

Basic MCMC algorithms and formulating the ODE inference problem

Ben Lambert¹

ben.c.lambert@gmail.com

¹University of Oxford

Monday 8th February, 2021

Day's activity

Morning:

- 9.30-10.30: Lecture: “Basic MCMC algorithms and formulating the ODE inference problem”.
- 10:30-12:30: Problem set: write your own MCMC algorithm and perform inference for the logistic growth model.

Afternoon:

- 13:50-14:50: Lecture: “ODE inference in practice”.
- 14:50-17:00: Problem set: use PINTS to perform inference for the Lotka-Volterra model.

Lecture outcomes

- ① Understand the mechanics of Random Walk Metropolis and how it works intuitively.
- ② Know that judging convergence of chains to the posterior is *hard*.
- ③ Learn how adaptive covariance MCMC can speed up sampling in most cases.
- ④ See how to formulate the inference problem for ODEs and PDEs.

- 1 Bayesian inference refresher
- 2 MCMC refresher
- 3 Judging convergence of chains to posterior
- 4 Adaptive covariance MCMC
- 5 Ordinary differential equations

Bayes' rule for inference

$$p(\theta|X) = \frac{p(X|\theta) \times p(\theta)}{p(X)} \quad (1)$$

Why do sampling in the first place?

To normalise the posterior, need:

$$p(X) = \int p(X|\theta) \times p(\theta) d\theta \quad (2)$$

where this really means:

$$p(X) = \int p(X|\theta_1, \theta_2, \dots, \theta_k) \times p(\theta_1, \theta_2, \dots, \theta_k) d\theta_1 d\theta_2 \dots d\theta_k \quad (3)$$

- 1 Bayesian inference refresher
- 2 MCMC refresher
- 3 Judging convergence of chains to posterior
- 4 Adaptive covariance MCMC
- 5 Ordinary differential equations

Motivation

Whilst we can't analytically calculate the posterior, we can still sample from it:

$$\theta_i \sim p(\theta|X). \quad (4)$$

If we have a large enough sample $(\theta_1, \theta_2, \dots, \theta_n)$ provides a good approximation to underlying distribution properties.

Defining Random Walk Metropolis

Has the following form:

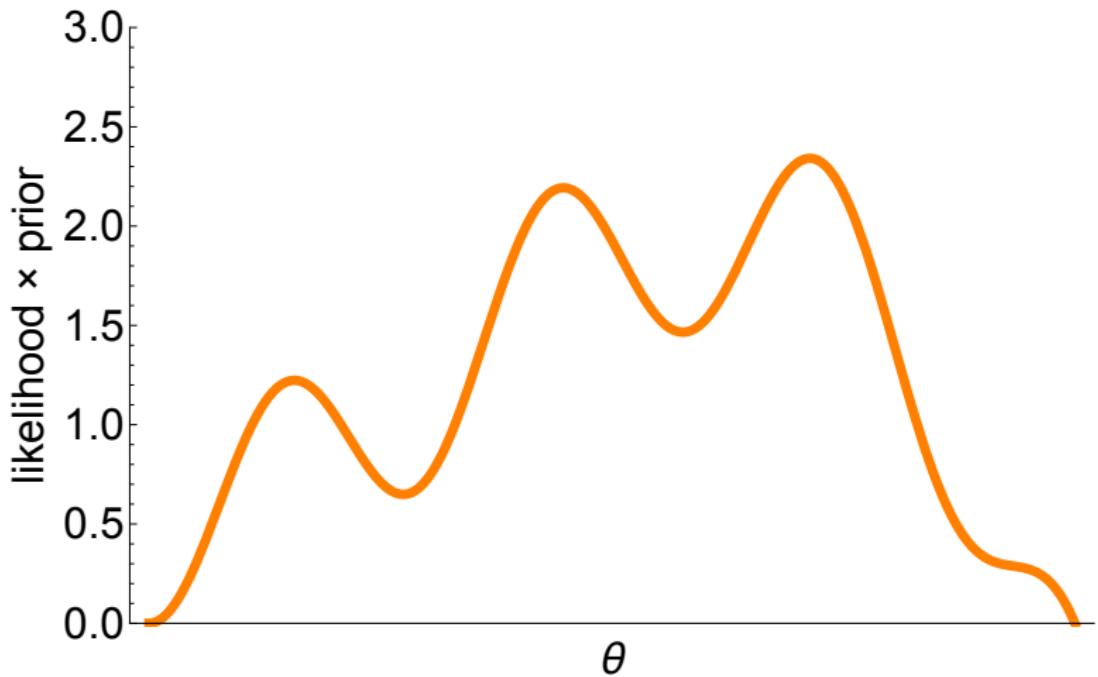
- Generate a random starting location θ_0 .
- Iterate the following for $t = 1, \dots, T$:
 - Propose a new location from a jumping distribution:
 $\theta_{t+1} \sim J(\theta_{t+1} | \theta_t)$.
 - Calculate the ratio:

$$r = \frac{\text{likelihood}(\theta_{t+1}) \times \text{prior}(\theta_{t+1})}{\text{likelihood}(\theta_t) \times \text{prior}(\theta_t)} \quad (5)$$

- 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.

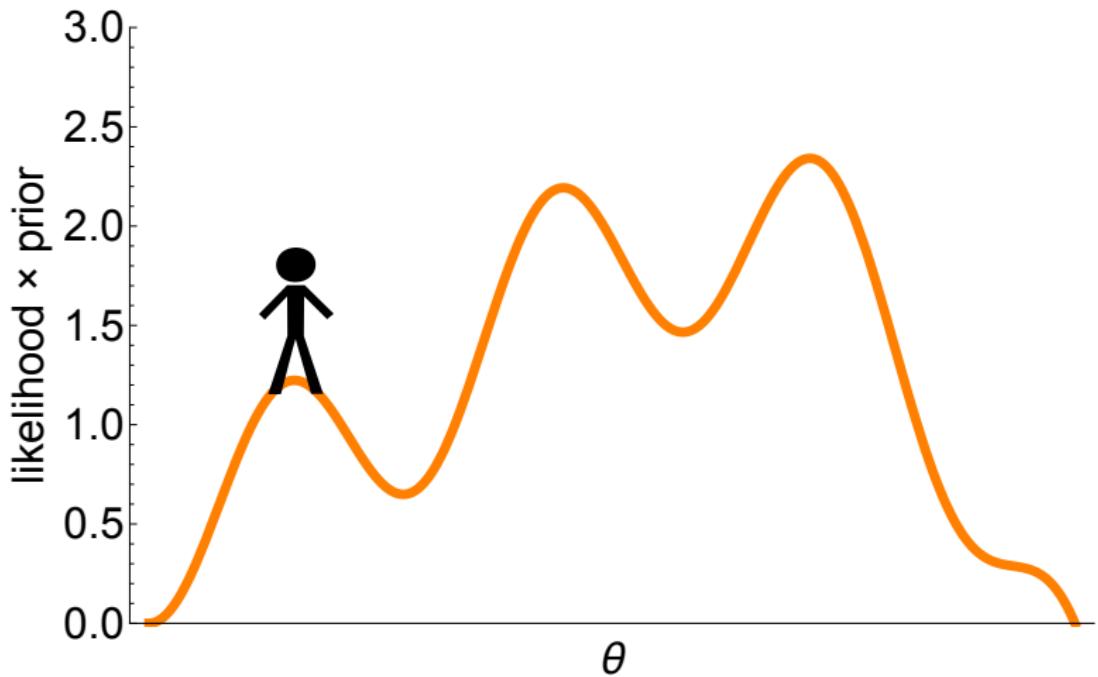
Defining Random Walk Metropolis

Start with the un-normalised density.



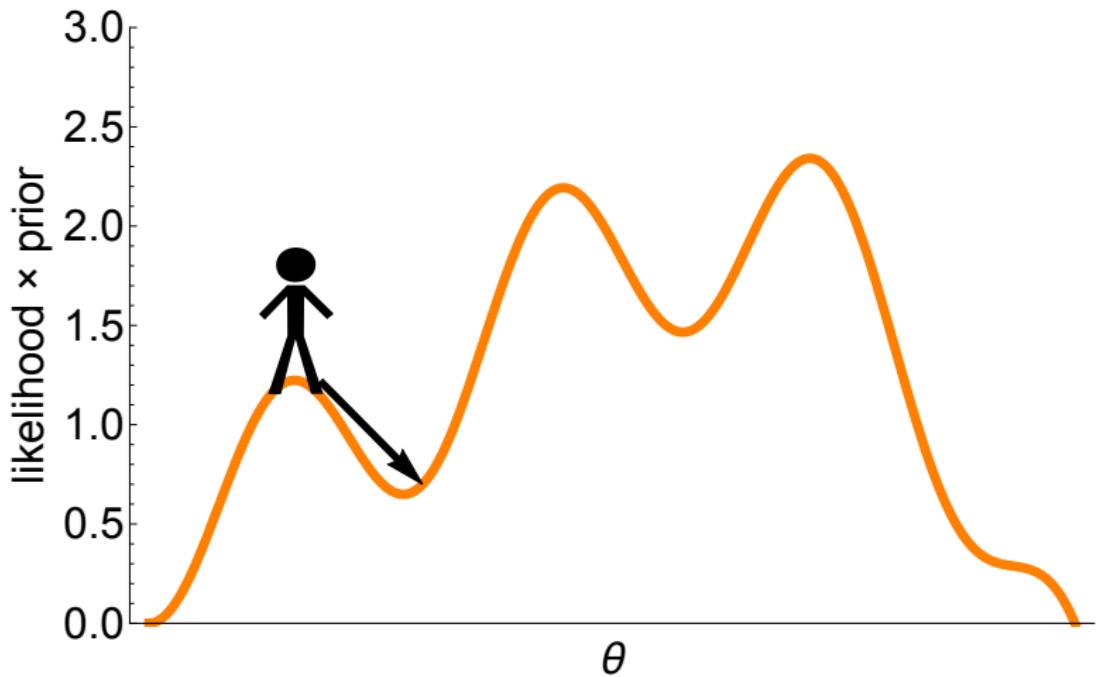
Defining Random Walk Metropolis

Select a random starting location.



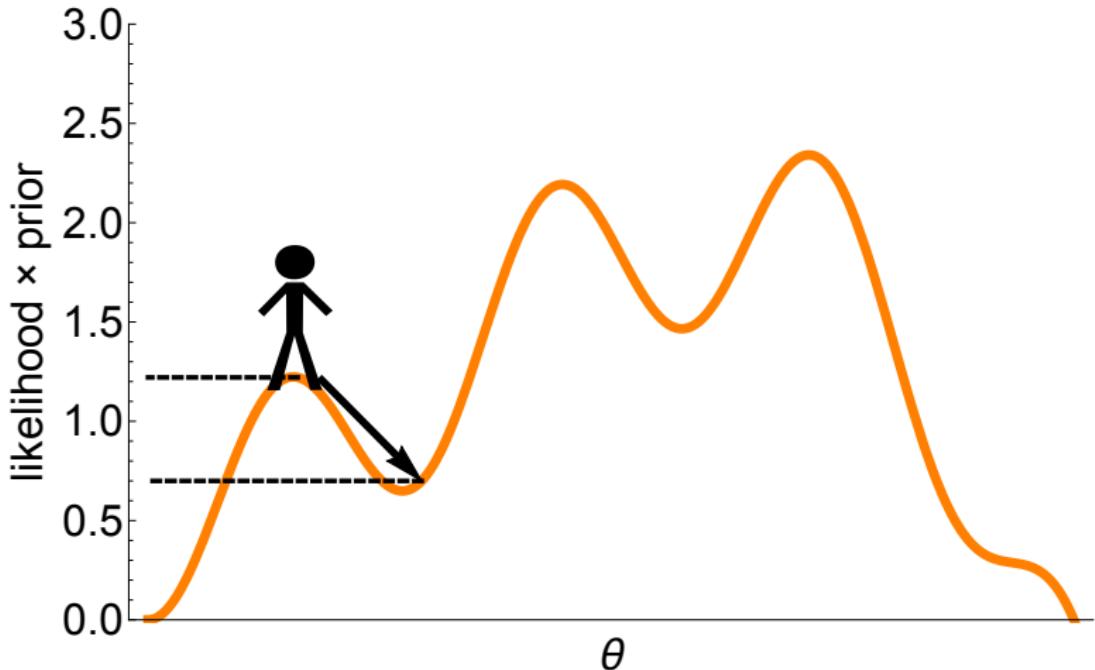
Defining Random Walk Metropolis

Propose a new location using jumping distribution.



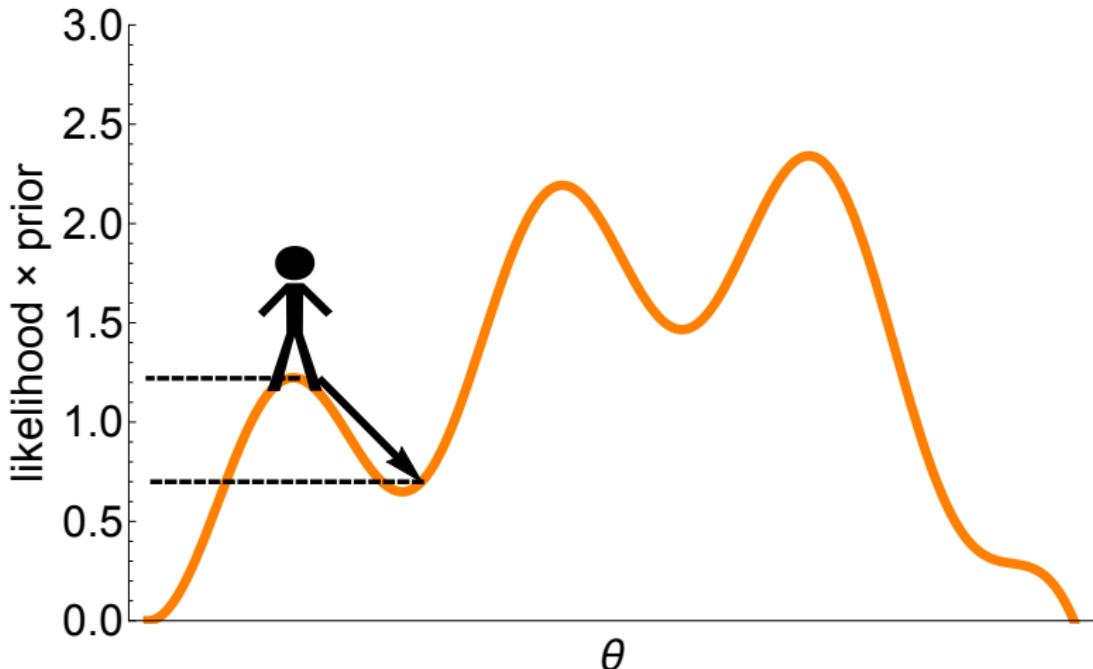
Defining Random Walk Metropolis

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



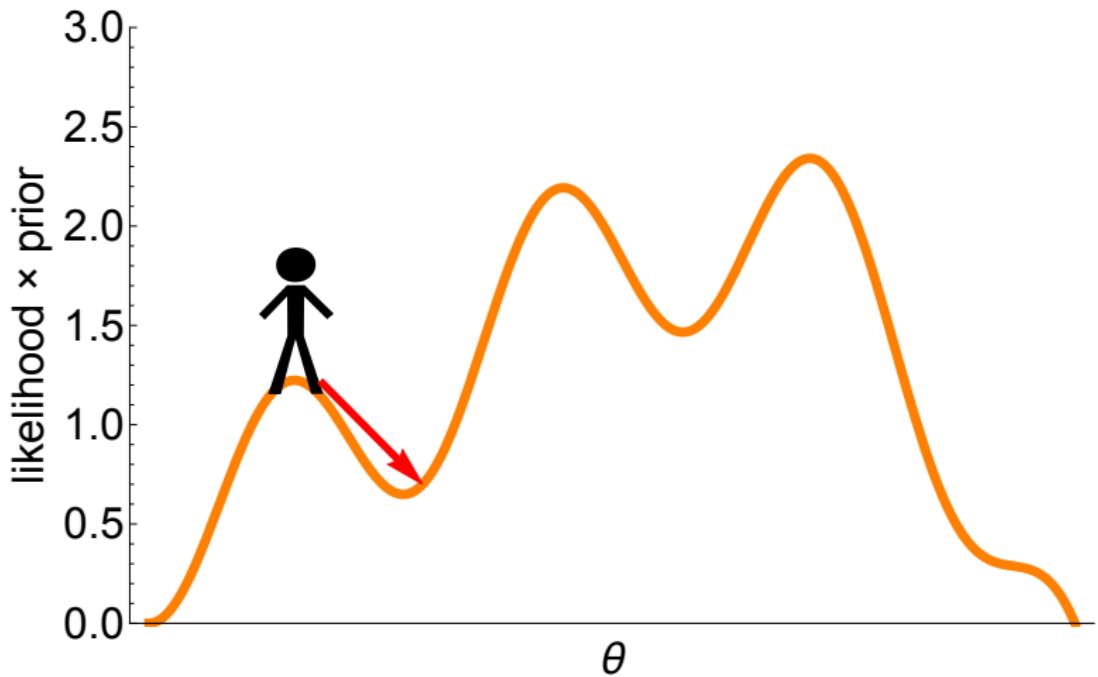
Defining Random Walk Metropolis

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



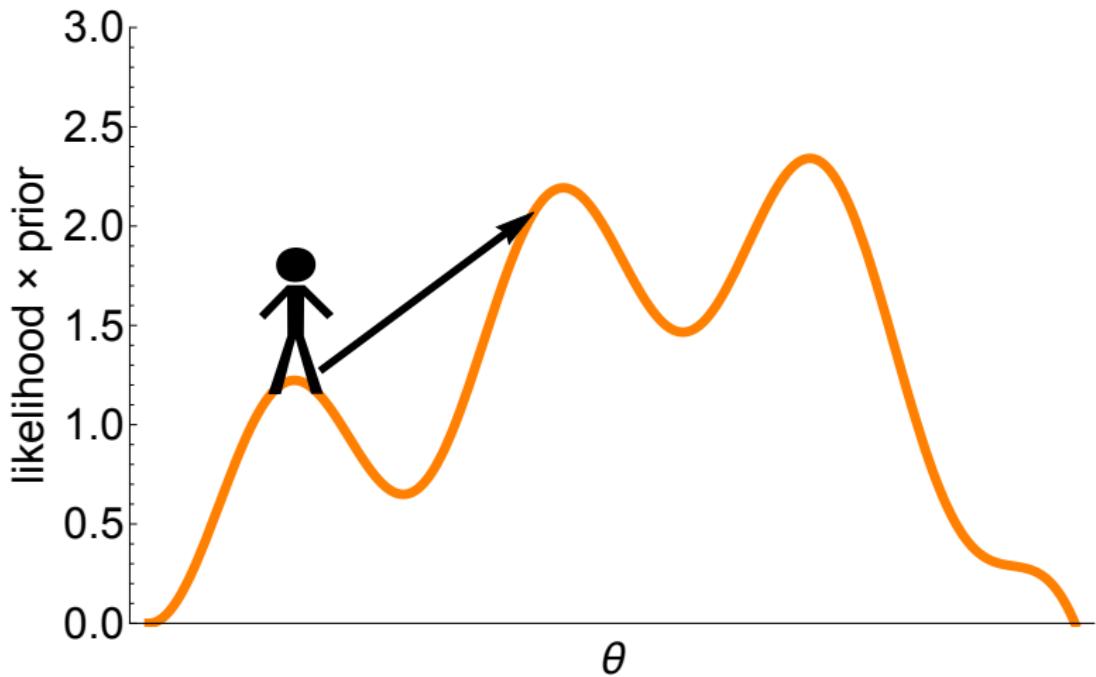
Defining Random Walk Metropolis

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



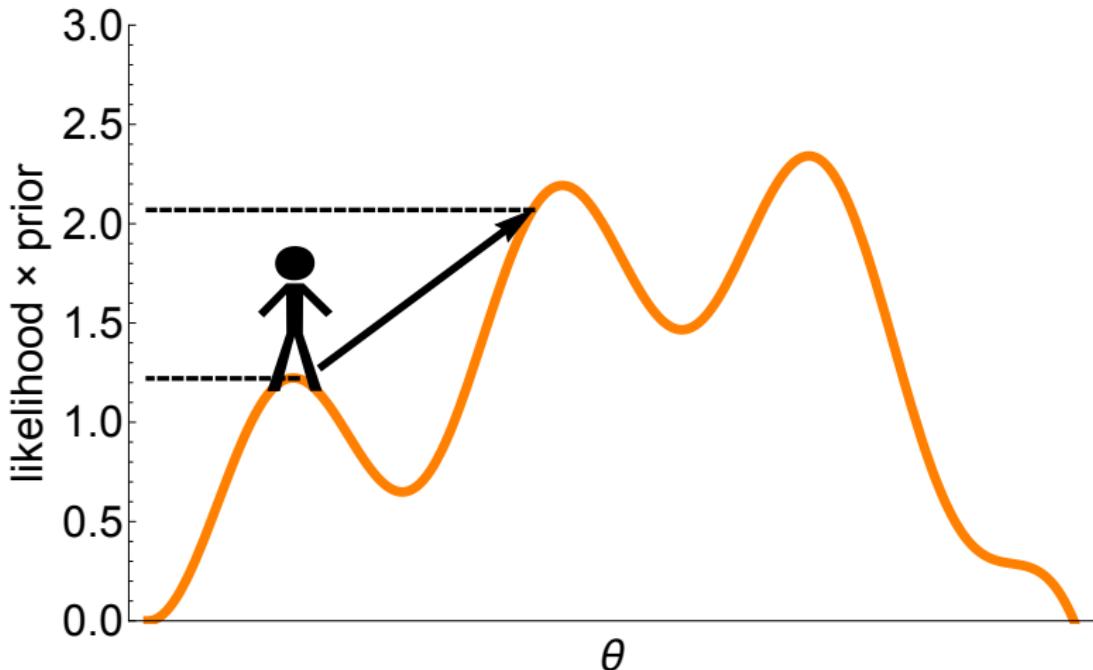
Defining Random Walk Metropolis

Generate a new proposed step using jumping distribution.



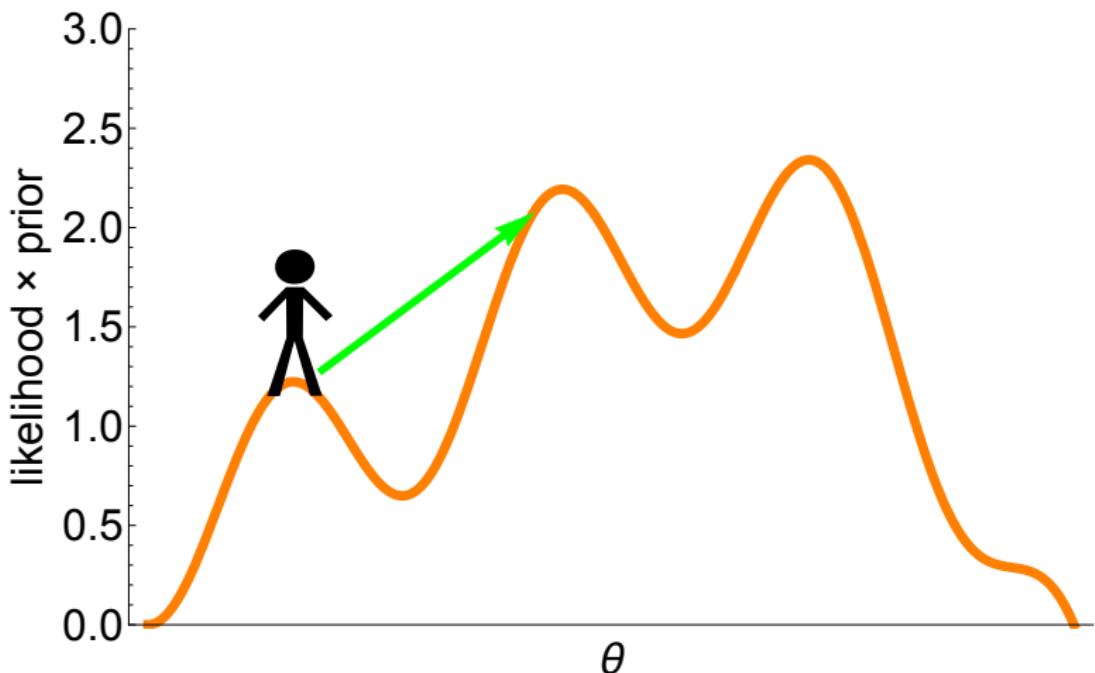
Defining Random Walk Metropolis

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



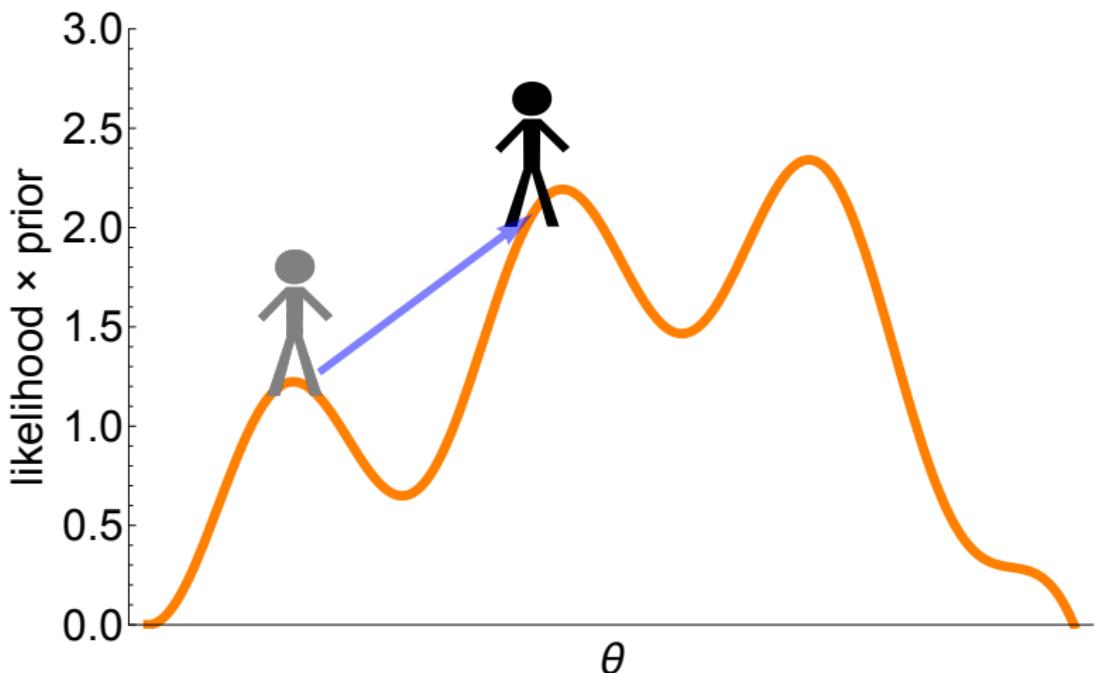
Defining Random Walk Metropolis

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



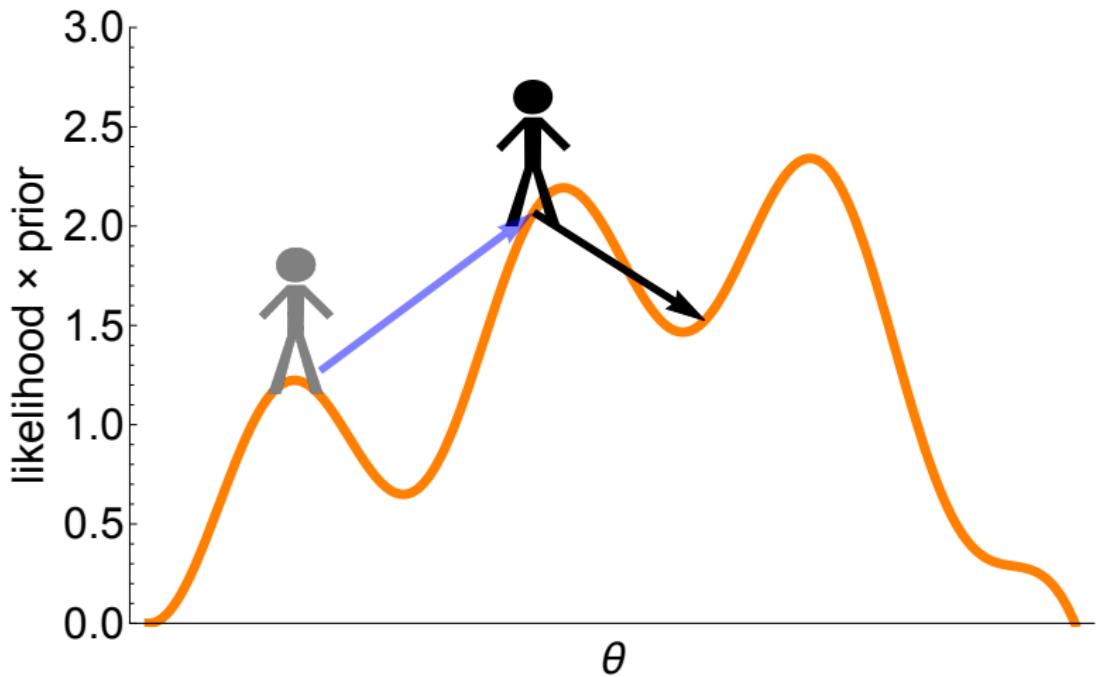
Defining Random Walk Metropolis

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



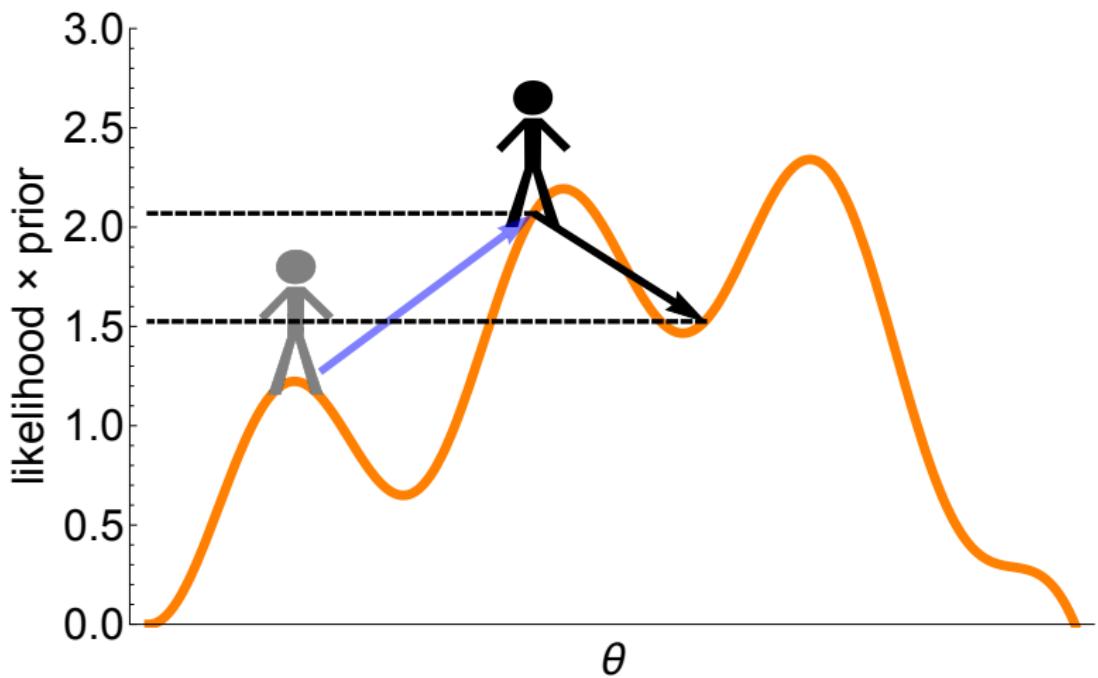
Defining Random Walk Metropolis

Propose a new step using jumping distribution.



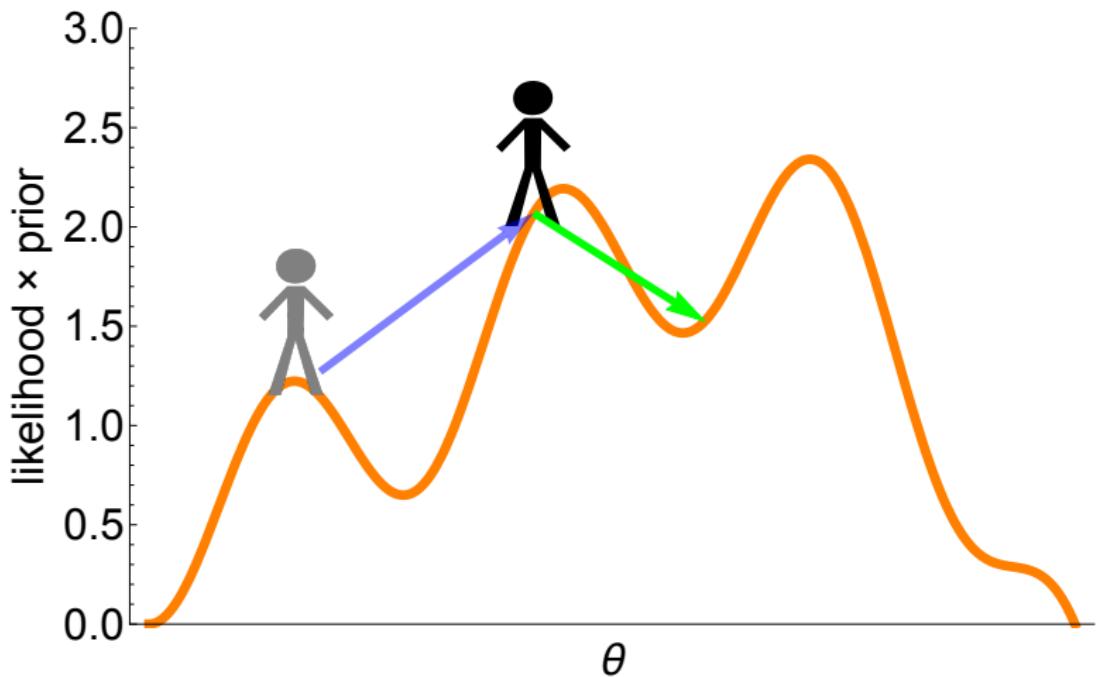
Defining Random Walk Metropolis

Calculate $r \approx 0.75$.



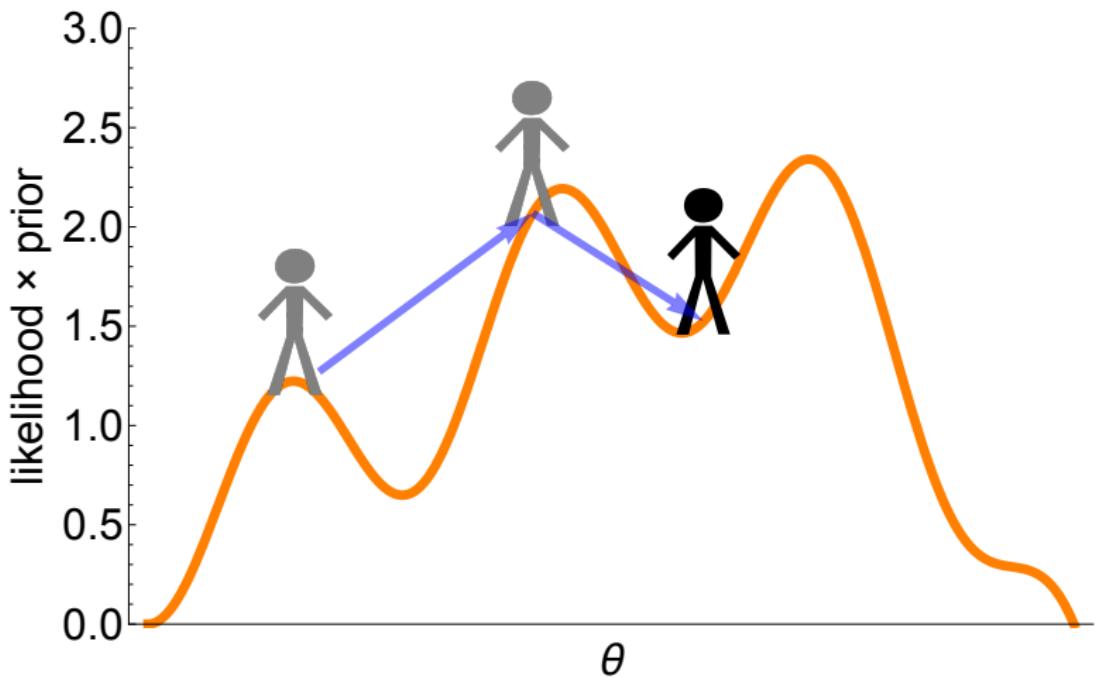
Defining Random Walk Metropolis

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



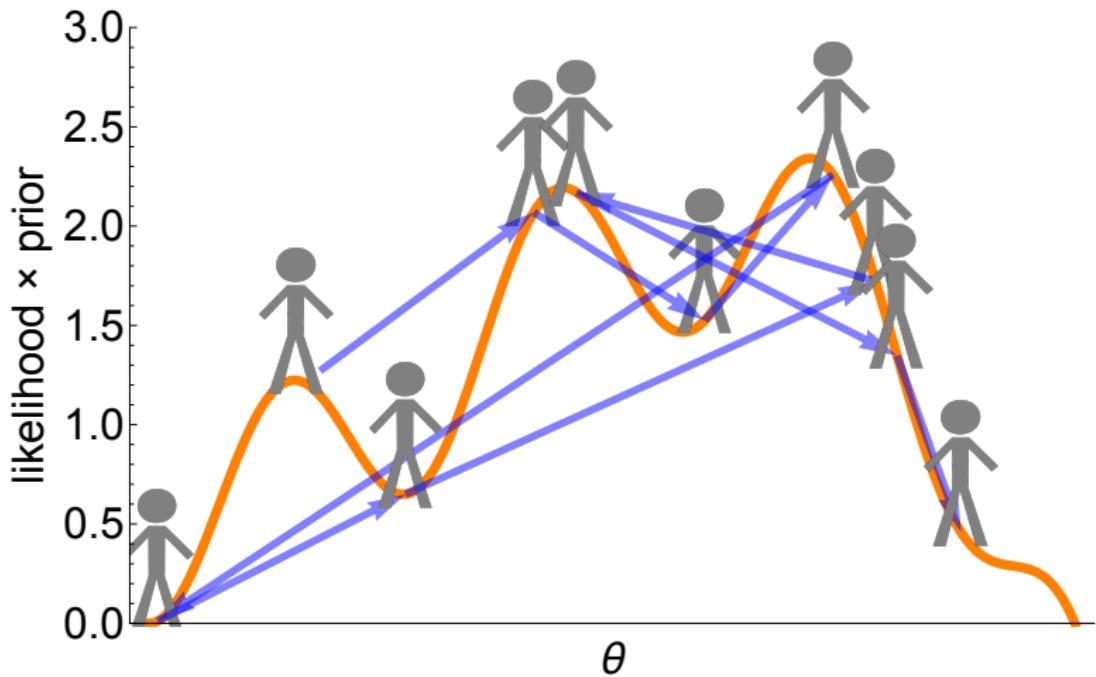
Defining Random Walk Metropolis

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



Defining Random Walk Metropolis

Repeat a large number of times.



Random Walk Metropolis: benefits

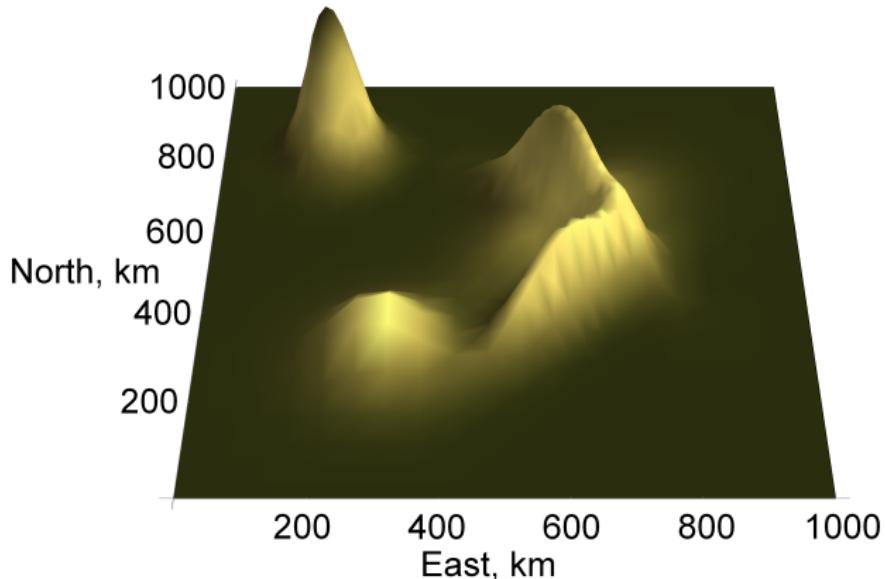
- Under quite general conditions the Random Walk Metropolis sampler converges **asymptotically** to the posterior.
- However for a sufficiently large sample size the sampling distribution may be practically indistinguishable from the true posterior.
- The algorithm requires us to be able to calculate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

$$r = \frac{\text{likelihood}(\theta_{t+1}) \times \text{prior}(\theta_{t+1})}{\text{likelihood}(\theta_t) \times \text{prior}(\theta_t)} \quad (6)$$

- 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

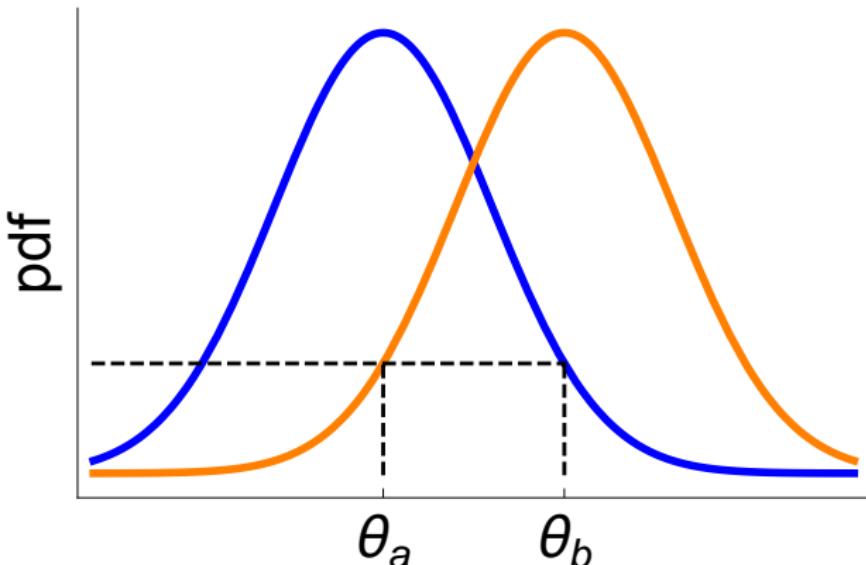
Random Walk Metropolis in action

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.

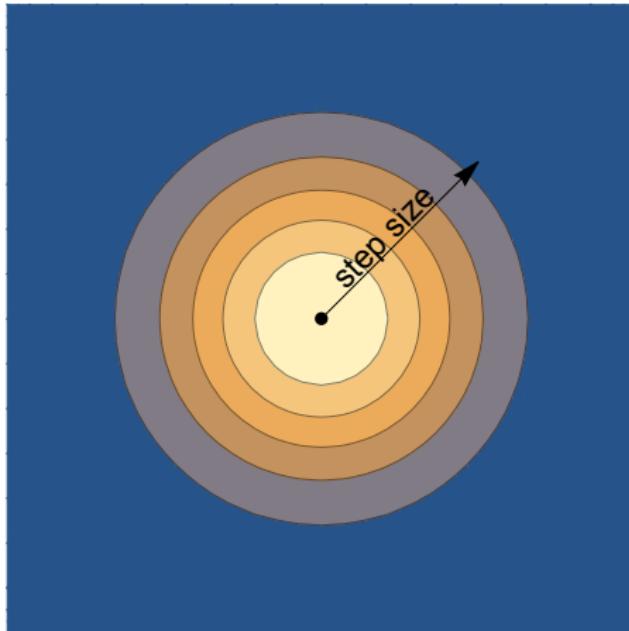
How do we choose the jumping distribution?

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



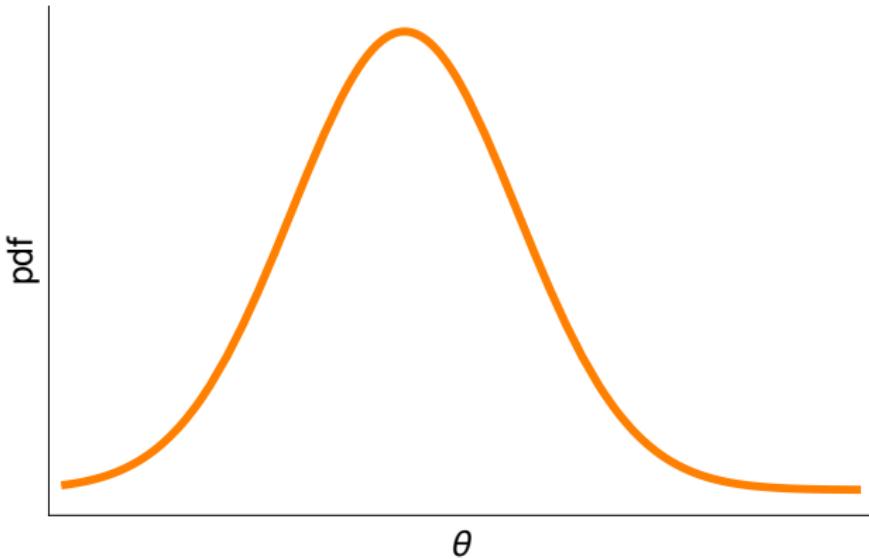
The importance of step size

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



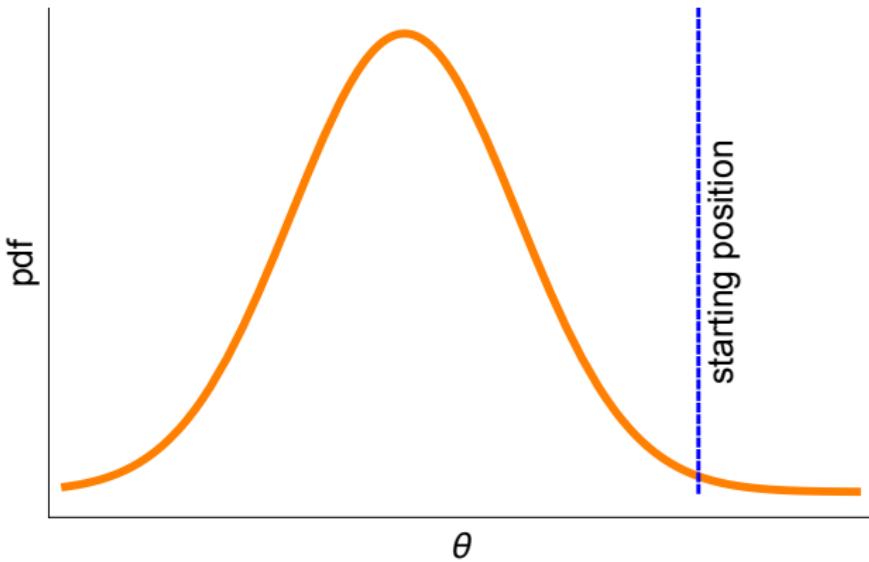
Another example posterior distribution

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.

- ➊ Bayesian inference refresher
- ➋ MCMC refresher
- ➌ Judging convergence of chains to posterior
- ➍ Adaptive covariance MCMC
- ➎ Ordinary differential equations

Why do we need to monitor convergence?

Recap the steps of Metropolis:

- ① Propose an initial position θ_0 using a initial proposal distribution $\pi(\theta) \neq p(\theta|X)$.
- ② For $t = 1, \dots, T$ do:
 - Propose a new location: $\theta_{t+1} \sim J(\theta_{t+1}|\theta_t)$.
 - Accept/reject move based on

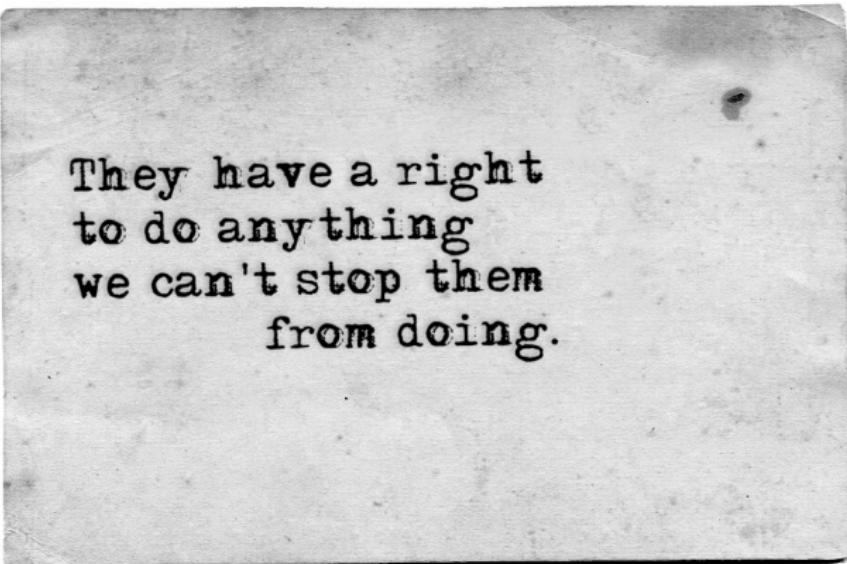
$$r = \frac{p(X|\theta_{t+1})p(\theta_{t+1})}{p(X|\theta_t)p(\theta_t)} > u \sim Unif(0, 1) \quad (7)$$

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 \implies need the posterior.
- But we don't have the posterior \Leftarrow the reason we are doing the sampling in the first place!



They have a right
to do anything
we can't stop them
from doing.

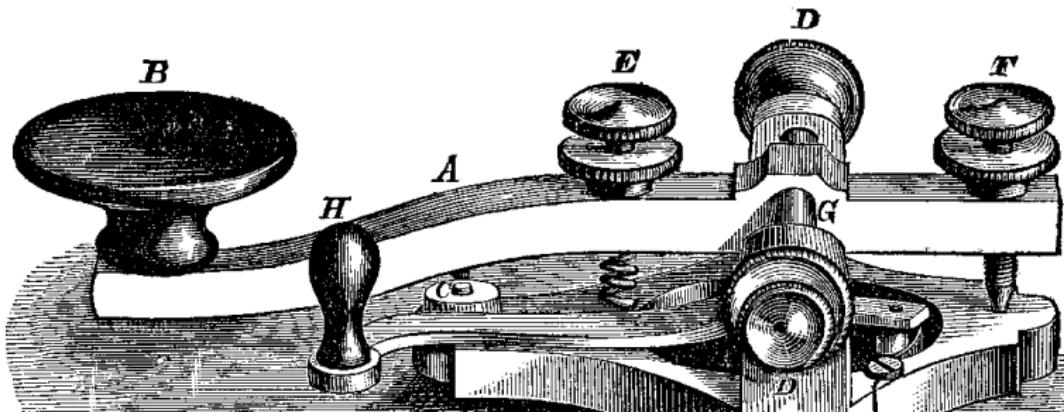
Two strategies for monitoring convergence

Strategy 1: measure distributional separation.

- For example Kullback-Leibler:

$$KL = \int p(\theta|X) \log \left(\frac{p(\theta|X)}{\pi(\theta_t)} \right) d\theta \quad (8)$$

- 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.

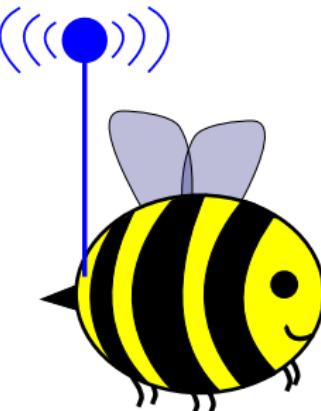
Monitoring convergence of a single chain

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 later

Convergence monitoring: single bee

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

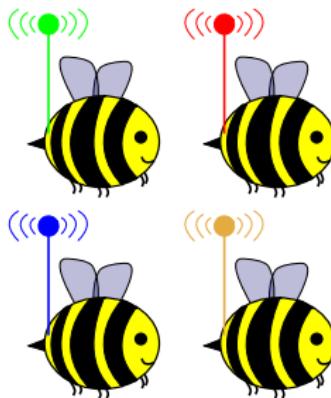
Convergence monitoring: single bee

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 θ 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 θ 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:

$$\hat{R} = \sqrt{\frac{W + \frac{1}{n}(B - W)}{W}} \quad (9)$$

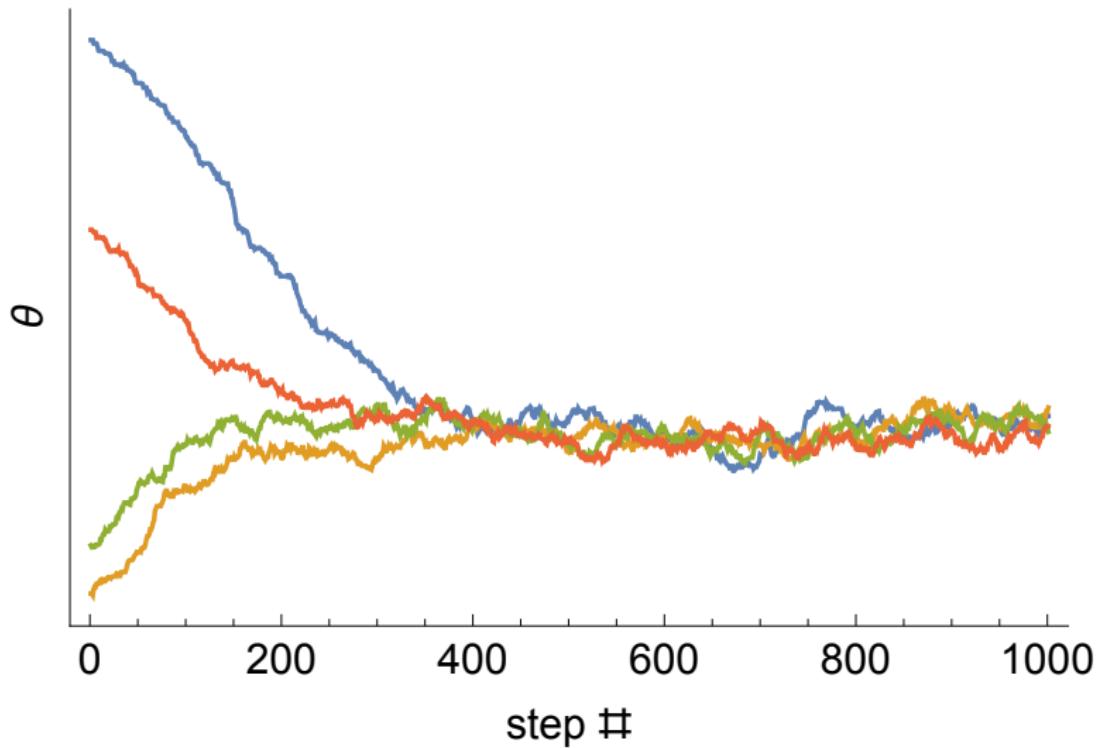
- Where “within-chain” variability, $W = \frac{1}{m} \sum_{j=1}^m s_j^2$, for m chains.
- And “between-chain” variability, $B = \frac{n}{m-1} \sum_{j=1}^m (\bar{\theta}_j - \bar{\theta})^2$.
- When we start $B \gg W$ since we start in an overdispersed position.
- In convergence $B \rightarrow W \implies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

Warm up period

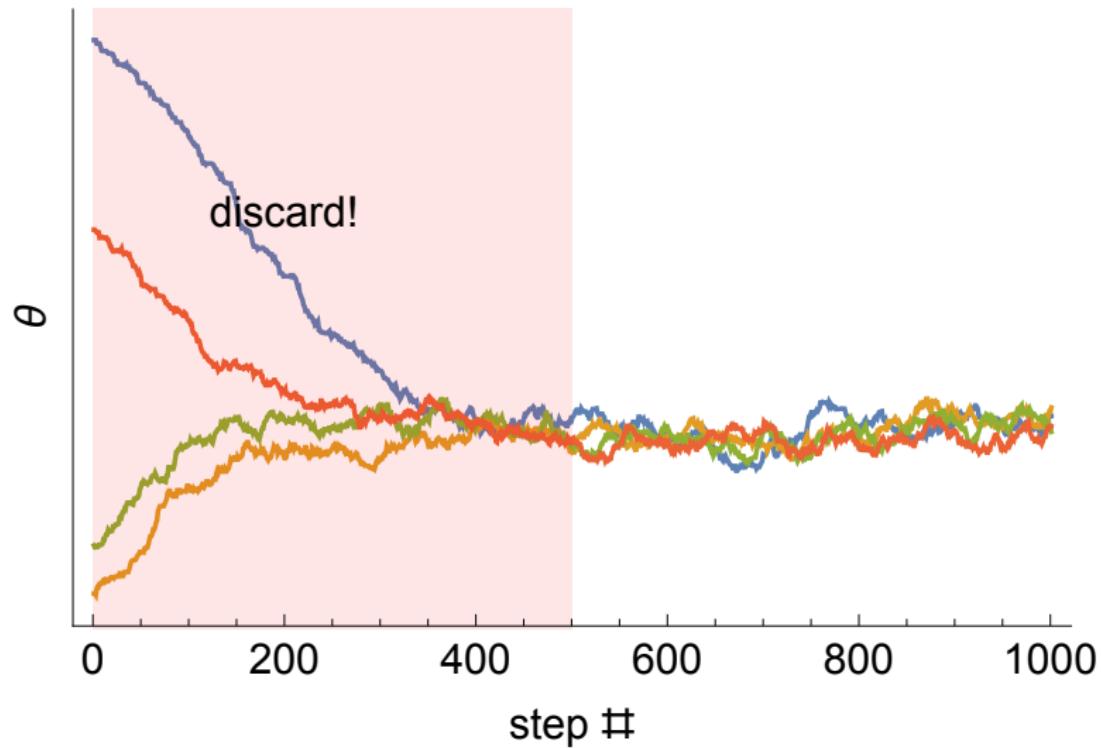
- The initial proposal distribution is *not* the posterior.
- We therefore discard the beginning part of the chain called the “warm up” to lessen the effect of the starting position.
- Typically discard first half of converged chains (can also cut chains in two to monitor intra-chain convergence).



Warm up period



Warm up period



- 1 Bayesian inference refresher
- 2 MCMC refresher
- 3 Judging convergence of chains to posterior
- 4 Adaptive covariance MCMC
- 5 Ordinary differential equations

Inefficient exploration of the typical set by Random Walk Metropolis



Adaptive covariance MCMC: adjusting the proposal kernel to posterior geometry

- The problem with RWM is that the proposals - being in random directions - are unlikely by chance to align with areas of high density.
- Adaptive covariance MCMC adjusts the proposal kernel dynamically to obtain a higher acceptance probability.

Sketch of adaptive covariance algorithm(s)

Begin by generating n samples using Random Walk Metropolis.
Start with $\Sigma_0 = \text{identity matrix}$. Then,

- ① Estimate sample mean: $\mu_t = \frac{1}{n} \sum_{i=1}^t \theta_i$.
- ② Estimate sample covariance matrix:
$$\Omega_t = (\theta^{\{t\}} - \mu_t)(\theta^{\{t\}} - \mu_t)'$$
.
- ③ Proposal kernel: $\Sigma_t = (1 - a^t)\Sigma_{t-1} + a^t\Omega_t$.

$\lim_{t \rightarrow \infty} a^t = 0$ is key to ensuring convergence to posterior distribution.

Adaptive covariance MCMC: adjusting the proposal kernel to posterior geometry

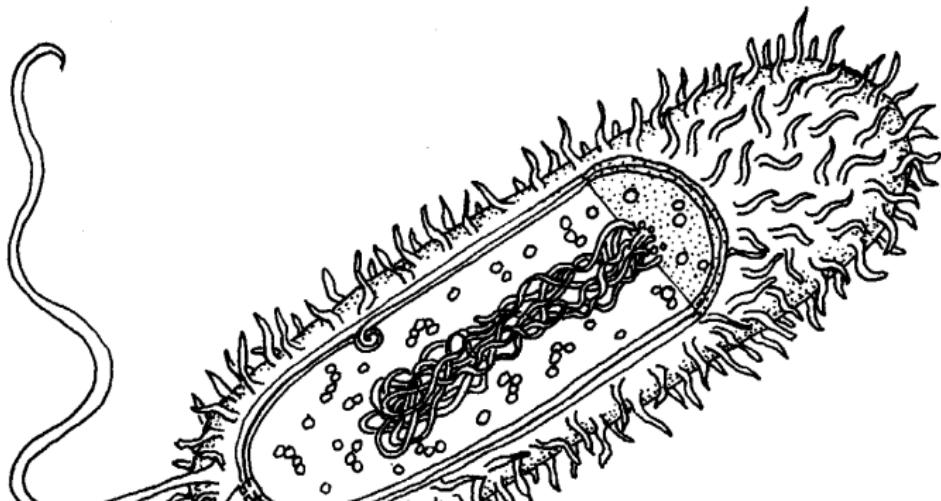
Adaptive covariance MCMC: summary

- RWM is inefficient due to random directionality of proposals.
- Adaptive covariance MCMC dynamically changes proposal kernel to match (global) posterior geometry leading to significant speed ups.
- ACMCMC can be used whenever RWM can be \Rightarrow very general algorithm; particularly useful for ODE and PDE models, where gradients of solution (necessary for HMC) are expensive.
- ACMCMC and loads of other algorithms are available in PINTS: <https://github.com/pints-team/pints>.
- Problem: adapting to global geometry often leads to very poor local exploration.

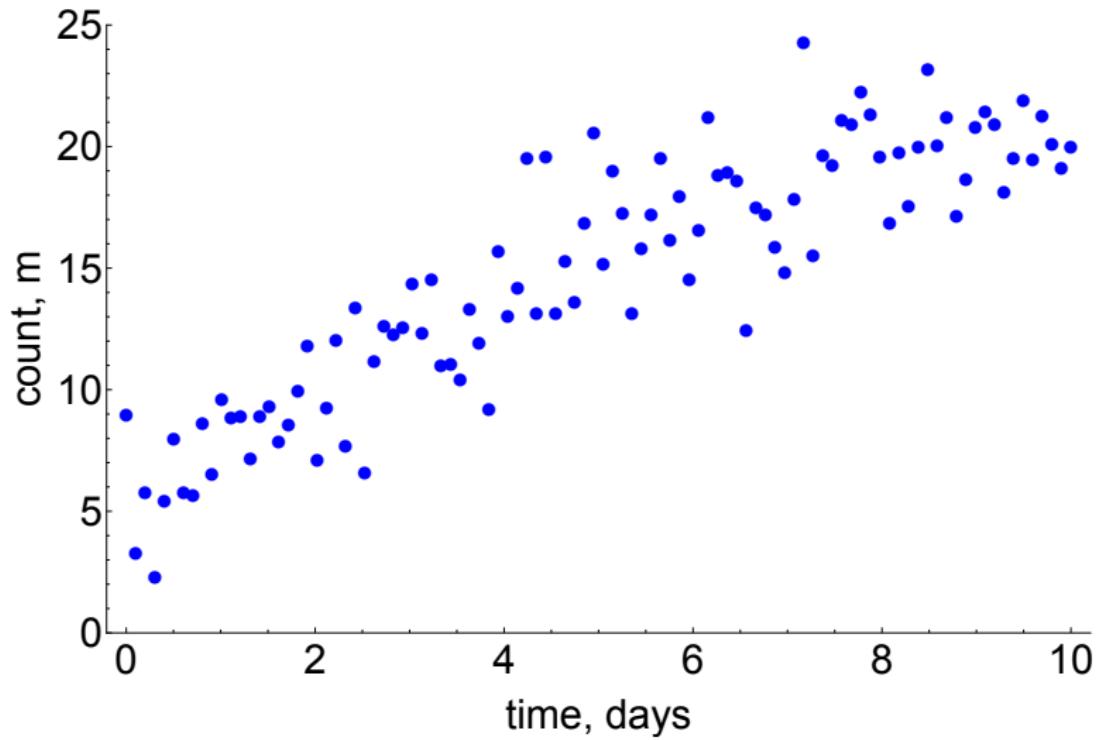
- ➊ Bayesian inference refresher
- ➋ MCMC refresher
- ➌ Judging convergence of chains to posterior
- ➍ Adaptive covariance MCMC
- ➎ Ordinary differential equations

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:

$$\frac{dN}{dt} = \alpha N(1 - \beta N) \quad (10)$$

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:

$$N^*(t) \sim \text{normal}(N(t), \sigma) \quad (11)$$

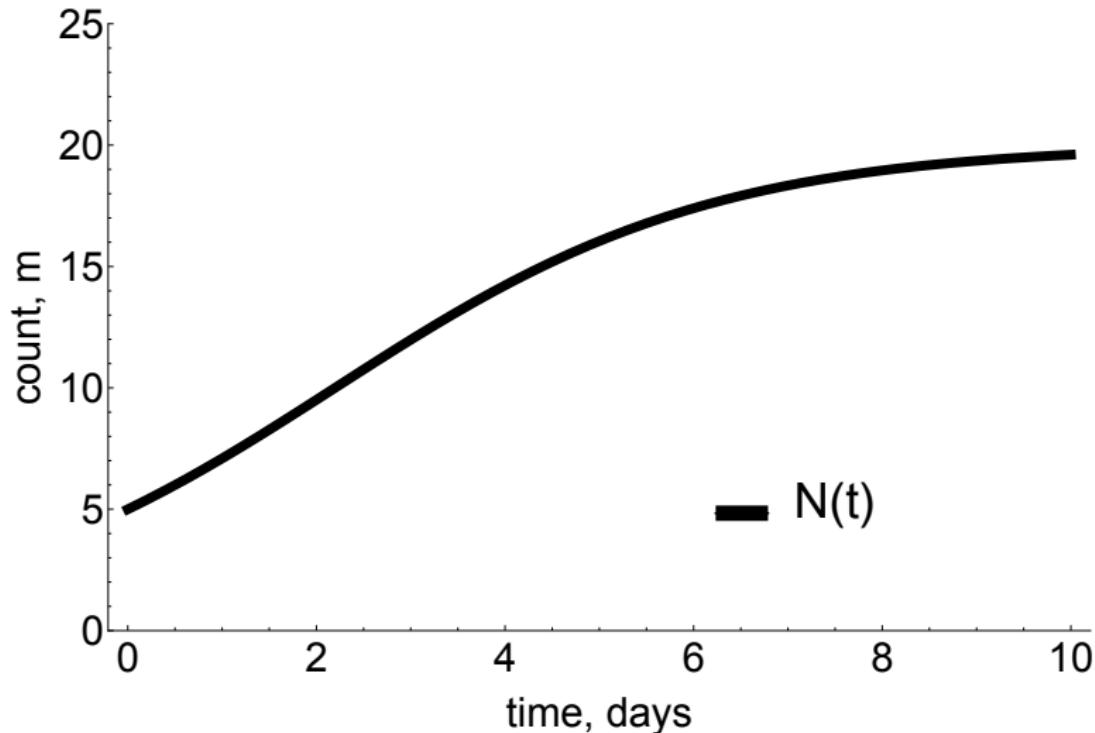
where

- $N^*(t)$ 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?

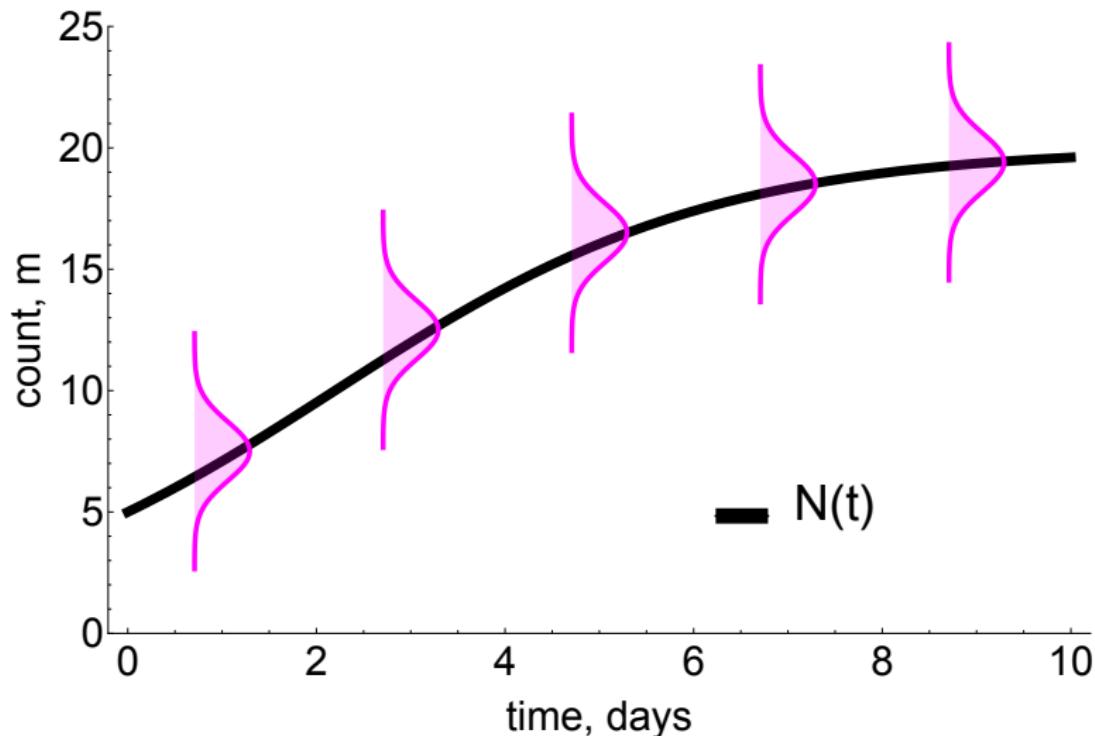
Example: bacterial growth model

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



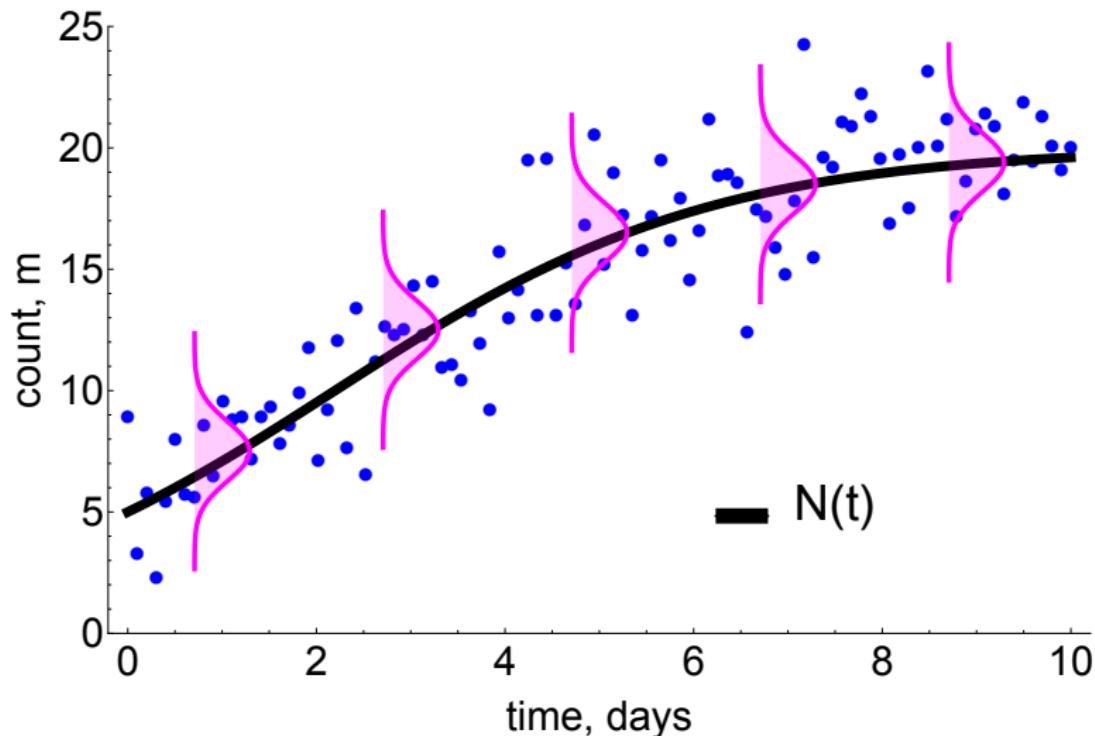
Example: bacterial growth model

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:

$$N^*(t) \sim \text{normal}(N(t), \sigma) \quad (12)$$

\implies likelihood for all observations:

$$L(N(t), \sigma) = \prod_{t=t_1}^T \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left[\frac{-(N^*(t) - N(t))^2}{2\sigma^2} \right] \quad (13)$$

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

Example: bacteria growth model inference

$$\frac{dN}{dt} = \alpha N(1 - \beta N) \quad (14)$$

- 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.**
- \Rightarrow 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:

$$N(t) = f(t, \alpha, \beta) \quad (15)$$

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 (α, β, σ) space.
- For $t=1, \dots, T$ 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
 \Rightarrow calculate r .
 - ④ Based on r move to new location or stay at original.
 \Rightarrow 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.
- \Rightarrow 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.