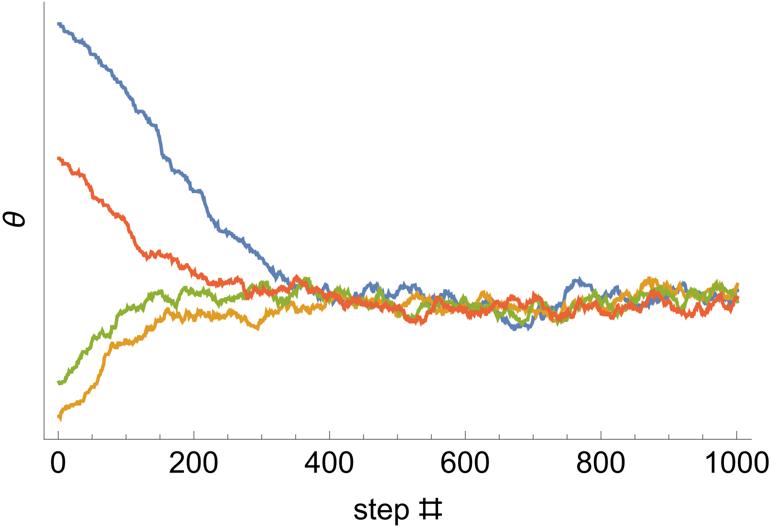
$$\hat{R} = \sqrt{\frac{W + \frac{1}{n}(B - W)}{W}} \tag{6}$$

- Where "within-chain" variability, $W = \frac{1}{m}\sum_{j=1}^{m} s_j^2$, for $m$ chains.
- And "between-chain" variability, $B = \frac{n}{m-1}\sum_{j=1}^{m}(\overline{\theta}_j - \overline{\theta})^2$.
- When we start $B >> W$ since we start in an overdispersed position.
- In convergence $B \to W \implies \hat{R} \to 1$ (in practice $\hat{R} < 1.1$ usually suffices).

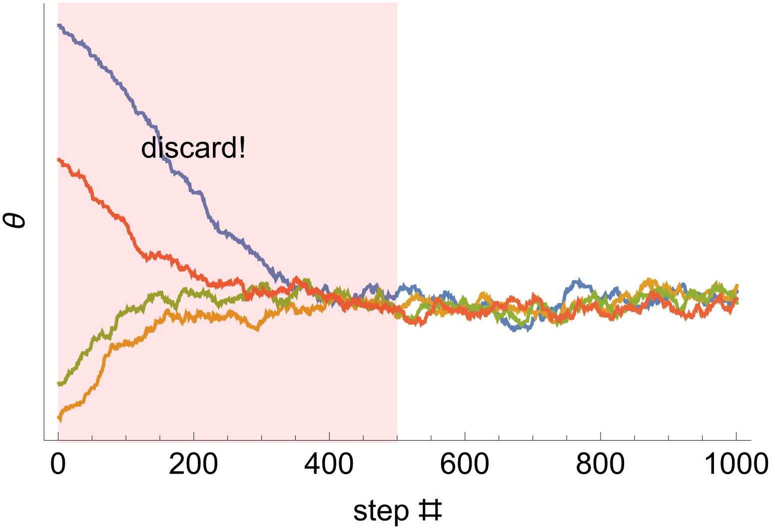
# Warm up period

- The initial proposal distribution is *not* the posterior.
- We therefore discard the beginning part of the chain called the "warm up" to lessen the effect of the starting position.
- Typically discard first half of converged chains (can also cut chains in two to monitor intra-chain convergence).

# Warm up period



# Warm up period

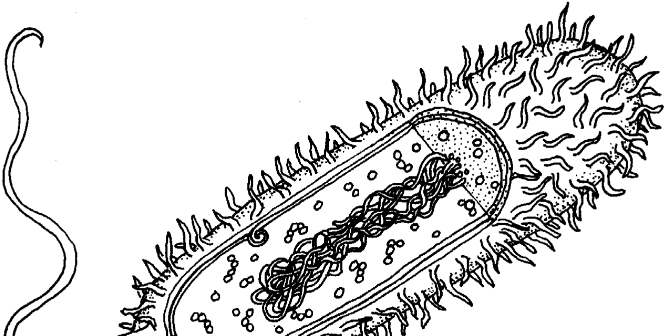


# Summary

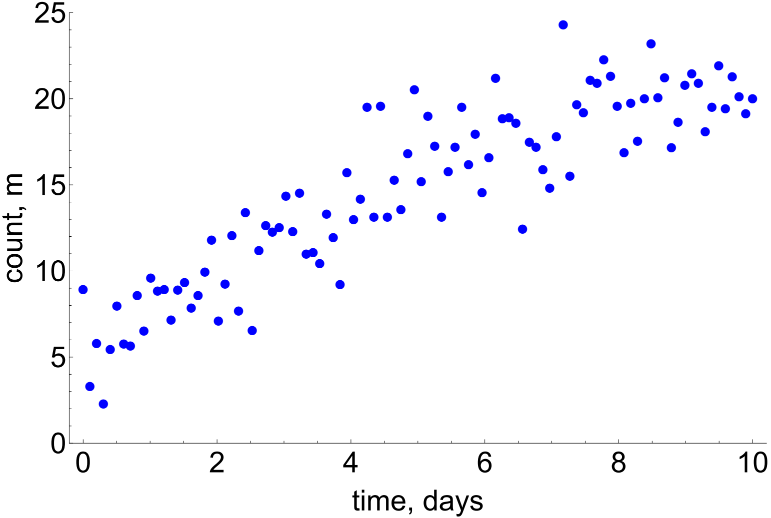
1. Sampling can be used to gain insight into a distribution.
2. Independent sampling from posterior not generally possible $\Longrightarrow$ shift to dependent sampling.
3. Random Walk Metropolis is a MCMC algorithm that allows *dependent* sampling from the posterior.
4. The efficiency of Metropolis depends on choosing the right step size.
5. Monitoring of sampler's convergence to the posterior is non-trivial.
6. The use of multiple chains makes it harder to make a mistake although not impossible.

# Example: bacterial growth

- We carry out experiments where we inoculate agar plates with bacteria at time 0.
- At pre-defined time intervals we count the number of bacteria on each plate, $N(t)$.
- Suppose we want to model bacterial population growth over time.



# Example: bacteria growth data



# Example: bacterial growth model

- Assume the following model for bacterial population growth:

$$\frac{\mathrm{d}N}{\mathrm{d}t} = \alpha N(1 - \beta N) \tag{7}$$

where $\alpha > 0$ is the rate of growth due to bacterial cell division, and $\beta > 0$ measures the reduction in growth rate due to "crowding".

**Question:** how should we infer the parameters of this model?

# Example: bacterial growth model

**Answer:** assume measurement error around true value:

$$N^*(t) \sim \text{normal}(N(t), \sigma) \tag{8}$$

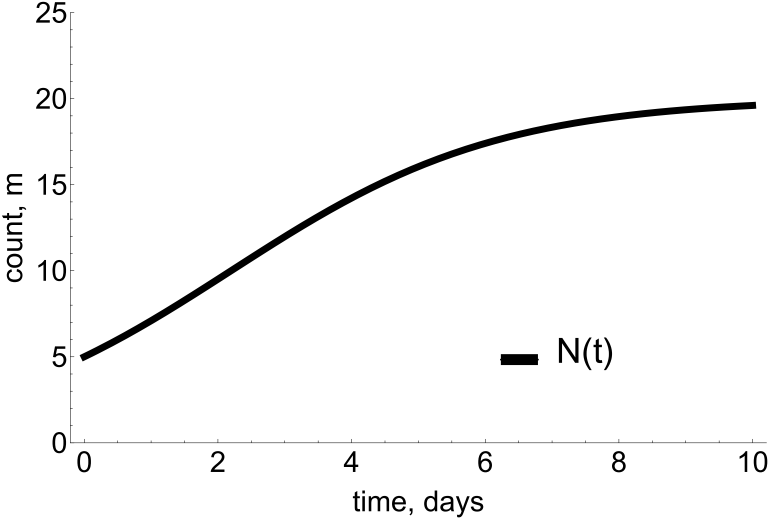
where

- $N^*(t)$ is the **measured** count of bacteria at time $t$.
- $N(t)$ is the solution to the ODE at time $t$ (true number of bacteria on plate).
- $\sigma > 0$ measures the magnitude of the measurement error about the true value.

**Question:** how does this model work?

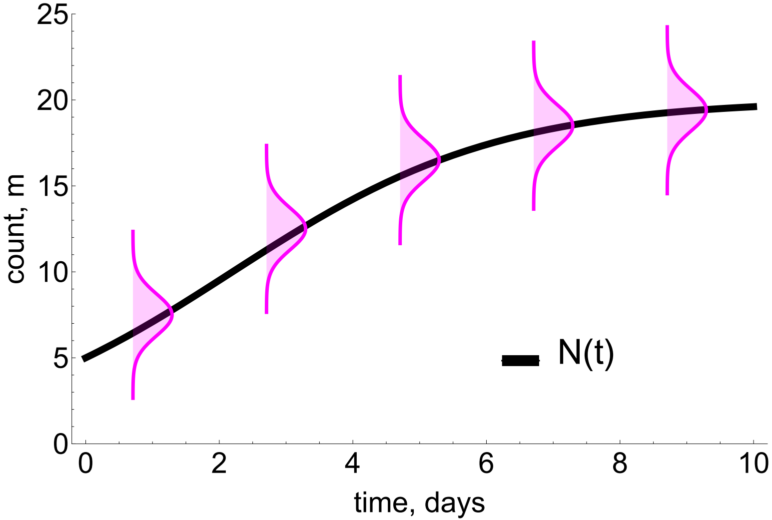
# Example: bacterial growth model

Start with true number of bacterial cells, $N(t)$.



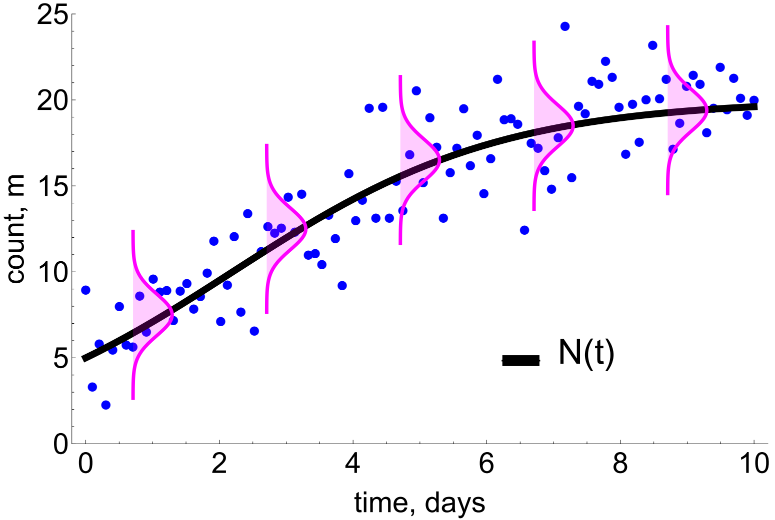
# Example: bacterial growth model

Overlay sampling distribution representing measurement error.



# Example: bacterial growth model

And data generated from this process.



# Example: bacteria growth model inference

Remember we are using a normal likelihood:

$$N^*(t) \sim \text{normal}(N(t), \sigma) \tag{9}$$

$\implies$ likelihood for all observations:

$$L(N(t), \sigma) = \prod_{t=t_1}^{T} \frac{1}{\sqrt{2\pi\sigma^2}} exp\left[\frac{-(N^*(t) - N(t))^2}{2\sigma^2}\right] \tag{10}$$

**Question:** how do we calculate $N(t)$?

# Example: bacteria growth model inference

$$\frac{\mathrm{d}N}{\mathrm{d}t} = \alpha N(1 - \beta N) \tag{11}$$

- In most ODE models, the mean $N(t)$ cannot be solved for exactly so we **can't write down a "closed-form" expression for the likelihood.**
- $\implies$ approximate answer using a numerical method.
- However any solution for $N(t)$ - exact or numerical - depends on the parameters of the ODE model. For our example:

$$N(t) = f(t, \alpha, \beta) \tag{12}$$

**Question:** how do we do MCMC in this setting?

# Example: bacteria growth model inference

For example, in Random Walk Metropolis:

- Start at random location in $(\alpha, \beta, \sigma)$ space.
- For t=1,...,T do:
  1. Propose a new location $(\alpha', \beta', \sigma')$ using a jumping distribution.
  2. Numerically (or analytically) integrate ODE to solve for $N(t, \alpha', \beta')$.
  3. Calculate un-normalised posterior at proposed location $\implies$ calculate $r$.
  4. Based on $r$ move to new location or stay at original.

$\implies$ at every step we must solve ODE for $N(t)$; can be computationally expensive!

# Issues with inference for ODEs and PDEs

- ODE models are very often non-identifiable $\implies$ need to reparameterise model.
- (Linked) ODE models can be slower to converge than simpler models $\implies$ need to run MCMC for longer before $\hat{R} < 1.1$ achieved.

$\implies$ important that we "know" our model well before we start to do inference explicitly.
Worth putting energy into mathematical analysis before trying MCMC.

# Inference for ODEs: summary

- ODE models are no harder to formulate than “traditional” problems.
- However for ODE models we cannot typically write down a “closed-form” expression for the likelihood.
- $\Longrightarrow$ use integrator to numerically solve for mean for each set of parameters.
- Posteriors for ODE models are often of a more complex geometry than regular models and are often unidentified.
- Check out: `https://github.com/pints-team/pints` for ODE inference.

# Intractable likelihood

A class of models have the property that the likelihood is too costly to evaluate exactly,

- Population genetics,
- Evolutionary biology,
- Epidemiology,
- Spatial models (e.g. cellular automata, cellular Potts, CHASTE),
- Models involving stochasticity.

However it may be (relatively) inexpensive to run a model for a given parameter set $\theta$.

# Example: stochastic Lotka-Volterra

Predator $Y$ and prey $X$,

$$X \xrightarrow{c_1} 2X \tag{13}$$

$$X + Y \xrightarrow{c_2} 2Y \tag{14}$$

$$Y \xrightarrow{c_3} \emptyset \tag{15}$$

$\Longrightarrow$ can simulate dynamics exactly using the Gillespie algorithm, but difficult to determine $p(\boldsymbol{X}, \boldsymbol{Y} | c_1, c_2, c_3)$.

# Basic ABC algorithm

**Question:** how can we infer $(c_1, c_2, c_3)$?

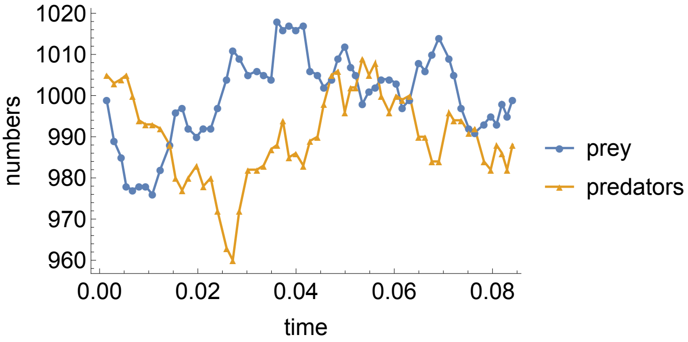
**Idea**: iterate the following,

1. Simulate from algorithm using $(c_1, c_2, c_3) \sim \pi(.)$, the prior.
2. If $||T(\boldsymbol{X}, \boldsymbol{Y})_{sim} - T(\boldsymbol{X}, \boldsymbol{Y})|| < \epsilon \implies$ accept parameters.
3. Else reject.

Where $T(.)$ is some type of informative summary statistic. For example, here we might choose the sum of squared errors.

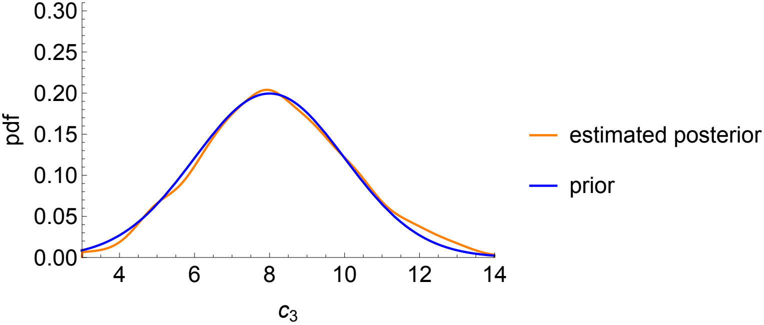
# Basic ABC algorithm: the role of $\epsilon$

**Test:** simulate data from stochastic Lotka-Volterra model where, $(c_1, c_2, c_3) = (10, 0.01, 10)$, and $(X_0, Y_0) = (1000, 1000)$ for $T = 0.1$.



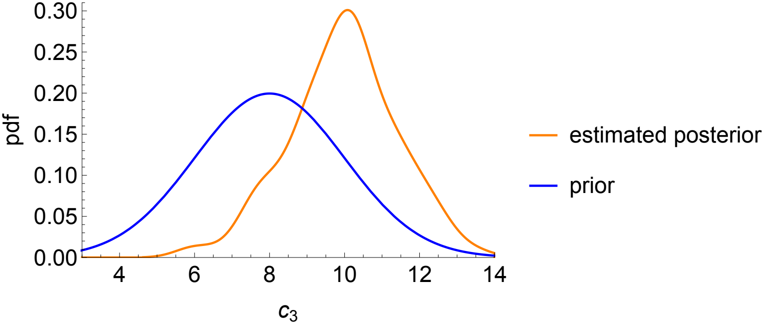

**Question:** how does choice of $\epsilon$ affect posterior?

# Basic ABC: high $\epsilon$




quick to run but $\implies$ approximate posterior same as priors!

# Basic ABC: low $\epsilon$




slow to run but $\implies$ approximate posterior near true posterior!

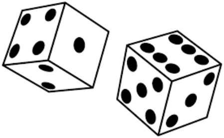
# ABC: summary

- ABC can be used to estimate approximate posteriors for some models where likelihood is intractable to calculate,
- so long as the time for a simulation is $\mathcal{O}$(seconds).
- Many variants of ABC exist, for example, Sequential/Particle MC, MCMC, regression $\implies$ generally help with rate of convergence to the posterior.
- Often useful (and less time intensive) step towards full Bayesian analysis of ODE and PDE models, and can help with questions around model identification.
- For slower simulations, either require large parallelism (e.g. ARCUS) or use more approximate methods (e.g. surrogate models).

# Not sure I understand?

Hierarchy of samplers:

**MC:**



**MCMC:**

# Not sure I understand?

**MC ?:**