Markov-chain Monte Carlo

An (almost) maths-less introduction

Lorenzo Pellis
(with the help of many others!
Thanks: Sam Brand, Simon Spencer, Samik Datta…!)

University of Warwick, UK

UGA, 4th May 2016

Outline

- Introduction
    - The main example – coin tossing
    - Explaining confusing concepts
    - Monte Carlo methods
    - MCMC
- MCMC in practice – Metropolis-Hastings
    - MCMC for coin tossing (1 dim)
    - Diagnosing your MCMC
    - Tricks
    - MCMC in 2 dim
- Play with Matlab or R

The main example – coin tossing
Explaining confusing concepts
Monte Carlo methods
MCMC

INTRODUCTION

The main example – coin tossing
Explaining confusing concepts
Monte Carlo methods
MCMC

INTRODUCTION

Example – coin tosses

- You toss a coin $n = 10$ times. You get:

H H T H H T H H H H

- Arbitrarily, call H a success (1) and T a failure (0):

1 1 0 1 1 0 1 1 1 1

- The number of heads is: $k = 8$
- Questions:
    - is the coin is fair?
    - how confident are you in your answer?
    - and what if you had 79 head out of 100 tosses?
- Key point:
    - Estimating a single number is usually not enough information!

Binomial distribution

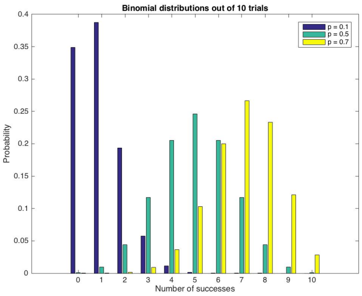
- If we believe that:
    - all tosses are independent of each other
    - output can only be head or tail (and nothing else)
    - head occurs with the same probability p each time
- Then the probability of getting k heads out of n trials, each with success probability p is:

$$\mathbb{P}(k \mid n, p) = \binom{n}{k} p^k (1-p)^{n-k}$$

binomial
coefficient

- Key point: we are **assuming** a model (often without realising it)

Binomial distribution



If $p = 0.5$ we would expect 8 heads out of 10 tosses only 5% of the time

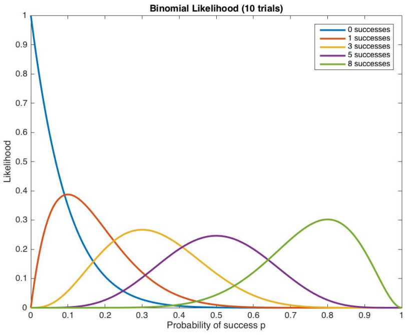
Likelihood function

- For parameters $\boldsymbol{\theta}$ and data $\boldsymbol{D}$, the likelihood function is defined as:

$$\mathcal{L}_{\boldsymbol{D}}(\boldsymbol{\theta}) = \mathbb{P}(\boldsymbol{D}\,|\,\boldsymbol{\theta})$$

- Numerically same as the probability, but different interpretation.
- In $\mathbb{P}(\boldsymbol{D}\,|\,\boldsymbol{\theta})$,
  - $\boldsymbol{\theta}$ is thought as fixed, $\boldsymbol{D}$ varies; and
  - $\int_{\boldsymbol{D}} \mathbb{P}(\boldsymbol{D}\,|\,\boldsymbol{\theta}) = 1$
- In $\mathcal{L}_{\boldsymbol{D}}(\boldsymbol{\theta})$,
  - $\boldsymbol{D}$ is thought as fixed, $\boldsymbol{\theta}$ varies; and
  - $\int_{\boldsymbol{\theta}} \mathcal{L}_{\boldsymbol{D}}(\boldsymbol{\theta}) \neq 1$ in general

Binomial likelihood



The main example – coin tossing

Explaining confusing concepts

Monte Carlo methods

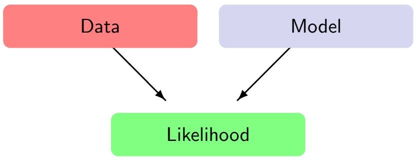
MCMC

INTRODUCTION

Classical VS Bayesian

- This is mostly a philosophical question, but in summary:
- Frequentist (classical) perspective:
  - a parameter has a **true exact value**, which we don't know
- Bayesian perspective:
  - a parameter is a **random variable**, which we can describe using its distribution function

Classical statistics



Data

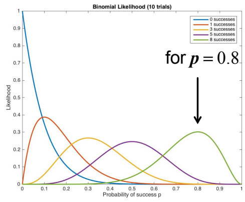
Model

Likelihood

- The goal is to study the likelihood

Maximum likelihood estimator (MLE)

- Given there is only 1 true value of the parameter (classical stats), the interest is in finding the maximum likelihood (ML) estimate:
  - the parameter value at which the likelihood is maximal

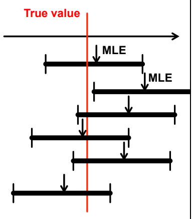


  - This can be found analytically or using numerical methods that “climb” up the hill

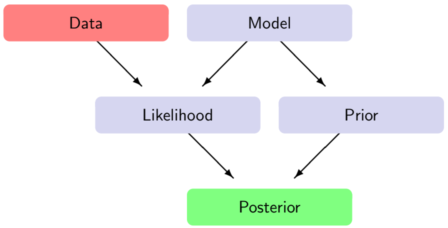
---

Confidence interval

- How confident I am in my ML estimate?
- I can draw **confidence intervals** (CI)
  - assuming asymptotic normality
- Note that:
  - The parameter is fixed
  - The MLE is a random variable
  - The CI is an interval centred in the MLE
  - a 95% CI is a random interval that covers the true value 95% of the times



Bayesian statistics



Data

Model

Likelihood

Prior

Posterior

- The likelihood is still key, but the goal now is to study the posterior

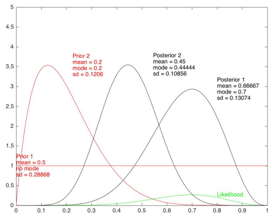
---

Bayes’ theorem

$$\mathbb{P}(\boldsymbol{\theta} \mid \boldsymbol{D}) = \frac{\mathbb{P}(\boldsymbol{D} \mid \boldsymbol{\theta})\mathbb{P}(\boldsymbol{\theta})}{\int_{\boldsymbol{\theta}} \mathbb{P}(\boldsymbol{D} \mid \boldsymbol{\theta})\, \mathrm{d}\boldsymbol{\theta}}$$

Bayes’ theorem

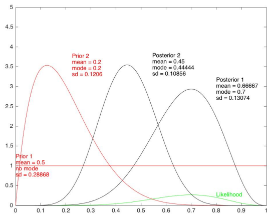
$$\mathbb{P}(\boldsymbol{\theta} \mid \boldsymbol{D}) = \frac{\mathbb{P}(\boldsymbol{D} \mid \boldsymbol{\theta})\mathbb{P}(\boldsymbol{\theta})}{\int_{\boldsymbol{\theta}} \mathbb{P}(\boldsymbol{D} \mid \boldsymbol{\theta})\, \mathrm{d}\boldsymbol{\theta}}$$

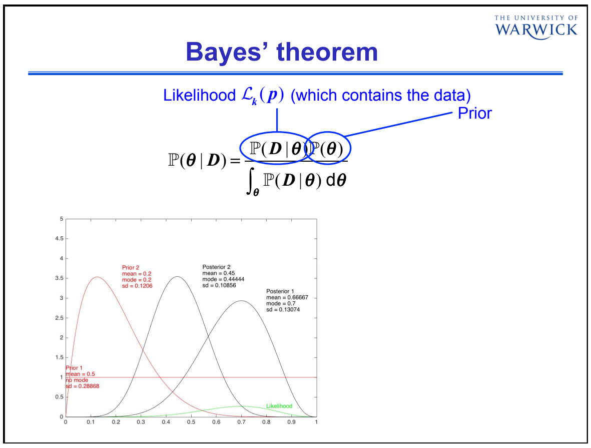


Bayes’ theorem

$$\mathbb{P}(\boldsymbol{\theta} \mid \boldsymbol{D}) = \frac{\mathbb{P}(\boldsymbol{D} \mid \boldsymbol{\theta})\boxed{\mathbb{P}(\boldsymbol{\theta})}}{\int_{\boldsymbol{\theta}} \mathbb{P}(\boldsymbol{D} \mid \boldsymbol{\theta})\, \mathrm{d}\boldsymbol{\theta}}$$

Prior



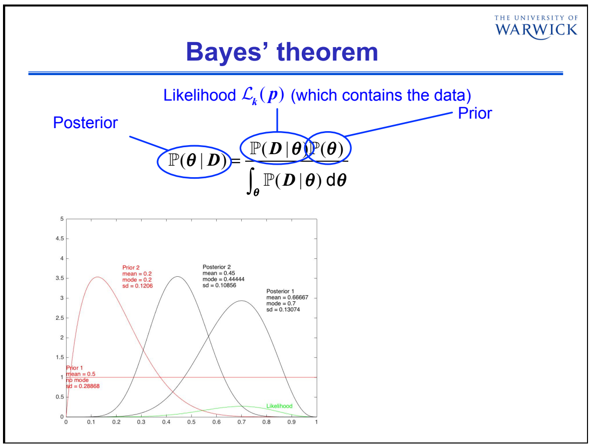


Bayes' theorem

Likelihood $\mathcal{L}_k(\boldsymbol{p})$ (which contains the data)

Prior

$$\mathbb{P}(\boldsymbol{\theta}\mid\boldsymbol{D})=\frac{\mathbb{P}(\boldsymbol{D}\mid\boldsymbol{\theta})\mathbb{P}(\boldsymbol{\theta})}{\int_{\boldsymbol{\theta}}\mathbb{P}(\boldsymbol{D}\mid\boldsymbol{\theta})\,\mathrm{d}\boldsymbol{\theta}}$$



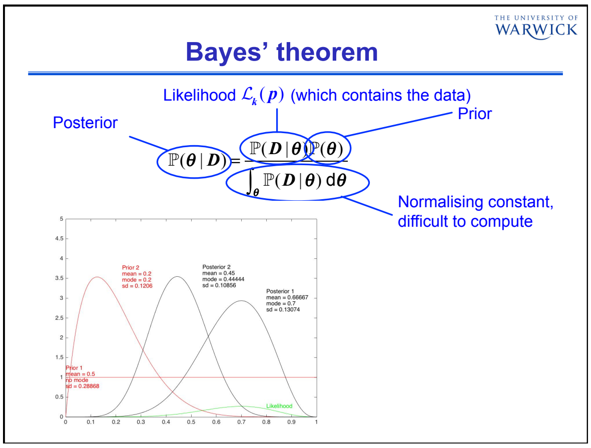
Bayes' theorem

Likelihood $\mathcal{L}_k(\boldsymbol{p})$ (which contains the data)

Posterior

Prior

$$\mathbb{P}(\boldsymbol{\theta}\mid\boldsymbol{D})=\frac{\mathbb{P}(\boldsymbol{D}\mid\boldsymbol{\theta})\mathbb{P}(\boldsymbol{\theta})}{\int_{\boldsymbol{\theta}}\mathbb{P}(\boldsymbol{D}\mid\boldsymbol{\theta})\,\mathrm{d}\boldsymbol{\theta}}$$



Bayes' theorem

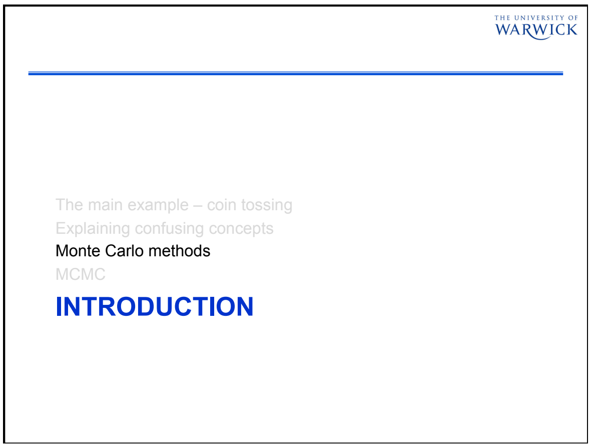
Likelihood $\mathcal{L}_k(\boldsymbol{p})$ (which contains the data)

Posterior

Prior

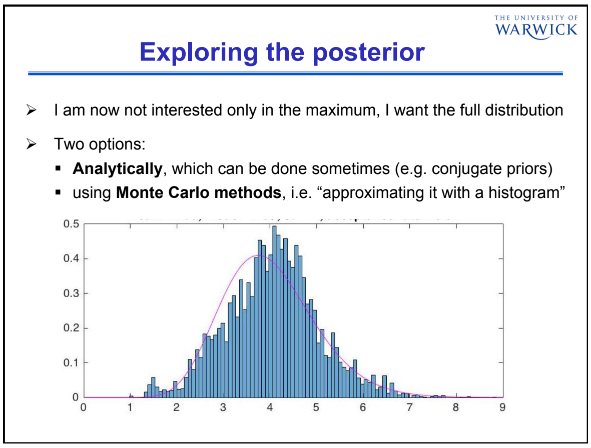
$$\mathbb{P}(\boldsymbol{\theta}\mid\boldsymbol{D})=\frac{\mathbb{P}(\boldsymbol{D}\mid\boldsymbol{\theta})\mathbb{P}(\boldsymbol{\theta})}{\int_{\boldsymbol{\theta}}\mathbb{P}(\boldsymbol{D}\mid\boldsymbol{\theta})\,\mathrm{d}\boldsymbol{\theta}}$$

Normalising constant, difficult to compute



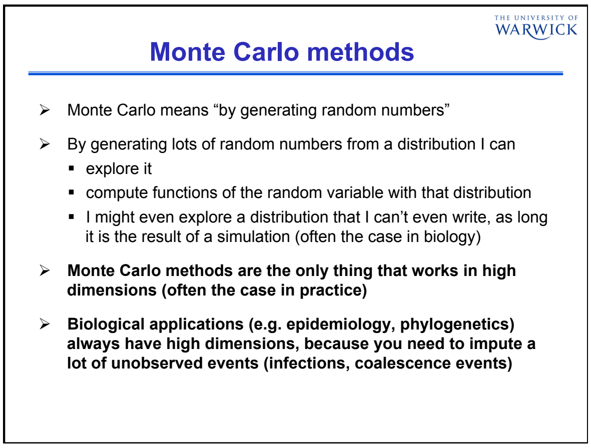
The main example – coin tossing
Explaining confusing concepts
Monte Carlo methods
MCMC

INTRODUCTION



Exploring the posterior

- I am now not interested only in the maximum, I want the full distribution
- Two options:
  - **Analytically**, which can be done sometimes (e.g. conjugate priors)
  - using **Monte Carlo methods**, i.e. "approximating it with a histogram"

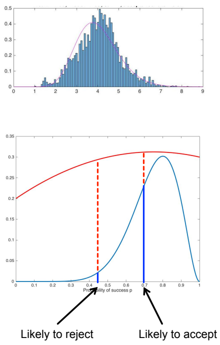


Monte Carlo methods

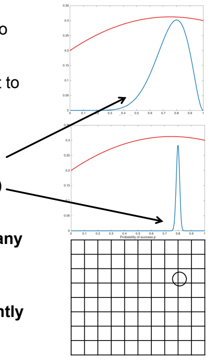
- Monte Carlo means "by generating random numbers"
- By generating lots of random numbers from a distribution I can
  - explore it
  - compute functions of the random variable with that distribution
  - I might even explore a distribution that I can't even write, as long it is the result of a simulation (often the case in biology)
- **Monte Carlo methods are the only thing that works in high dimensions (often the case in practice)**
- **Biological applications (e.g. epidemiology, phylogenetics) always have high dimensions, because you need to impute a lot of unobserved events (infections, coalescence events)**

Monte Carlo Rejection Sampling

- Ordinary Monte Carlo methods explore a function by generating lots of **independent samples**
- If I can draw directly from the distribution of interest (blue line), then I simply do it
- If I can't I can use the **rejection sampling** method:
  - Sampling from another distribution I can sample from (red line) which "cover" the other distribution
  - accept/reject with some probability



Problems with rejection sampling



- Hard to find a good distribution (red line) to sample from
- Hard to find how much I need to "inflate" it to "cover the other curve
- Can be very inefficient
  - Sampling this is easy (few rejections)
  - Sampling this is hard (lots of rejections)
- In general it is **really hard** to explore a distribution by independent samples in **many dimensions** (too many rejections – very inefficient)
- **MCMC is a way to explore more efficiently distributions in many dimensions**

The main example – coin tossing
Explaining confusing concepts
Monte Carlo methods
MCMC

INTRODUCTION

Markov chains

- A Markov chain is a stochastic process, i.e. a system that evolves in time according to a probabilistic rule, where
  - the time is discrete
  - the probabilistic rule at each step depends only on the current step (and nothing before then) – **Markov property**
- It is described by a family of random variables $X_0, X_1, X_2, \dots, X_n, \dots$ with values in a state space $\mathcal{S}$ (discrete or continuous)
- where for each n, X_n depends only on X_{n-1} and not on $X_{n-2}, X_{n-3}, \dots$
- Example: snakes and ladders (it doesn't matter how you arrived where you are; it only matter where you are and the result of rolling the die)
- Under some assumptions, the chain has a **stationary distribution**: if you let it run long enough, no matter where you started from, you end up bouncing around this distribution

Markov chain Monte Carlo

- Goal: we need to explore the posterior distribution
- It might be difficult to sample from it, but it is surprisingly easy to construct a Markov chain that has as stationary distribution the posterior distribution. So,
  - run the Markov chain for long enough, until it has "converged"
  - from that moment onwards I am sampling from the posterior
- Pros: I explore the posterior efficiently, even in high dimensions (because I stay in the regions of high probability, if already there)
- Cons: I am not drawing independent samples anymore:
  - I need more samples to have the same "exploratory power"
  - I don't know many "more samples" are enough
- It's a "*dark art* ": I know that if it has converged it's giving me the right answer, but there is no principled way of telling it has converged

Summary: why MCMC

- In a Bayesian framework, we pull together prior and likelihood (i.e. data) to obtain the posterior
- We want to explore the posterior, but it's difficult to do it analytically
- This is particularly the case for many applied problems
- Monte Carlo methods are ductile and can in principle work in high dimensions, but in practice are very inefficient
- MCMC methods improve the efficiency, at the price of having dependent samples (rather than "more powerful" independent ones)
- We need to make sure that this "dependency" is not ruining our job

MCMC for coin tossing (1 dim)
Diagnosing your MCMC
Tricks
MCMC in 2 dim

MCMC IN PRACTICE

MCMC for coin tossing (1 dim)
Diagnosing your MCMC
Tricks
MCMC in 2 dim

MCMC IN PRACTICE

MCMC structure

3 main steps:

① **Propose new values** for the parameters

② **Compute the likelihood** at the new values

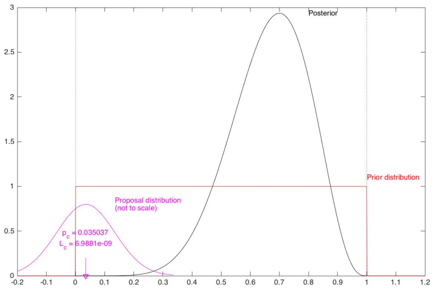
③ Decide whether to:
- **accept**: move to the new values
- or **reject**: stay where you are and count again the current values

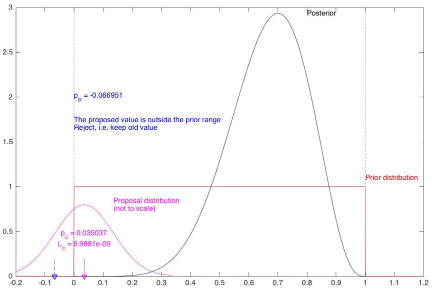
and these three steps are repeated many times!

The most expensive step is typically the computation of the likelihood

---

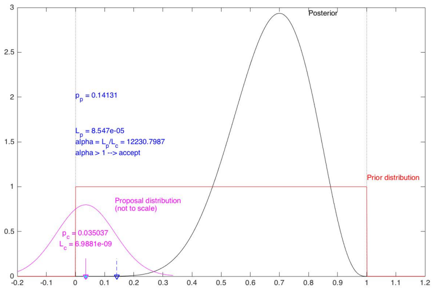
MCMC





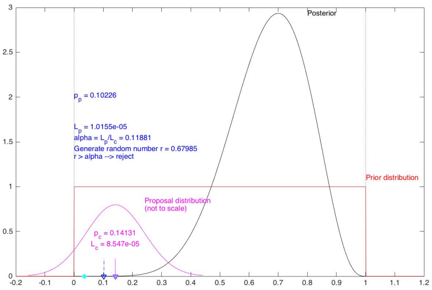
p_p = -0.066951

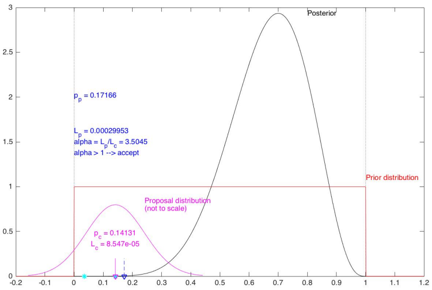
The proposed value is outside the prior range
Reject, i.e. keep old value

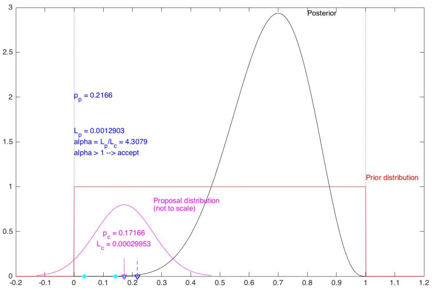


p_p = 0.14131

L_p = 8.547e-05
alpha = L_p/L_c = 12230.7987
alpha > 1 --> accept







Where to start

- First, you need your **data** – this is given
- Second, you need to work out how to compute the **likelihood** – this comes from the model you have assumed
- Then you need to choose (once and for all):
  1. a **prior** distribution for your parameters
  2. a **proposal** distribution
  3. a starting value for the parameters (it should have no influence on the final result)
- Finally you need to choose (and play around with):
  1. the length of your chain
  2. width of proposal distribution
  3. thinning and burn-in

MCMC for coin tosses

- Rules of the game: n coin tosses
- Data:
  - number of heads k
- Goal: explore the **posterior distribution** for the probability p of the coin giving heads, given the data you have seen. This includes information from:
  - the prior distribution on p (my belief / expert opinion)
  - the data

Likelihood $\mathcal{L}_k(p)$ (which contains the data)

Posterior

Prior

$$\mathbb{P}(p \mid k) = \frac{\mathbb{P}(k \mid p)\,\mathbb{P}(p)}{\int_0^1 \mathbb{P}(k \mid p)\,\mathrm{d}p}$$

Good news:
this you can ignore!

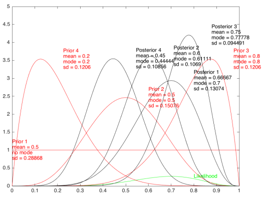
The likelihood

- Our model assumes that the number of heads is binomially distributed. Implicitly, that means:
  - all tosses are independent of each other
  - output can only be head or tail (and nothing else)
  - head occurs with the same probability p each time
- I have tossed the coin n times and I have seen head k times:

$$\mathcal{L}_k(p) = \mathbb{P}(k \text{ heads} \mid p) = \binom{n}{k} p^k (1-p)^{n-k}$$

Prior distribution

- The prior reflects your knowledge (or expert opinion), e.g.
  - if you believe the coin is likely to be fair (maybe because you have seen many fair coins and very few non-fair ones), you will choose a prior highly peaked around $p = 0.5$
  - if you know nothing at all and don't trust anybody, or if you want all the information to come from your data, choose a flat one
- The choice of the prior matters:
- The difference between your prior and your posterior tells you how much you learnt from the data
- Simplest choice: flat



Common prior distributions

- On $[0,1]$ (e.g. for probabilities):
  - uniform, i.e. flat (the pdf is constant = 1 in $[0,1]$ and 0 outside)
  - Beta distribution
- On $[0,+\infty)$ (e.g. variances, infection rate, recovery rate…):
  - flat – this is an improper prior (the pdf is constant "= 0")
  - exponential distribution
  - lognormal distribution
- On $\mathbb{R} = (-\infty,+\infty)$ (e.g. mean of normal distribution)
  - flat – again, improper (the pdf is constant "=0")
  - Normal
- The parameters describing your prior are called hyper-parameters

Proposal distribution

- In theory, it doesn't matter which one you choose
- In practice, you want one that guarantees your chain "mixes well", i.e.
  - you move around (instead of staying for long on the same place)
  - you explore your parameter space quickly
- Unless you know what you are doing (e.g. Gibbs sampling), **choose a Normal distribution**:
  - with **mean** the previous value of your parameter
  - with a **standard deviation** you will tune by trial and error
- The Normal is convenient because it's symmetric (see below)
- In more than 1 dimension, use a multivariate Normal (you can play with the covariance matrix)

Acceptance probability

- This is the clever bit that allows the miracle to work
- At each step you have:
  - An old (current) parameter p and a new (proposed) one p'
  - The likelihood $\mathcal{L}$ and the prior $\mathbb{P}$ computed in both p and p'
  - The probability of your proposal distribution Q making you jump:
    - » from p to p': $Q(p \to p')$
    - » and backwards: $Q(p' \to p)$
- Then you compute:

$$\tilde{\alpha} = \frac{\mathcal{L}(p')\,Q(p' \to p)\,\mathbb{P}(p')}{\mathcal{L}(p)\,Q(p \to p')\,\mathbb{P}(p)} = \frac{\mathcal{L}(p')\,\mathbb{P}(p')}{Q(p \to p')} \Big/ \frac{\mathcal{L}(p)\,\mathbb{P}(p)}{Q(p' \to p)}$$

- and you accept p' with probability: $\alpha = \min\{1, \tilde{\alpha}\}$

Acceptance probability

- This is the clever bit that allows the miracle to work
- At each step you have:
  - An old (current) parameter p and a new (proposed) one p'
  - The likelihood $\mathcal{L}$ and the prior $\mathbb{P}$ computed in both p and p'
  - The probability of your proposal distribution Q making you jump:
    - » from p to p': $Q(p \to p')$
    - » and backwards: $Q(p' \to p)$
- Then you compute:

Prior

$$\tilde{\alpha} = \frac{\mathcal{L}(p')\,Q(p' \to p)\,\boxed{\mathbb{P}(p')}}{\mathcal{L}(p)\,Q(p \to p')\,\boxed{\mathbb{P}(p)}} = \frac{\mathcal{L}(p')\,\mathbb{P}(p')}{Q(p \to p')} \Big/ \frac{\mathcal{L}(p)\,\mathbb{P}(p)}{Q(p' \to p)}$$

- and you accept p' with probability: $\alpha = \min\{1, \tilde{\alpha}\}$

Acceptance probability

- This is the clever bit that allows the miracle to work
- At each step you have:
  - An old (current) parameter p and a new (proposed) one p'
  - The likelihood $\mathcal{L}$ and the prior $\mathbb{P}$ computed in both p and p'
  - The probability of your proposal distribution Q making you jump:
    - » from p to p': $Q(p \to p')$
    - » and backwards: $Q(p' \to p)$
- Then you compute:

Posterior

Prior

$$\tilde{\alpha} = \frac{\mathcal{L}(p')\,Q(p' \to p)\,\boxed{\mathbb{P}(p')}}{\mathcal{L}(p)\,Q(p \to p')\,\boxed{\mathbb{P}(p)}} = \frac{\boxed{\mathcal{L}(p')\,\mathbb{P}(p')}}{Q(p \to p')} \Big/ \frac{\boxed{\mathcal{L}(p)\,\mathbb{P}(p)}}{Q(p' \to p)}$$

- and you accept p' with probability: $\alpha = \min\{1, \tilde{\alpha}\}$

Simpler acceptance probability

- When computing

$$\tilde{\alpha} = \frac{\mathcal{L}(p')\,Q(p' \to p)\,\mathbb{P}(p')}{\mathcal{L}(p)\,Q(p \to p')\,\mathbb{P}(p)}$$

- If the prior is constant in the allowed range, it cancels out
- If you fall outside the allowed range of the prior $\mathbb{P}(p') = 0$, i.e. $\tilde{\alpha} = 0$ and you reject for sure
- If Q is symmetric, the probability of jumping in one direction or backwards are identical, and Q also cancels out
  - This is why the Normal distribution is a good choice
- If the likelihood has some common factors independent of p, they cancel out (e.g. the binomial coefficient)

Starting value for parameters

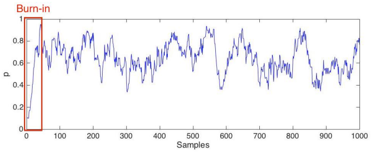
- When running MCMC, you have to wait until your chain has "converged", i.e. has run long enough that:
  - you have forgotten your initial conditions
  - you are bouncing around the stationary distribution (which is exactly your posterior by construction!)
- Therefore, it doesn't matter where you start from – **choose what you prefer**!
- Further, it might be useful to choose different starting values to ensure that each run still converges to the same stationary distribution (as it should)

Length of chain

- You should run your chain long enough
- You want 1,000-10,000 (almost) independent samples (nice histogram)
- Rule of thumb:
  - run your chain for 1,000 steps
  - look at the autocorrelation plot
  - check after how many steps the correlation becomes small (i.e. between the horizontal lines in the plot), say 50
  - you want to run your chain 50 x how many (almost) independent samples
- If the chain is too big to save on your computer you can "thin" it by a factor τ, i.e. save one output every τ

Burn-in

- You only want a sample from the stationary distribution (= the posterior)
- The **burn-in** is the initial part of the chain, that depends of where you started from, and that looks "different" from the rest of the chain, because you haven't yet converged
- The burn-in should be identified by eye from the trace plot and discarded



Width of the proposal

- If your proposal is very wide, you want to jump very far, usually out of the range of the prior or into regions of low posterior probability:
  - high rejection rates
  - chain visibly constant in bits

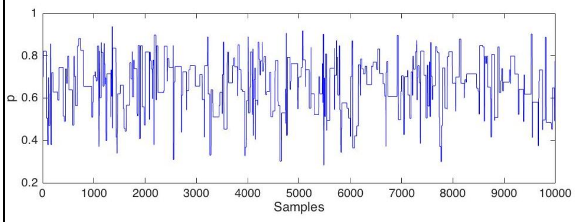
  $\Longrightarrow$ Choose a smaller standard deviation
- If your proposal is very narrow, your proposals are almost always accepted, but you move very slowly around the parameter space:
  - high acceptance rates
  - chain with visible broad oscillations

  $\Longrightarrow$ Choose a larger standard deviation
- Both the problems above lead to large autocorrelation (bad)
- In 1 dim, acceptance rates should be between 20 and 60%

MCMC for coin tossing (1 dim)
Diagnosing your MCMC
Tricks
MCMC in 2 dim

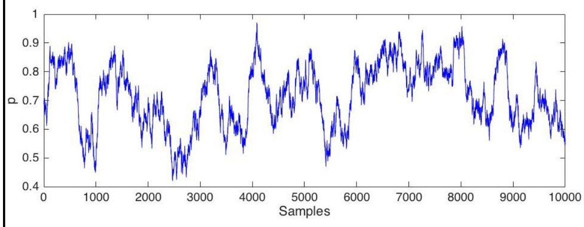
MCMC IN PRACTICE

Diagnostics (1 dim)



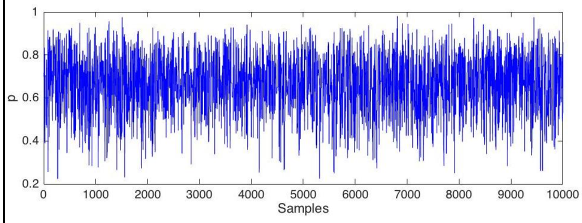
- Proposal too wide: reduce standard deviation

Diagnostics (1 dim)



- Proposal too narrow: increase standard deviation

Diagnostics (1 dim)



- Good mixing!
- To be sure, you should zoom into a section towards the beginning and a section towards the end and they should look similar!

MCMC for coin tossing (1 dim)
Diagnosing your MCMC
Tricks
MCMC in 2 dim

MCMC IN PRACTICE

Trick 1: log-likelihood

Instead of using the likelihood, use the log-likelihood

- This should always be done:
  - it has no drawbacks
  - reduce numerical error (likelihood can have really small values)
  - cheaper computations (products become sums, exponentials become products)
- E.g. binomial likelihood $\mathcal{L}_k(p)=\binom{n}{k}p^k(1-p)^{n-k}$
  - Ignore constant factors independent of p (they cancel out in $\tilde{\alpha}$)
  - Take the log
  $$\log\mathcal{L}_k(p)\propto k\log p+(n-k)\log(1-p)$$
- At this point, it is convenient to work with $\log\tilde{\alpha}$ and to use the log for all its factors (proposal and prior)

Trick 1: log-likelihood

Instead of using the likelihood, use the log-likelihood

- This should always be done:
  - it has no drawbacks
  - reduce numerical error (likelihood can have really small values)
  - cheaper computations (products become sums, exponentials become products)
- E.g. binomial likelihood $\mathcal{L}_k(p)=\cancel{\binom{n}{k}}p^k(1-p)^{n-k}$
  - Ignore constant factors independent of p (they cancel out in $\tilde{\alpha}$)
  - Take the log
  $$\log\mathcal{L}_k(p)\propto k\log p+(n-k)\log(1-p)$$
- At this point, it is convenient to work with $\log\tilde{\alpha}$ and to use the log for all its factors (proposal and prior)

Trick 2: avoid useless calculations

- Calculating the likelihood is usually very expensive
  - If you propose a new parameter where the prior is 0 (e.g. p outside $[0,1]$), reject immediately, before calculating $\tilde{\alpha}$
- Cheaper to test an “if” statement than to generate random numbers:
  - If $\tilde{\alpha} > 1$, accept without generating a random number to choose whether to accept or reject

Trick 3: Always plot the prior

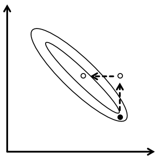
- If your posterior is different, it tells you how much you have learnt from the data
- If your posterior is very similar, it means the data contains very little or no information about that parameters (unidentifiability issue)

MCMC for coin tossing (1 dim)
Diagnosing your MCMC
Tricks
MCMC in 2 dim

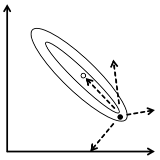
MCMC IN PRACTICE

Proposal in 2 dimension

You can choose whether to:



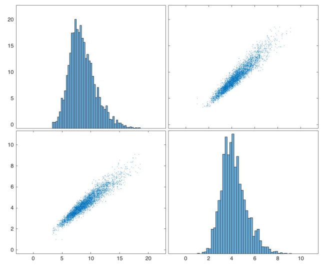
- propose 1 at a time (e.g. from a simple Normal distribution in 1 dim), while keeping the other fixed:
  - OK if parameters are relatively uncorrelated
  - Bad if they are strongly correlated
  - Very good if you can specify analytically the conditional posterior (Gibbs sampling)



- or propose both parameters in one go (e.g. from a multivariate normal) – this is called <u>block update</u>
  - A good solution when the parameters are strongly correlated

Plotmatrix

- Plotmatrix is a command (in Matlab) that creates a fancy plot that can reveal correlation between parameters



Strongly correlated parameters

- Tricks to improve mixing:
  - Block updates, possibly from a multivariate normal “elongated” in the direction of the correlation
  - Re-parameterise your model with new parameters that are less correlated.

PLAY WITH MATLAB OR R

Codes

- "MCMCBinomial" file: play with
  - n_tosses
  - n_heads
  - n_iters
  - thinning
  - sd_proposal (try 0.01 or 5)
  - burnin
- "MCMCEpidemicFinalSizeLargePop" file: play also with
  - Updating in block or not
  - Tuning the sd_proposal for either parameter