Rejection Sampling

Chapter 4: two fundamental sampling algorithms

Setting: we want samples from a target density $\pi(x)$, but cannot sample from it directly.

Chapter 4: two fundamental sampling algorithms

Setting: we want samples from a target density $\pi(x)$, but cannot sample from it directly.

Fact to utilize: We can often generate **Uniform**$(0,1)$ random numbers efficiently.

Chapter 4: two fundamental sampling algorithms

Setting: we want samples from a target density $\pi(x)$, but cannot sample from it directly.

Fact to utilize: We can often generate **Uniform**(0,1) random numbers efficiently.

Goal: Use Uniform$(0,1)$ samples to generate samples from more "exotic" distributions.

Chapter 4: two fundamental sampling algorithms

Setting: we want samples from a target density $\pi(x)$, but cannot sample from it directly.

Fact to utilize: We can often generate **Uniform**(0,1) random numbers efficiently.

Goal: Use Uniform$(0, 1)$ samples to generate samples from more "exotic" distributions.

Two methods:

1. Inverse transform sampling
2. Rejection sampling

Recap: Inverse transform sampling

Goal: sample $X \sim \pi$ when we can work with the CDF $F(x)$ and its inverse F^{-1}.

Recap: Inverse transform sampling

Suppose we want samples from a continuous distribution with CDF $F(\cdot)$.

Recap: Inverse transform sampling

Suppose we want samples from a continuous distribution with CDF $F(\cdot)$.

Idea:

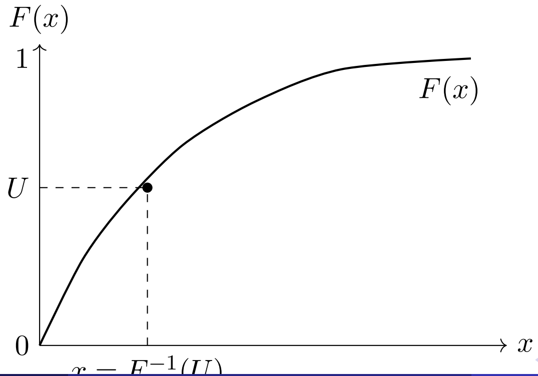
- draw $U \sim (0, 1)$ (a vertical coordinate on the CDF scale)
- map it to the corresponding x via $x = F^{-1}(U)$

Recap: Inverse transform sampling

Suppose we want samples from a continuous distribution with CDF $F(\cdot)$.

Idea:

- draw $U \sim (0, 1)$ (a vertical coordinate on the CDF scale)
- map it to the corresponding x via $x = F^{-1}(U)$



Inverse transform sampling

Given: continuous CDF F and quantile function F^{-1}.

Algorithm

Repeat:

1. Draw $U \sim \mathrm{Unif}(0,1)$.
2. Return $X = F^{-1}(U)$.

Guarantee: X has CDF F.

Where inverse transform sampling breaks down

Inverse transform sampling requires:

(i) $F(x)$ is tractable and (ii) $F^{-1}(u)$ is tractable.

Where inverse transform sampling breaks down

Inverse transform sampling requires:

(i) $F(x)$ is tractable and (ii) $F^{-1}(u)$ is tractable.

Many common families fail one (or both) of these in closed form:

- Normal distribution: F involves a non-elementary integral; F^{-1} not closed form
- Beta/Gamma: F^{-1} not closed form

So we need a second tool: rejection sampling.

Rejection sampling

Given: target density π, proposal density q, constant C such that $\pi(x) \leq Cq(x)$.

Algorithm

1. Repeat:
    1. Sample $X \sim q$.
    2. Sample $U \sim \mathrm{Unif}(0, 1)$.
    3. If $U \leq \pi(X)/(Cq(X))$, **accept** and output X.

Guarantee: accepted X has density π.

Rejection sampling: intuition

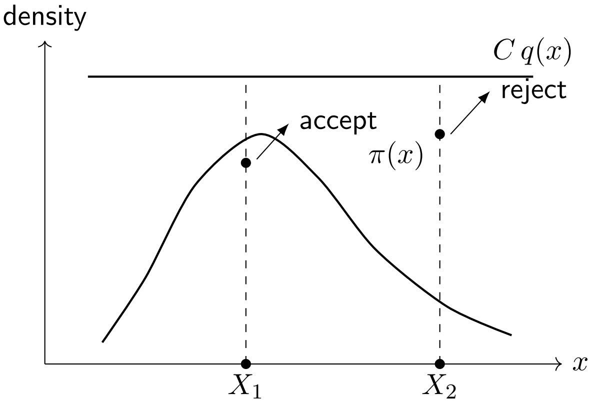
We choose a **proposal density** $q(x)$ that *we can* sample from, and a constant $C > 0$ such that

$$\frac{\pi(x)}{q(x)} \leq C \quad \text{for all } x.$$

Geometric picture:

- sample $X \sim q$
- generate a uniform "height" and accept if it falls under the target curve
- rejected samples are thrown away

Rejection sampling: the picture



Acceptance is more likely where $\pi(x)$ is large relative to the envelope.

Rejection sampling: formal condition

We want to sample from target density $\pi(x)$.

Assume we can sample from proposal density $q(x)$ and that there exists $C > 0$ such that

$$\frac{\pi(x)}{q(x)} \leq C \quad \text{for all } x \text{ where } q(x) > 0.$$

Equivalently:

$$\pi(x) \leq C\, q(x) \quad \text{for all } x.$$

Interpretation: $Cq(x)$ is an *envelope* that sits above $\pi(x)$ everywhere.

Rejection sampling: algorithm (step-by-step)

Inputs: target π, proposal q, constant C with $\pi(x) \leq Cq(x)$.

Repeat until you accept:

1. Sample $X \sim q(x)$.
2. Sample $U \sim (0, 1)$.
3. Compute acceptance probability
$$\alpha(X) = \frac{\pi(X)}{C\,q(X)}.$$
4. **Accept** X if $U \leq \alpha(X)$; otherwise **reject** and try again.

The accepted X is a valid sample from π.

Why the constant C matters

C scales the proposal: Cq must dominate π.

Bigger $C \Rightarrow$ easier to satisfy $\pi \leq Cq$ but:

$$\alpha(X) = \frac{\pi(X)}{Cq(X)} \text{ gets smaller } \Rightarrow \text{ more rejections.}$$

Smaller C (closer envelope) $\Rightarrow$ higher acceptance rate, but might fail the dominance condition.

Design principle: choose q and C so that Cq tightly "hugs" π.

Efficiency: three qualitative cases

Think of Cq as a shape over π.

- **Very inefficient:** envelope is far above π (large wasted area)
- **Reasonably efficient:** envelope just touches the maximum of π
- **Very efficient:** envelope closely matches π across its support

Rule of thumb: acceptance rate is roughly

$$\text{Acc} \approx \frac{\text{area under } \pi}{\text{area under } Cq}.$$

Since π is a density, $\int \pi(x)\,dx = 1$, so

$$\text{Acc} = \frac{1}{C} \quad \text{if } q \text{ is a normalized density and } C \text{ is valid.}$$

(We will formalize expected acceptance and expected trial counts later.)

Proof sketch that rejection sampling works

Let $X \sim q$ and $U \sim (0, 1)$ independent. We accept when

$$U \leq \frac{\pi(X)}{Cq(x)}.$$

Proof sketch that rejection sampling works

Let $X \sim q$ and $U \sim (0, 1)$ independent. We accept when

$$U \leq \frac{\pi(X)}{Cq(x)}.$$

Consider the density of an accepted draw at value x:

$$\mathbb{P}(X \in dx \text{ and accept}) = q(x)\,dx \cdot \mathbb{P}\left(U \leq \frac{\pi(x)}{Cq(x)}\right).$$

Proof sketch that rejection sampling works

Let $X \sim q$ and $U \sim (0, 1)$ independent. We accept when

$$U \leq \frac{\pi(X)}{Cq(x)}.$$

Consider the density of an accepted draw at value x:

$$\mathbb{P}(X \in dx \text{ and accept}) = q(x)\, dx \cdot \mathbb{P}\left(U \leq \frac{\pi(x)}{Cq(x)}\right).$$

Since $U \sim \mathrm{Unif}(0, 1)$,

$$\mathbb{P}\left(U \leq \frac{\pi(x)}{Cq(x)}\right) = \frac{\pi(x)}{Cq(x)}.$$

Proof sketch that rejection sampling works

Let $X \sim q$ and $U \sim (0,1)$ independent. We accept when

$$U \leq \frac{\pi(X)}{Cq(x)}.$$

Consider the density of an accepted draw at value x:

$$\mathbb{P}(X \in dx \text{ and accept}) = q(x)\,dx \cdot \mathbb{P}\left(U \leq \frac{\pi(x)}{Cq(x)}\right).$$

Since $U \sim \mathrm{Unif}(0,1)$,

$$\mathbb{P}\left(U \leq \frac{\pi(x)}{Cq(x)}\right) = \frac{\pi(x)}{Cq(x)}.$$

Therefore,

$$\mathbb{P}(X \in dx \text{ and accept}) = q(x)\,dx \cdot \frac{\pi(x)}{Cq(x)} = \frac{1}{C}\pi(x)\,dx.$$

Proof sketch that rejection sampling works

Let $X \sim q$ and $U \sim (0,1)$ independent. We accept when

$$U \leq \frac{\pi(X)}{Cq(x)}.$$

Consider the density of an accepted draw at value x:

$$\mathbb{P}(X \in dx \text{ and accept}) = q(x)\,dx \cdot \mathbb{P}\left(U \leq \frac{\pi(x)}{Cq(x)}\right).$$

Since $U \sim \mathrm{Unif}(0,1)$,

$$\mathbb{P}\left(U \leq \frac{\pi(x)}{Cq(x)}\right) = \frac{\pi(x)}{Cq(x)}.$$

Therefore,

$$\mathbb{P}(X \in dx \text{ and accept}) = q(x)\,dx \cdot \frac{\pi(x)}{Cq(x)} = \frac{1}{C}\pi(x)\,dx.$$

Conditioning on acceptance removes the factor $1/C$, so the accepted X has density $\pi(x)$.

Putting it together: what you should be able to do

After this lecture (and lab), you should be able to:

- explain why inverse transform fails for many common distributions
- state the rejection sampling condition $\pi(x) \leq Cq(x)$
- implement accept/reject logic and discuss efficiency qualitatively

Example 4.3

Let

$$\pi(y) = \frac{3}{4}\, y(2 - y)\, \mathbb{I}_{[0,2]}(y), \qquad \text{and } \pi(y) = 0 \text{ otherwise.}$$

Example 4.3

Let

$$\pi(y) = \frac{3}{4}\, y(2-y)\, \mathbb{I}_{[0,2]}(y), \qquad \text{and } \pi(y) = 0 \text{ otherwise.}$$

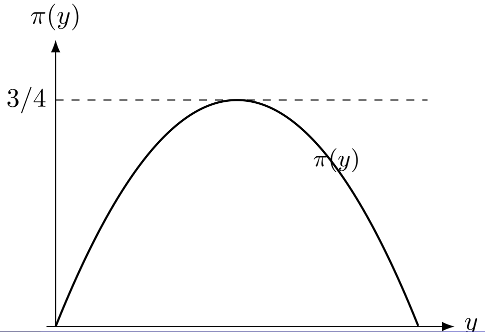
- On $[0, 2]$, $\pi(y) = \frac{3}{4}(2y - y^2)$ is an **upside-down quadratic**.
- Roots at $y = 0$ and $y = 2$.
- Maximum at $y = 1$ (by symmetry or differentiation): $\pi(1) = \frac{3}{4} \cdot 1 \cdot (2 - 1) = \frac{3}{4}$.

Example 4.3

Let

$$\pi(y) = \frac{3}{4}\, y(2-y)\, \mathbb{I}_{[0,2]}(y), \qquad \text{and } \pi(y) = 0 \text{ otherwise.}$$

- On $[0,2]$, $\pi(y) = \frac{3}{4}(2y - y^2)$ is an **upside-down quadratic**.
- Roots at $y = 0$ and $y = 2$.
- Maximum at $y = 1$ (by symmetry or differentiation): $\pi(1) = \frac{3}{4} \cdot 1 \cdot (2-1) = \frac{3}{4}$.



Choosing a proposal and the envelope constant

A natural choice: **uniform** proposal on the correct support.

Choosing a proposal and the envelope constant

A natural choice: **uniform** proposal on the correct support.
Let

$$q(y) = \frac{1}{2}\,\mathbb{I}_{[0,2]}(y) \quad \Longleftrightarrow \quad Y \sim \mathrm{Unif}(0,2).$$

Choosing a proposal and the envelope constant

A natural choice: **uniform** proposal on the correct support.
Let

$$q(y) = \frac{1}{2}\,\mathbb{I}_{[0,2]}(y) \quad \Longleftrightarrow \quad Y \sim \mathrm{Unif}(0,2).$$

We need a constant c such that

$$\pi(y) \le c\,q(y) \quad \text{for all } y.$$

Choosing a proposal and the envelope constant

A natural choice: **uniform** proposal on the correct support.
Let

$$q(y) = \frac{1}{2}\,\mathbb{I}_{[0,2]}(y) \quad\iff\quad Y \sim \mathrm{Unif}(0,2).$$

We need a constant c such that

$$\pi(y) \le c\,q(y) \quad \text{for all } y.$$

Since $\max_{y\in[0,2]} \pi(y) = 3/4$ and $q(y) = 1/2$ on $[0,2]$,

$$c \;\ge\; \frac{\max \pi(y)}{q(y)} = \frac{\frac{3}{4}}{\frac{1}{2}} = \frac{3}{2}.$$

So we can take $\boxed{\text{c=3/2}}$ (tightest envelope, hence most efficient).

Choosing a proposal and the envelope constant

A natural choice: **uniform** proposal on the correct support.
Let

$$q(y) = \frac{1}{2}\,\mathbb{I}_{[0,2]}(y) \quad \Longleftrightarrow \quad Y \sim \mathrm{Unif}(0,2).$$

We need a constant c such that

$$\pi(y) \leq c\,q(y) \quad \text{for all } y.$$

Since $\max_{y\in[0,2]} \pi(y) = 3/4$ and $q(y) = 1/2$ on $[0,2]$,

$$c \;\geq\; \frac{\max \pi(y)}{q(y)} = \frac{\frac{3}{4}}{\frac{1}{2}} = \frac{3}{2}.$$

So we can take $\boxed{\text{c=3/2}}$ (tightest envelope, hence most efficient).
Acceptance probability:

$$\mathbb{P}(\text{accept}) = \mathbb{E}\left[\frac{\pi(Y)}{c\,q(Y)}\right] = \frac{1}{c} = \frac{2}{3}.$$

Common pitfall (and the fix): sample from the right interval

A very common mistake is to sample $Y \sim \mathrm{Unif}(0,1)$ when the target lives on $[0,2]$.

Common pitfall (and the fix): sample from the right interval

A very common mistake is to sample $Y \sim \mathrm{Unif}(0, 1)$ when the target lives on $[0, 2]$.

- If you only propose on $[0, 1]$, you **miss half the support**.
- Your accepted samples cannot possibly follow $\pi(y)$ on $[0, 2]$.

Common pitfall (and the fix): sample from the right interval

A very common mistake is to sample $Y \sim \mathrm{Unif}(0,1)$ when the target lives on $[0,2]$.

- If you only propose on $[0,1]$, you **miss half the support**.
- Your accepted samples cannot possibly follow $\pi(y)$ on $[0,2]$.

Correct first step is:

$$Y \sim \mathrm{Unif}(0,2) \quad \text{(not } \mathrm{Unif}(0,1)\text{)}.$$

Then apply the acceptance test with $c = 3/2$ and $q(y) = 1/2$.

Rejection sampling algorithm for this example

With $q(y) = \frac{1}{2}\mathbb{I}_{[0,2]}(y)$ and $c = \frac{3}{2}$:

Rejection sampling algorithm for this example

With $q(y) = \frac{1}{2}\mathbb{I}_{[0,2]}(y)$ and $c = \frac{3}{2}$:

1. Sample $Y \sim \mathrm{Unif}(0, 2)$.
2. Sample $U \sim \mathrm{Unif}(0, 1)$ independently.
3. Accept Y if

$$U \leq \frac{\pi(Y)}{c\,q(Y)} = \frac{\frac{3}{4}Y(2-Y)}{\left(\frac{3}{2}\right)\left(\frac{1}{2}\right)} = Y(2-Y).$$

Otherwise reject and repeat.

Rejection sampling algorithm for this example

With $q(y) = \frac{1}{2}\mathbb{I}_{[0,2]}(y)$ and $c = \frac{3}{2}$:

1. Sample $Y \sim \mathrm{Unif}(0, 2)$.
2. Sample $U \sim \mathrm{Unif}(0, 1)$ independently.
3. Accept Y if

$$U \leq \frac{\pi(Y)}{c\,q(Y)} = \frac{\frac{3}{4}Y(2-Y)}{\left(\frac{3}{2}\right)\left(\frac{1}{2}\right)} = Y(2-Y).$$

   Otherwise reject and repeat.

So the acceptance test becomes especially clean:

$$\boxed{\text{Accept if } U \leq Y(2-Y).}$$

R implementation (accept–reject)

```
# Target: pi(y) = (3/4) * y * (2-y) on [0,2]
# Proposal: Y ~ Unif(0,2) => q(y)=1/2 on [0,2]
# Tight envelope: c = 3/2
# Acceptance test simplifies to: U <= Y*(2-Y)

rsample_pi <- function(n){
  out <- numeric(n)
  i <- 1
  while(i <= n){
    y <- runif(1, min = 0, max = 2) # propose
    u <- runif(1) # acceptance uniform
    if(u <= y*(2 - y)){
      out[i] <- y
      i <- i + 1
    }
  }
  return(out)
}
```

R: visualising accept vs reject (one iteration sketch)

One pedagogical way to visualise: sample a point in the rectangle and keep it if it lies under π.

R: visualising accept vs reject (one iteration sketch)

One pedagogical way to visualise: sample a point in the rectangle and keep it if it lies under π.

```
# Draw many proposal points (y, v) uniformly under the envelope height 3/4
# and colour by accepted/rejected.

N <- 4000
y <- runif(N, 0, 2)
v <- runif(N, 0, 3/4)

pi_y <- (3/4) * y * (2 - y)
acc <- (v <= pi_y)

plot(y, v, pch=16, cex=0.6, col=ifelse(acc, "black", "gray"),
     xlab="y", ylab="v", main="Rejection sampling picture: accept under pi(y)")

# overlay pi(y)
ys <- seq(0, 2, length.out=200)
lines(ys, (3/4)*ys*(2-ys), lwd=2)
```

R: visualising accept vs reject (one iteration sketch)

One pedagogical way to visualise: sample a point in the rectangle and keep it if it lies under π.

```
# Draw many proposal points (y, v) uniformly under the envelope height 3/4
# and colour by accepted/rejected.

N <- 4000
y <- runif(N, 0, 2)
v <- runif(N, 0, 3/4)

pi_y <- (3/4) * y * (2 - y)
acc <- (v <= pi_y)

plot(y, v, pch=16, cex=0.6, col=ifelse(acc, "black", "gray"),
     xlab="y", ylab="v", main="Rejection sampling picture: accept under pi(y)")

# overlay pi(y)
ys <- seq(0, 2, length.out=200)
lines(ys, (3/4)*ys*(2-ys), lwd=2)
```

When C is tight

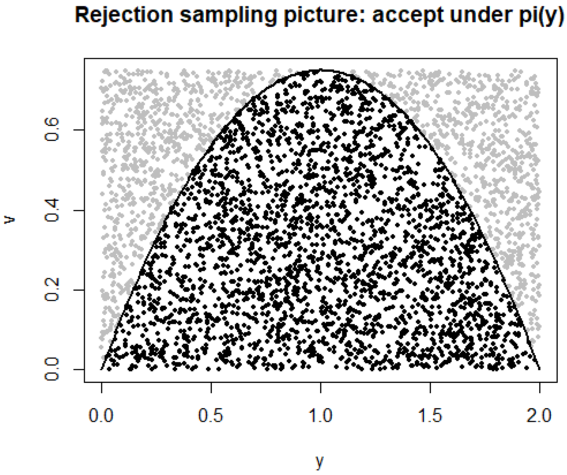
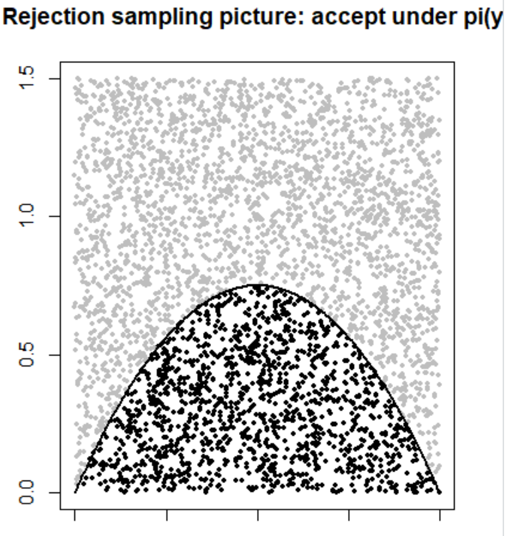


Figure: When envelope constant C is tight, the rejection sampling can be fairly efficient.

When C is badly chosen



Proposition 4.1 (Efficiency)

Proposition 4.1. In a rejection sampling algorithm with envelope constant c, the number of proposals needed until one sample is accepted follows a geometric distribution with success probability $1/c$. In particular, the mean number of proposals per accepted sample is c.

Proposition 4.1 (Efficiency)

Proposition 4.1. In a rejection sampling algorithm with envelope constant c, the number of proposals needed until one sample is accepted follows a geometric distribution with success probability $1/c$. In particular, the mean number of proposals per accepted sample is c.

Interpretation:

- **Smaller** $c \Rightarrow$ **higher acceptance rate** $\Rightarrow$ more efficient.
- Tight envelopes (minimal c) are best, when possible.

Lab preview: what you will code

Inverse transform sampling lab tasks typically look like:

- derive F and F^{-1} for a given distribution (when possible)
- generate many samples using $F^{-1}(U)$
- validate: histogram vs theoretical density, sample mean/variance checks

Rejection sampling lab tasks typically look like:

- choose a proposal q and constant C
- implement accept/reject
- estimate acceptance rate empirically
- compare different envelopes for efficiency

Why should we care about developing good sampling algorithms?

Because in Bayesian inference we often need samples from a posterior like

$$\pi(\theta \mid y) \;\propto\; \pi(y \mid \theta)\,\pi(\theta),$$

and this posterior is often:

- not in a conjugate family,
- high-dimensional,
- only known up to a normalising constant,
- difficult to sample from directly.

Why should we care about developing good sampling algorithms?

Because in Bayesian inference we often need samples from a posterior like

$$\pi(\theta \mid y) \;\propto\; \pi(y \mid \theta)\, \pi(\theta),$$

and this posterior is often:

- not in a conjugate family,
- high-dimensional,
- only known up to a normalising constant,
- difficult to sample from directly.

In many real problems, **sampling is the computational bottleneck** of Bayesian inference.

When we have two parameters: full conditionals

Suppose $\theta = (\theta_1, \theta_2)$ and we cannot sample from $\pi(\theta_1, \theta_2 \mid y)$ directly, but we can work with the full conditionals:

$$\pi(\theta_1 \mid \theta_2, y), \qquad \pi(\theta_2 \mid \theta_1, y).$$

When we have two parameters: full conditionals

Suppose $\theta = (\theta_1, \theta_2)$ and we cannot sample from $\pi(\theta_1, \theta_2 \mid y)$ directly, but we can work with the full conditionals:

$$\pi(\theta_1 \mid \theta_2, y), \qquad \pi(\theta_2 \mid \theta_1, y).$$

This motivates **Gibbs sampling** (one MCMC method):

$$\theta_2^{(t+1)} \sim \pi(\theta_2 \mid \theta_1^{(t)}, y), \qquad \theta_1^{(t+1)} \sim \pi(\theta_1 \mid \theta_2^{(t+1)}, y).$$

When we have two parameters: full conditionals

Suppose $\theta = (\theta_1, \theta_2)$ and we cannot sample from $\pi(\theta_1, \theta_2 \mid y)$ directly, but we can work with the full conditionals:

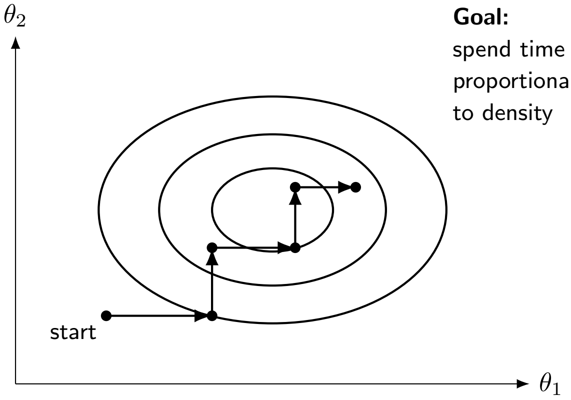
$$\pi(\theta_1 \mid \theta_2, y), \qquad \pi(\theta_2 \mid \theta_1, y).$$

This motivates **Gibbs sampling** (one MCMC method):

$$\theta_2^{(t+1)} \sim \pi(\theta_2 \mid \theta_1^{(t)}, y), \qquad \theta_1^{(t+1)} \sim \pi(\theta_1 \mid \theta_2^{(t+1)}, y).$$

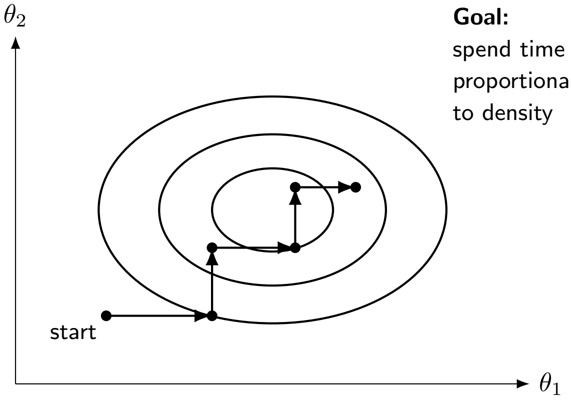
We are not optimising; we are generating a **dependent** sequence of samples whose long-run behaviour matches the target posterior.

MCMC in pictures (random walk around the posterior)



Goal:
spend time
proportional
to density

MCMC in pictures (random walk around the posterior)



Goal:
spend time
proportional
to density

Heuristic: if 90% of posterior mass lies in some region, then about 90% of samples should lie there.

Markov property (why it matters)

In MCMC, the next state depends only on the current state:

$$\mathbb{P}\big(\theta^{(t+1)} \in B \,\big|\, \theta^{(t)}, \theta^{(t-1)}, \dots\big) = \mathbb{P}\big(\theta^{(t+1)} \in B \,\big|\, \theta^{(t)}\big).$$

Markov property (why it matters)

In MCMC, the next state depends only on the current state:

$$\mathbb{P}\big(\theta^{(t+1)} \in B \,\big|\, \theta^{(t)}, \theta^{(t-1)}, \dots\big) = \mathbb{P}\big(\theta^{(t+1)} \in B \,\big|\, \theta^{(t)}\big).$$

This is the **Markov property**. It brings:

- transition kernels / transition matrices (in discrete state spaces),
- stationary distributions,
- convergence theorems (under conditions),
- diagnostic questions: burn-in, mixing, autocorrelation, effective sample size, . . .

Markov property (why it matters)

In MCMC, the next state depends only on the current state:

$$\mathbb{P}\big(\theta^{(t+1)} \in B \,\big|\, \theta^{(t)}, \theta^{(t-1)}, \dots\big) = \mathbb{P}\big(\theta^{(t+1)} \in B \,\big|\, \theta^{(t)}\big).$$

This is the **Markov property**. It brings:

- transition kernels / transition matrices (in discrete state spaces),
- stationary distributions,
- convergence theorems (under conditions),
- diagnostic questions: burn-in, mixing, autocorrelation, effective sample size, ...

That is the mathematical backbone of Chapter 5.

This Friday and next week

- More **worked examples** from Chapter 4 (inverse transform + rejection sampling), and some remaining topics (Zigguart sampling, ABC).
- After reading week: start Chapter 5 **Markov chain Monte Carlo**:
  - Gibbs sampling (full-conditionals)
  - Metropolis–Hastings (accept/reject within a Markov chain)
  - why it works (stationarity, detailed balance, irreducibility/aperiodicity)