

# Lecture 7: Estimation Continued

# Outline

- Likelihood refresher
- Go through a detailed example
- Bayesian methods - main ideas
  - Bayes rule thing
    - Basic idea
    - Cookies from book
  - MCMC
    - Monte carlo simulations
    - Idea behind MCMC
  - Some visual examples but not code
- How to assess goodness of fit
- Real world examples
  - Charlottesville PCA?
  - Maybe something COVID-y? Go through various talks

# Maximum likelihood

**Likelihood function:** the probability of seeing the data we have, assuming that we knew the parameter values.

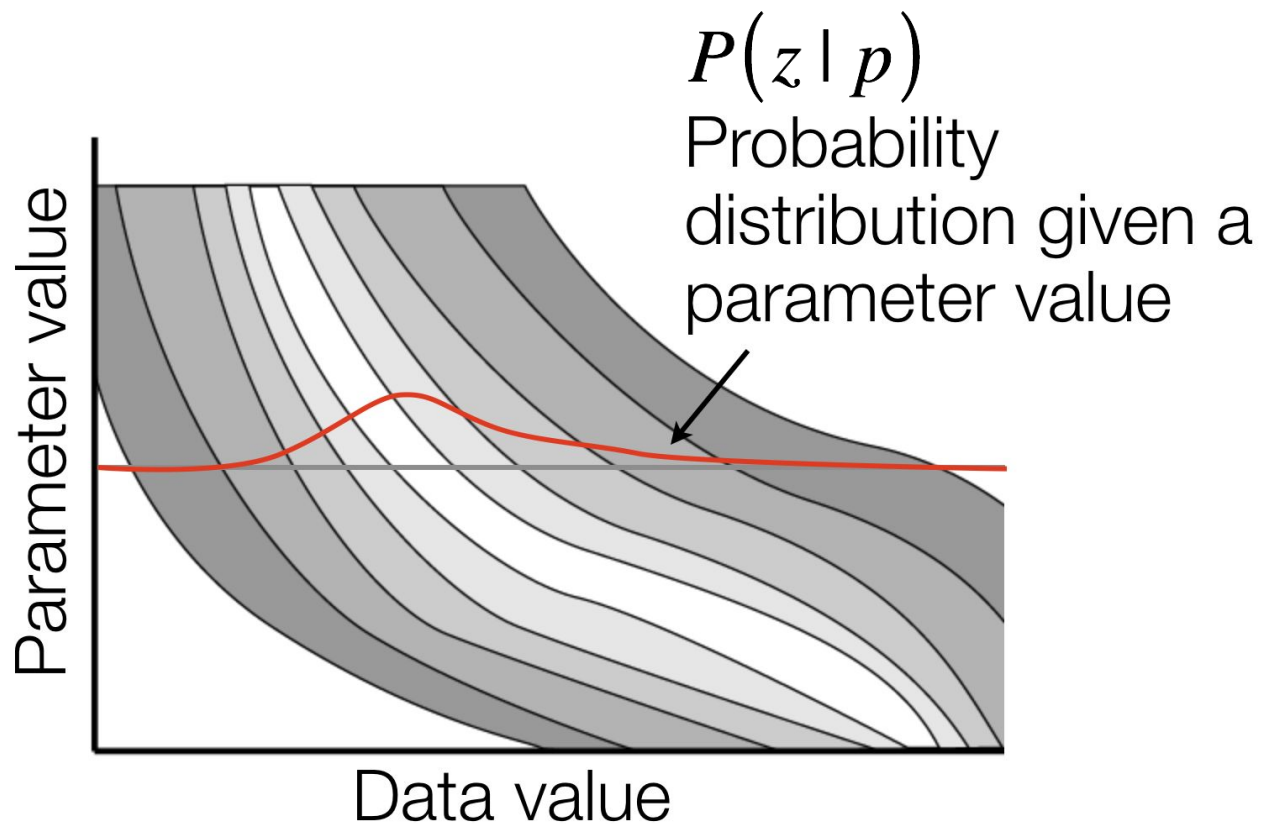
$$P(z \mid p) = f(z, p) = L(p \mid z)$$

Main idea: Re-think of the probability distribution as a function of the data instead of the parameters

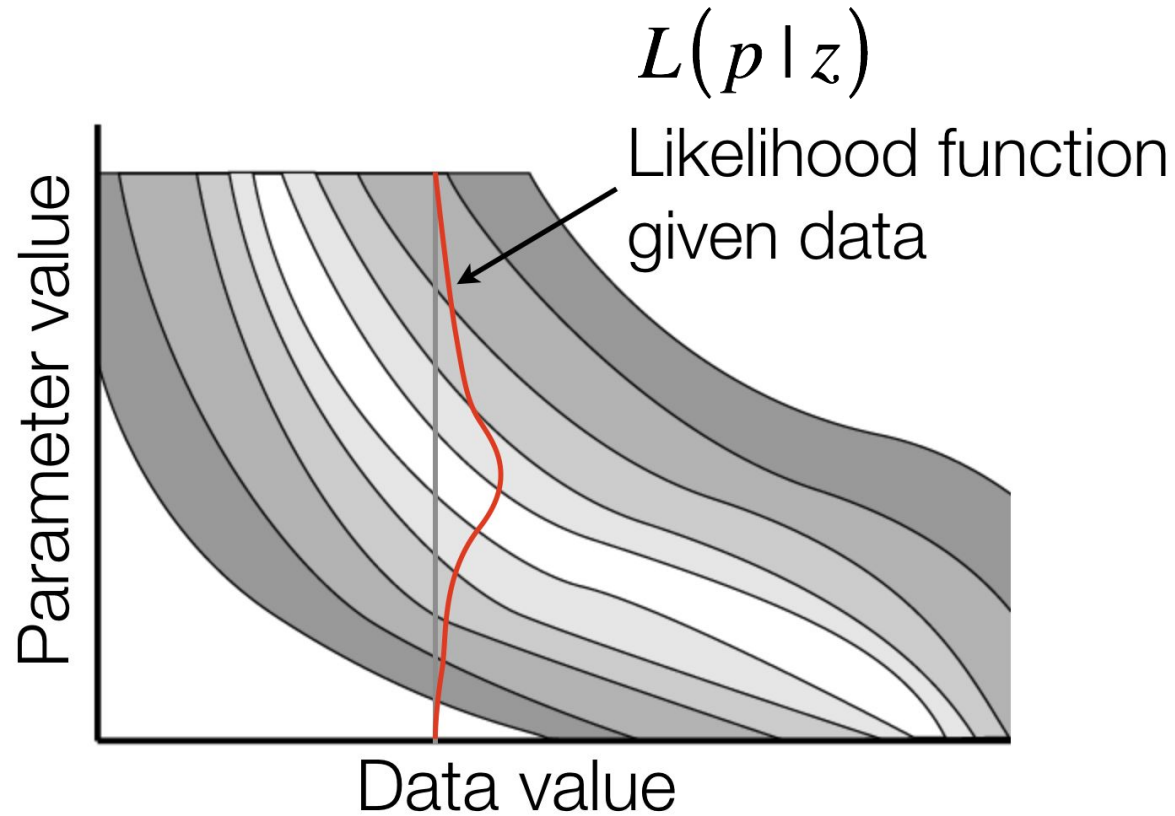
E.g. normal distribution:  $f(z \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(z - \mu)^2}{2\sigma^2}\right) = L(\mu, \sigma^2 \mid z)$

**Find the value of p that maximizes  $L(p \mid z)$**  - this is the **maximum likelihood estimate** (MLE) (most likely given the data) — in other words, L becomes our cost function! (usually actually - log(L) but this is the idea!)

# Likelihood function



# Likelihood function

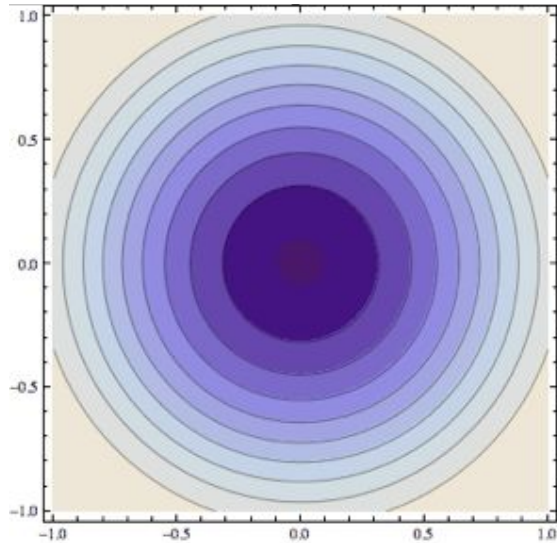


# Maximum likelihood recap

- In general, your likelihood is just the probability distribution of your data, assuming that you knew your parameter values
  - Then, we ‘re-think’ of that distribution as a function of the parameters with the data fixed—this is the likelihood
- Make the likelihood your cost function and find the parameter values that maximize it!
  - In other words, use optimization to figure out: what parameter values make your data very likely to be what the model would predict?
  - For some models, there are known formulas to find the optimum parameter values, but in many cases we have to use numerical (computational) optimization methods

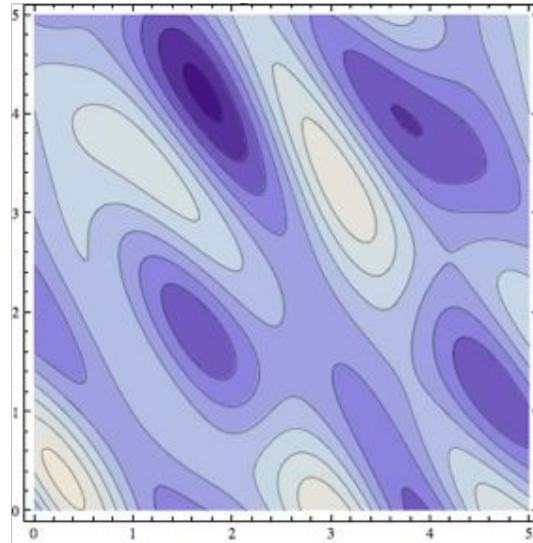
Optimization methods are often developed for single minimum surfaces — multiple minima or canyons (unidentifiability) can cause problems

Best case scenario



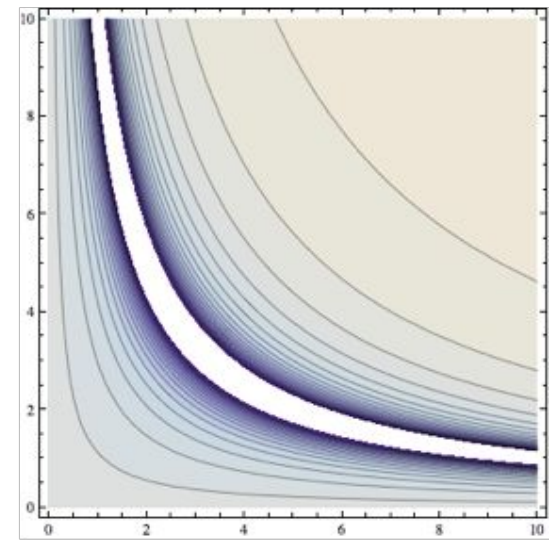
One unique minimum cost function value: we can identify the parameters

Multiple minima



Multiple minima (maybe equally good or maybe not): there are several potential parameter estimates

Unidentifiability



A canyon of equally good cost function values: infinitely many parameter values fit the data equally well so we cannot estimate the parameters

# Bayesian approaches to parameter estimation

- Bayes' Theorem, rewritten for inference problems:

$$P(p \mid z) = P(\text{params} \mid \text{data}) = \frac{P(z \mid p) \cdot P(p)}{P(z)}$$

- Allows one to account for prior information about the parameters
  - E.g. previous studies in a similar population
- Update parameter information based on new data



# Bayesian approaches to parameter estimation

Likelihood

Prior distribution

$$P(p | z) = P(\text{params} | \text{data}) = \frac{P(z | p) \cdot P(p)}{P(z)}$$

The diagram illustrates the components of the Bayesian formula. An arrow points from the word 'Likelihood' to the term  $P(z | p)$  in the numerator. Another arrow points from the words 'Prior distribution' to the term  $P(p)$  in the numerator. A third arrow points from the text 'Normalizing constant (can be difficult to calculate!)' to the denominator  $P(z)$ .

Normalizing constant  
(can be difficult to calculate!)

$$P(z) = \int_p P(z, p) dp$$

## Denominator term - $P(z)$

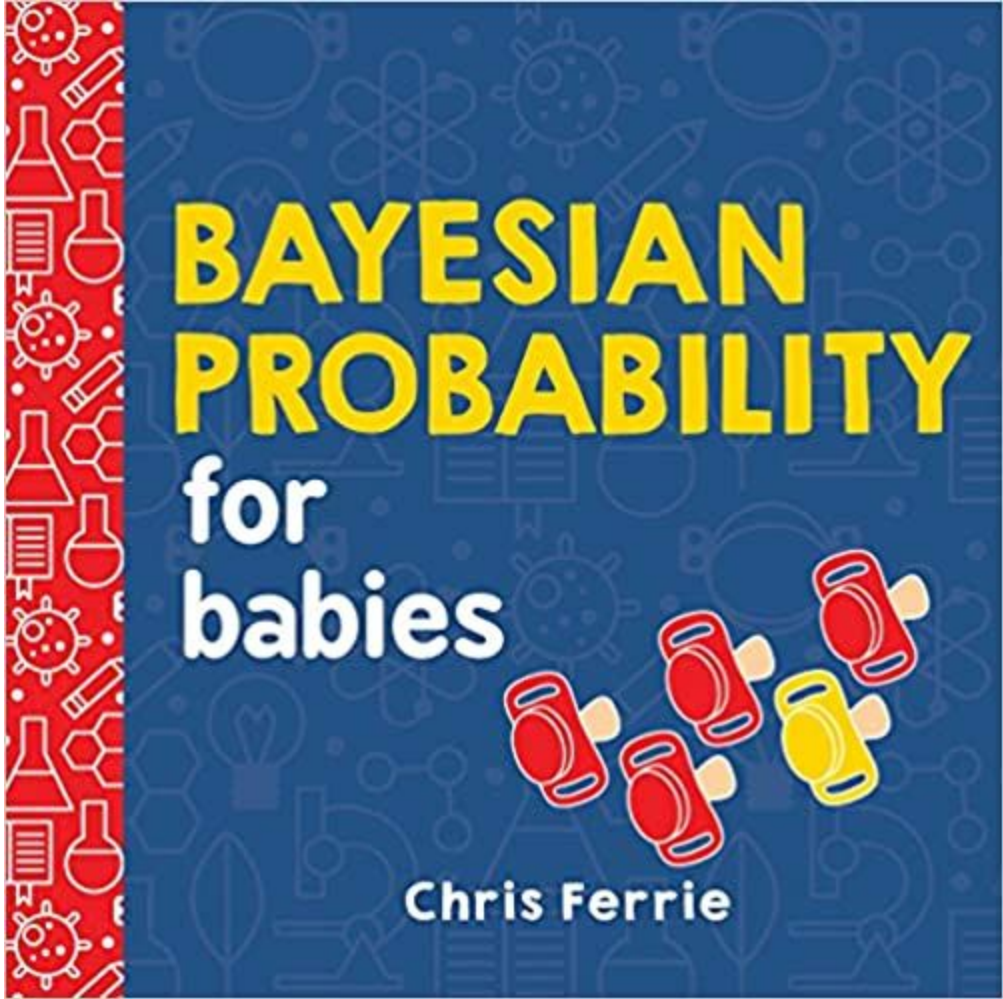
The denominator term:

$$P(z) = \int_p P(z, p) dp$$

This is the probability of seeing the data  $z$  from the model, over all parameter space

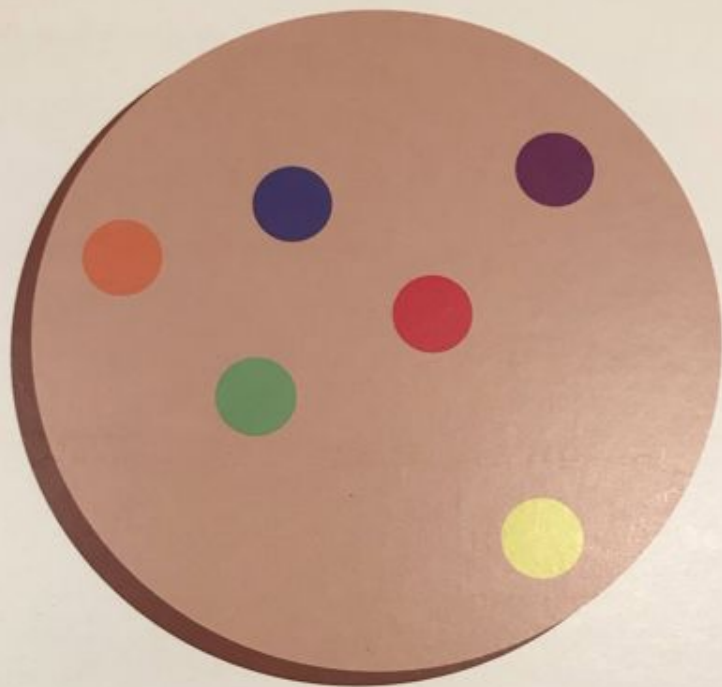
Often doesn't have a closed form solution—evaluating numerically can also be difficult

E.g. if  $p$  is a three dimensional, then if we took 1000 grid points in each direction, the grid representing the function to be integrated has  $1000^3 = 10^9$  points

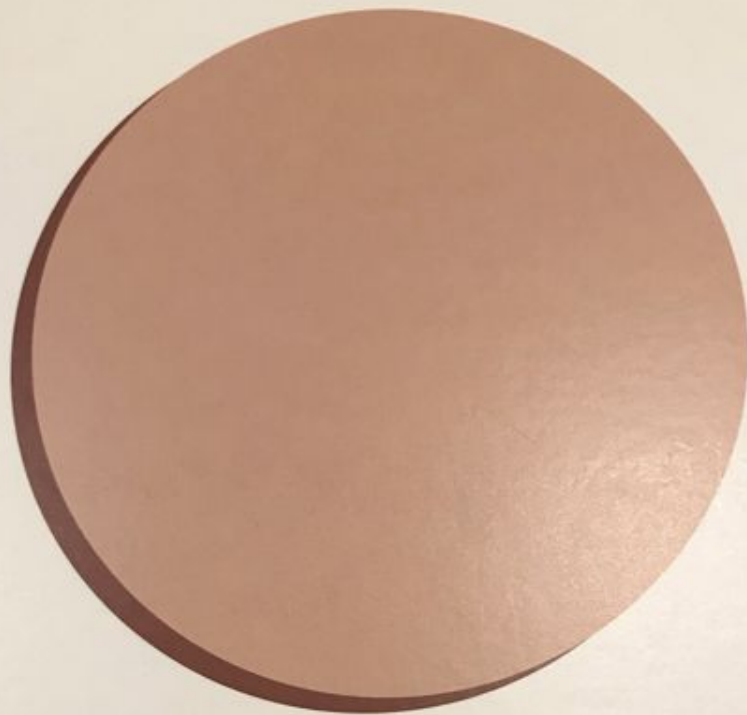


# **BAYESIAN PROBABILITY** for babies

**Chris Ferrie**



**Some cookies have candy.**



**Some don't.**

Our data:



Our model:

- Candy cookie vs. no-candy cookie
- Our model parameters? Just one—an indicator variable  $c$ , where:
  - $c = 0$  if it was a no-candy cookie
  - $c = 1$  if it was a candy cookie



**Did it come from a candy cookie?**

In other words, our  
parameter estimation  
problem is to estimate  $c$ !

Does  $c = 0$  or  $1$ ?

What is the likelihood?

$$L(c = 0 \mid \text{NC bite}) = P(\text{NC bite} \mid c = 0) = 1$$



If the cookie had no candy,  
then every bite would have no candy.

$$\text{Pr}(\text{no candy bite} \mid \text{no candy cookie}) = 1$$

The probability of a no-candy bite,  
given a no-candy cookie, is 1.

What is the likelihood?

$$L(c = 1 \mid \text{NC bite}) = P(\text{NC bite} \mid c = 1) = 1/3$$



If the cookie had candy, then very few bites would have no candy.

$$\Pr(\text{no candy bite} \mid \text{candy cookie}) = 1/3$$

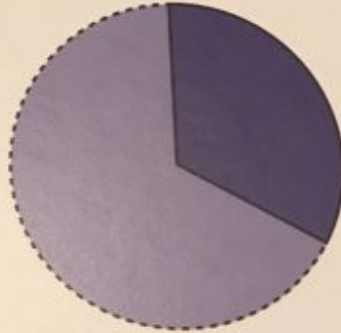
The probability of a no-candy bite, given a candy cookie, is 1/3.



What is the maximum likelihood estimate?

$1 > \frac{1}{3}$ , so we estimate that  $c = 0$

$$\Pr(\text{bite} | \text{no-candy}) > \Pr(\text{bite} | \text{candy})$$

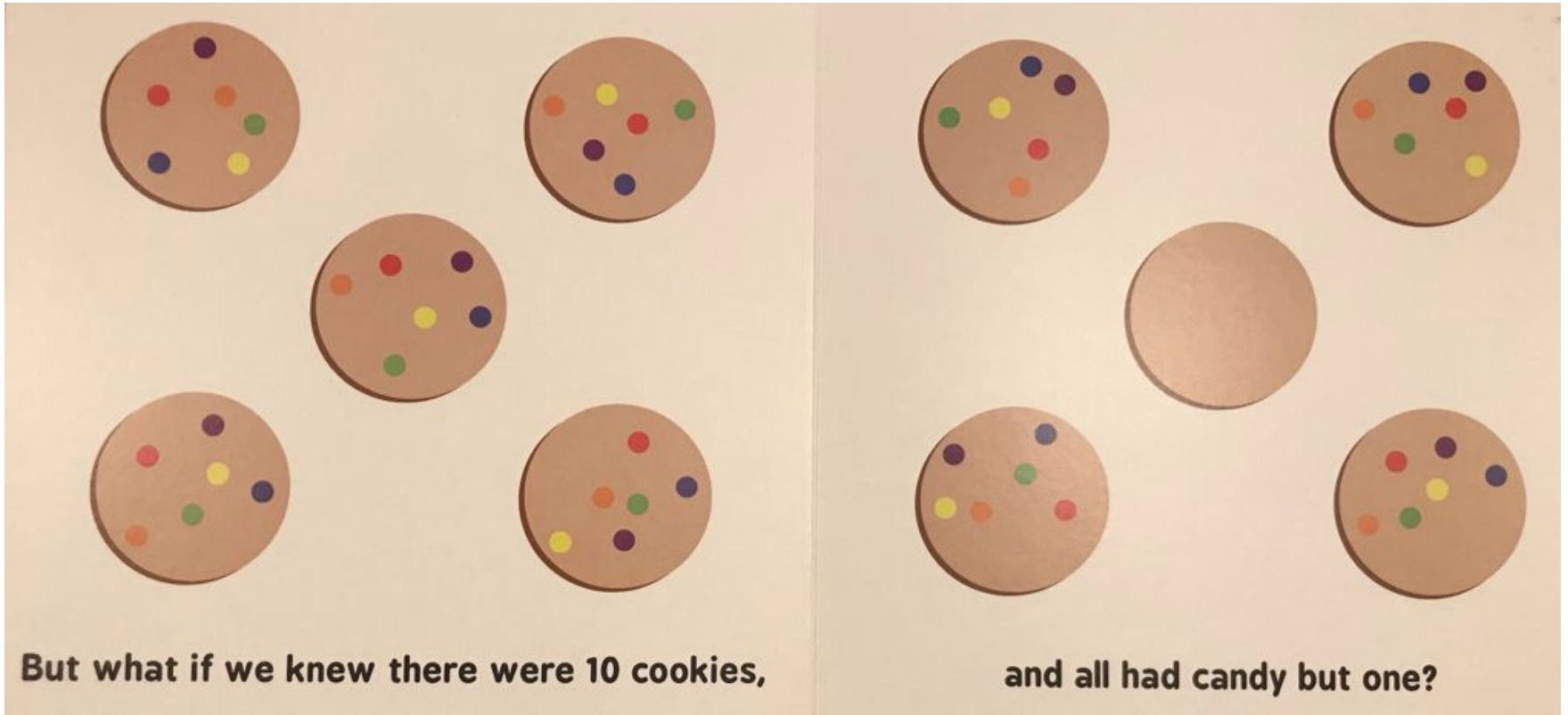


1 is greater than 1/3.

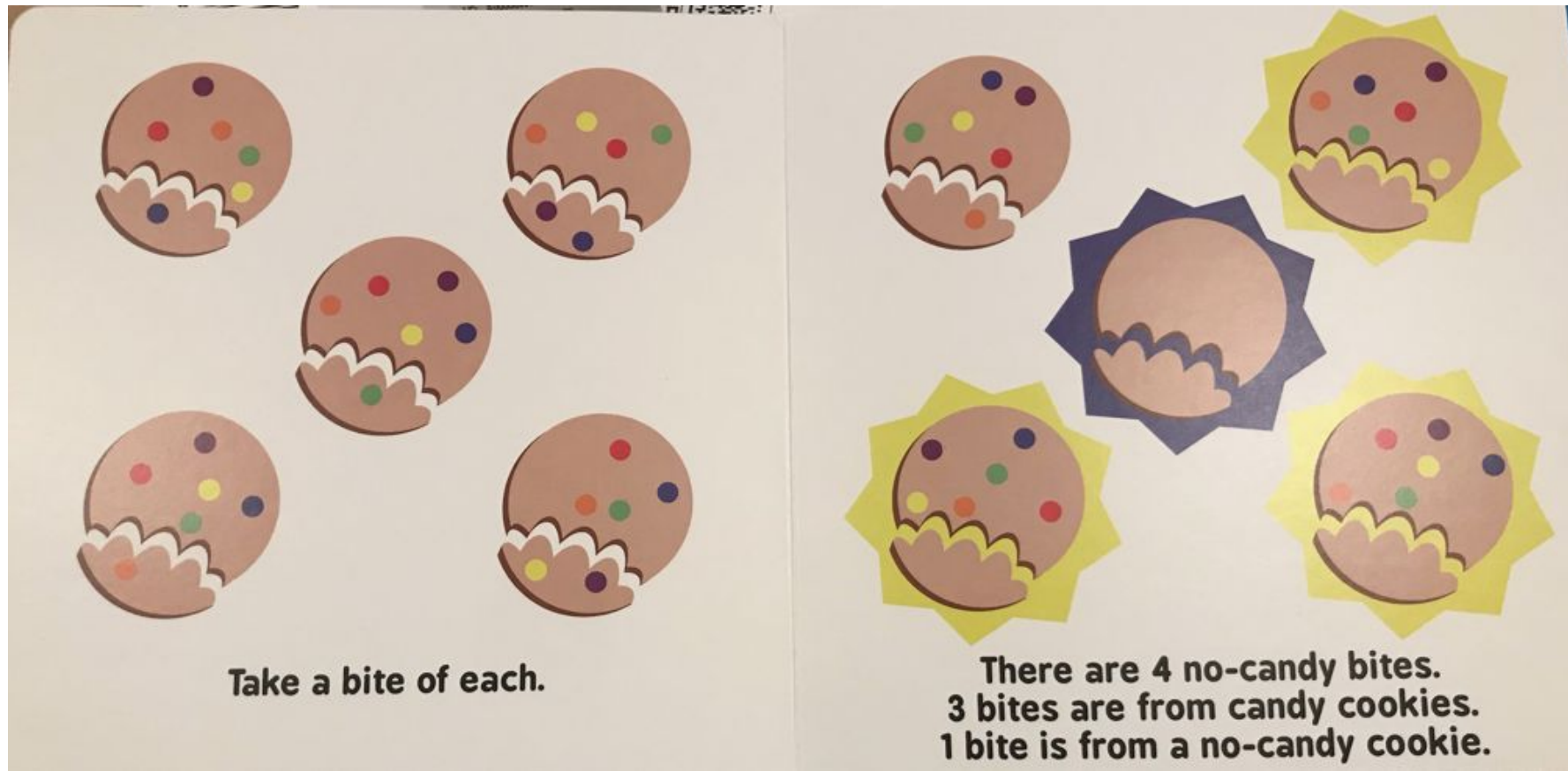


So the no-candy bite probably came from a no-candy cookie!

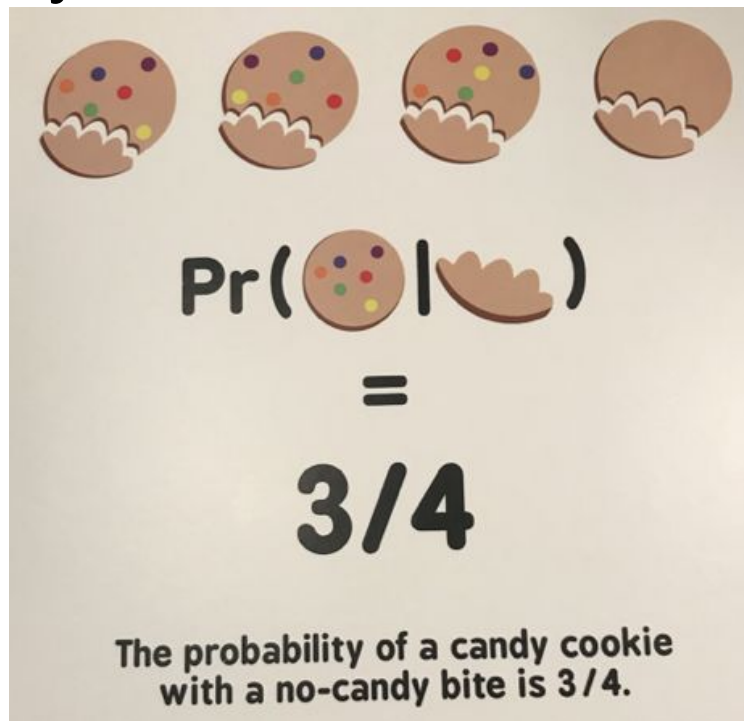
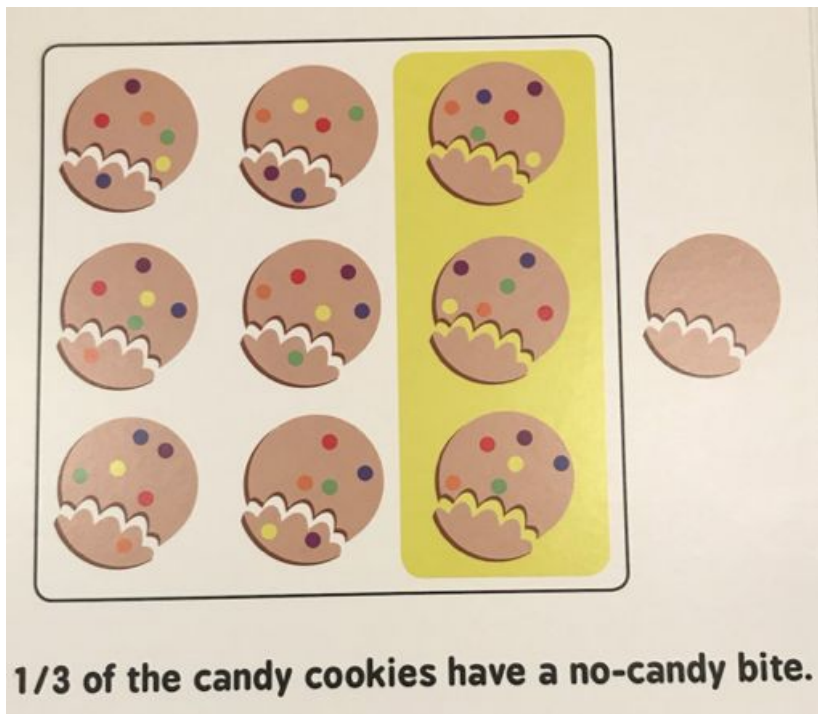
# What about the prior distribution of cookies?



Our data (likelihood) tells us we have a no-candy bite—how many of the bites we could take have no candy?



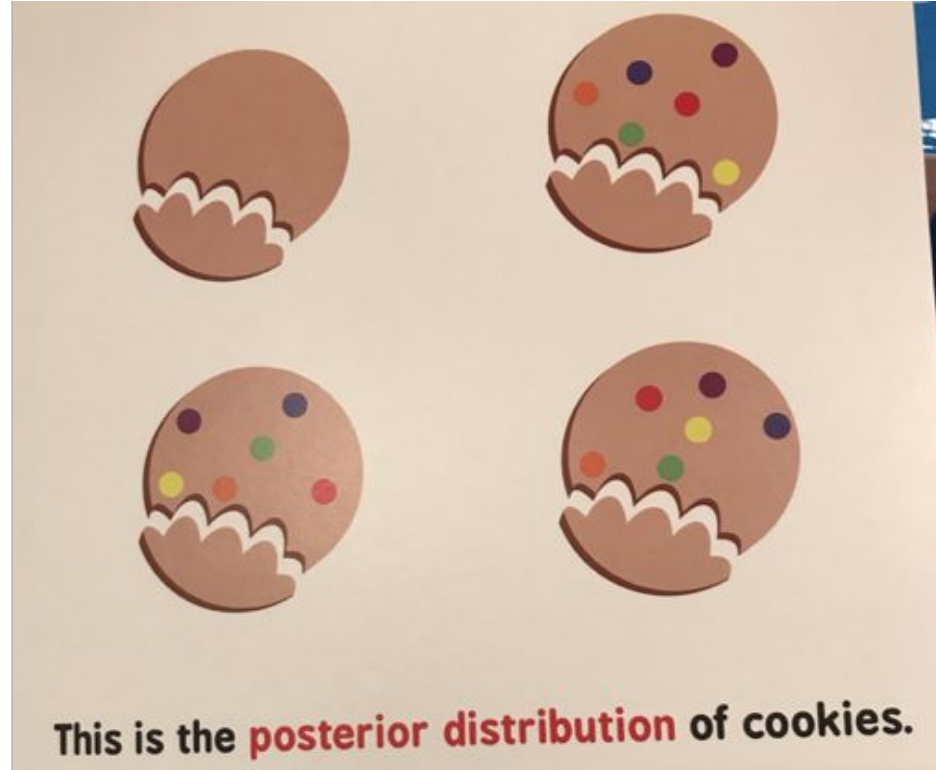
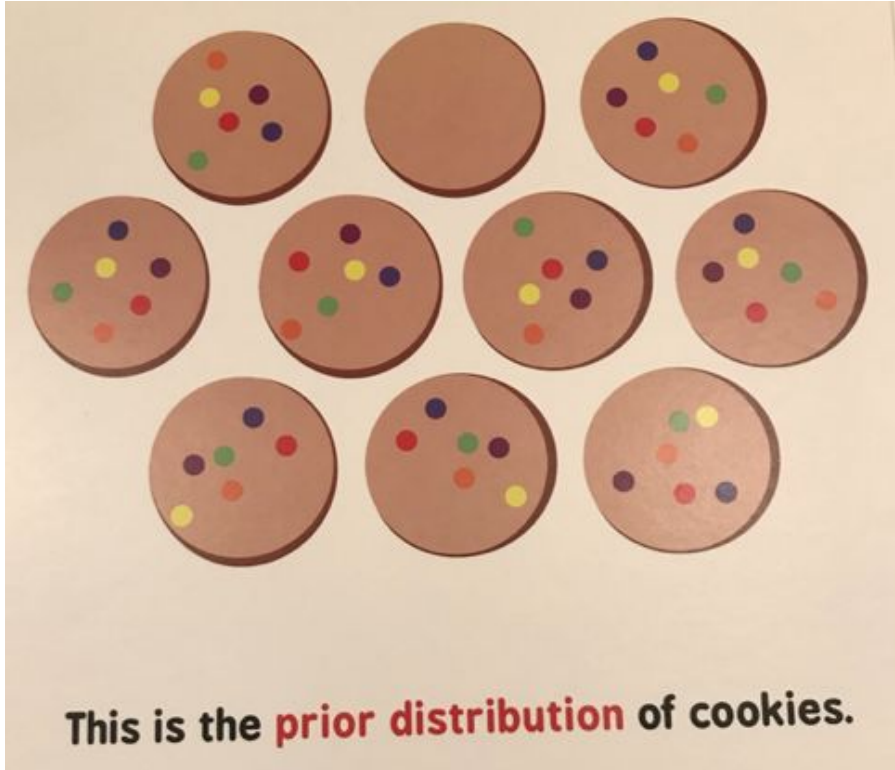
Only 1/3 of each candy cookie bites have no candy,  
but there are a lot more candy cookies!



Prior x Likelihood ~ Posterior

9 x 1/3 = 3 for candy cookies, vs. 1 x 1 = 1 for no-candy cookie

# Bayesian estimation!



# Bayesian approaches to parameter estimation

Likelihood

Prior distribution

$$P(p | z) = P(\text{params} | \text{data}) = \frac{P(z | p) \cdot P(p)}{P(z)}$$

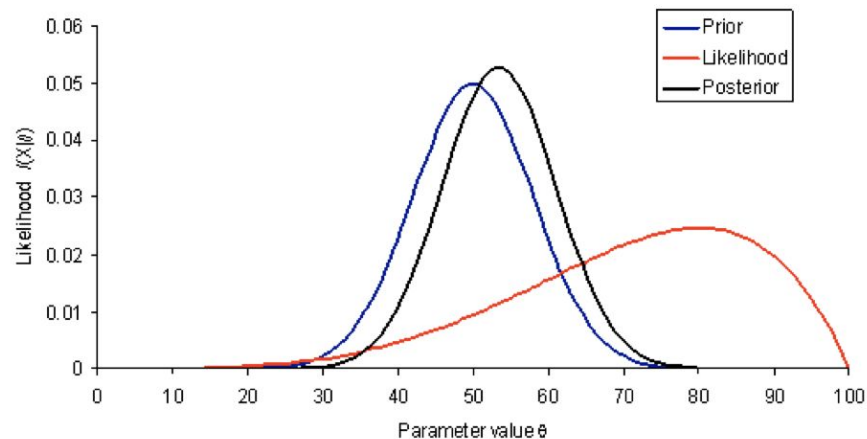
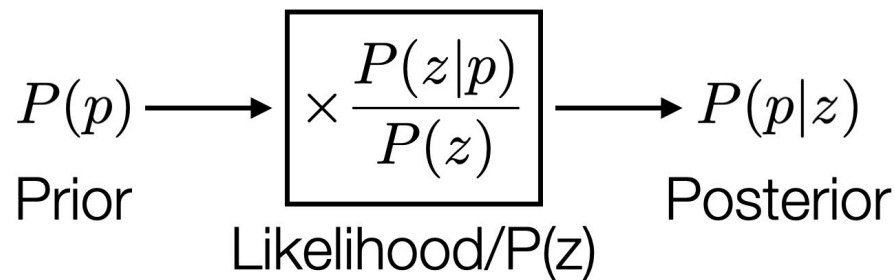
The diagram illustrates the components of the Bayesian formula. An arrow points from the word 'Likelihood' to the term  $P(z | p)$  in the numerator. Another arrow points from the words 'Prior distribution' to the term  $P(p)$  in the numerator. A third arrow points from the text 'Normalizing constant (can be difficult to calculate!)' to the denominator  $P(z)$ .

Normalizing constant  
(can be difficult to calculate!)

$$P(z) = \int_p P(z, p) dp$$

# Bayesian Parameter Estimation

Can think of Bayesian estimation as a function, where take in the prior as an input, and then use the data to update the prior to a new posterior (output)





# Maximum a posteriori (MAP) estimation

Instead of working with the full term, just use the numerator:

$$P(p|z) = \frac{P(z|p) \cdot P(p)}{P(z)}$$

The denominator is a constant, so the numerator is proportional to the posterior we are trying to estimate

Then the  $p$  which yields  $\max(P(z|p) \cdot P(p))$  is the same  $p$  that maximizes  $P(p|z)$

If we only need a point estimate, MAP gets around needing to estimate the denominator



# Sampling methods: approximating a distribution

What if we want to estimate the full posterior distribution—for example, if we want to estimate uncertainty ranges?

How to approximate the posterior distribution without having to calculate the denominator term?

Monte Carlo methods—specifically **Markov Chain Monte Carlo or MCMC**—give us a way to do this

# Markov Chain Monte Carlo (MCMC)

MCMC is a method for sampling from a distribution

**Markov chain:** a type of (discrete) Markov process—a Markov process is a process or procedure where each step only depends on the most recent step (i.e. you don't need the full history of what happened to do the next step of the process, just the current state)

**Monte Carlo methods** are a class of algorithms that use sampling/randomness to solve (usually) deterministic problems

- Talk over a few examples—e.g. the card trick from class, or just the idea of sampling randomly across the parameter space to build up the distribution

# Markov Chain Monte Carlo (MCMC)

MCMC is a method for sampling from a distribution

Here, the distribution we are going to sample is the posterior distribution!

But MCMC is also used for many other things! Can approximate distributions more generally—used in cryptography, calculating neutron diffusion, all sorts of things

# Markov Chain Monte Carlo (MCMC)

**Main idea:** make a Markov chain (i.e. some kind of random process) that converges to the distribution we're trying to sample from (the posterior)

Many MCMC methods are based on random walks (what is a random walk?)

- Set up the random walk to spend more time in higher probability regions

Typically don't need the actual distribution for this, just something proportional—so we can get the relative probability density at two points

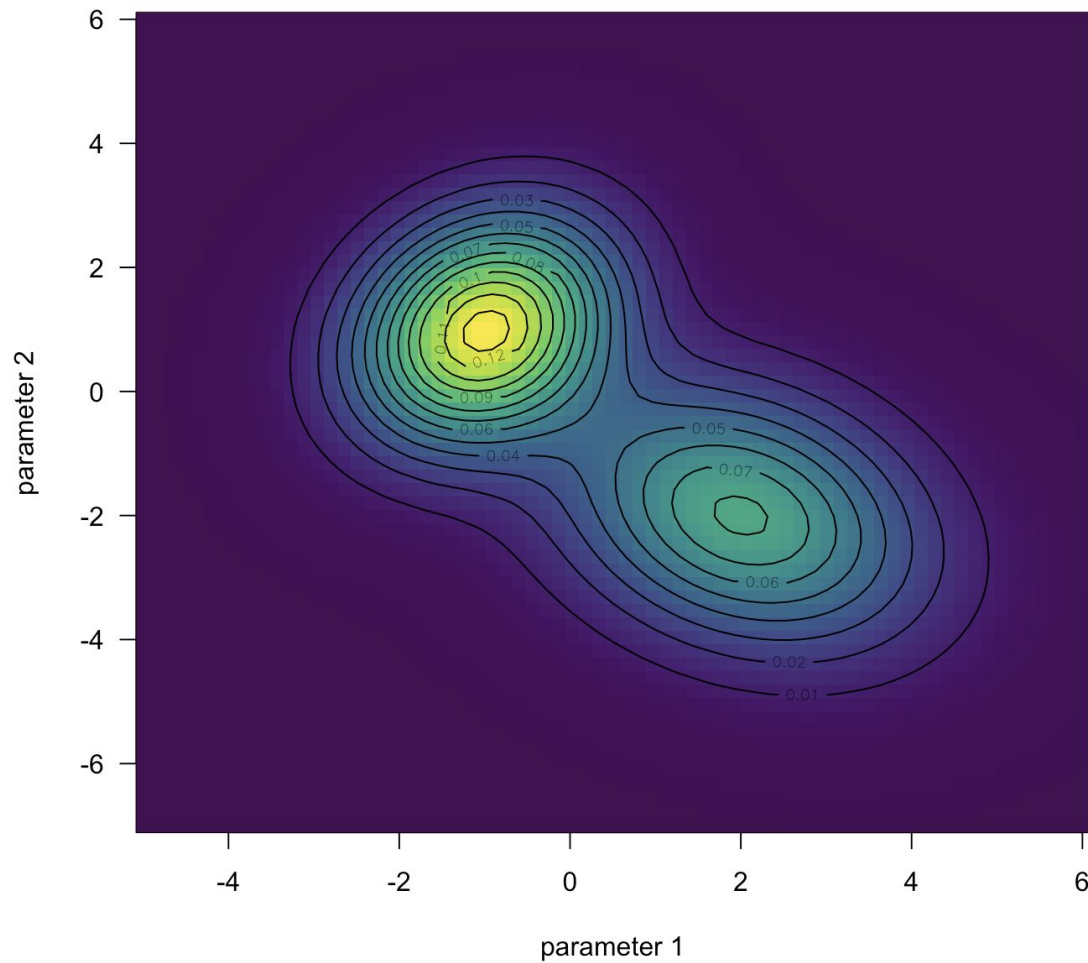
- So we don't need to calculate  $P(z)$ ! We can just use the numerator

# Markov Chain Monte Carlo (MCMC)

The Markov chain will have some transient dynamics (burn-in), and then reach an equilibrium distribution which is the one we're trying to approximate

# Example

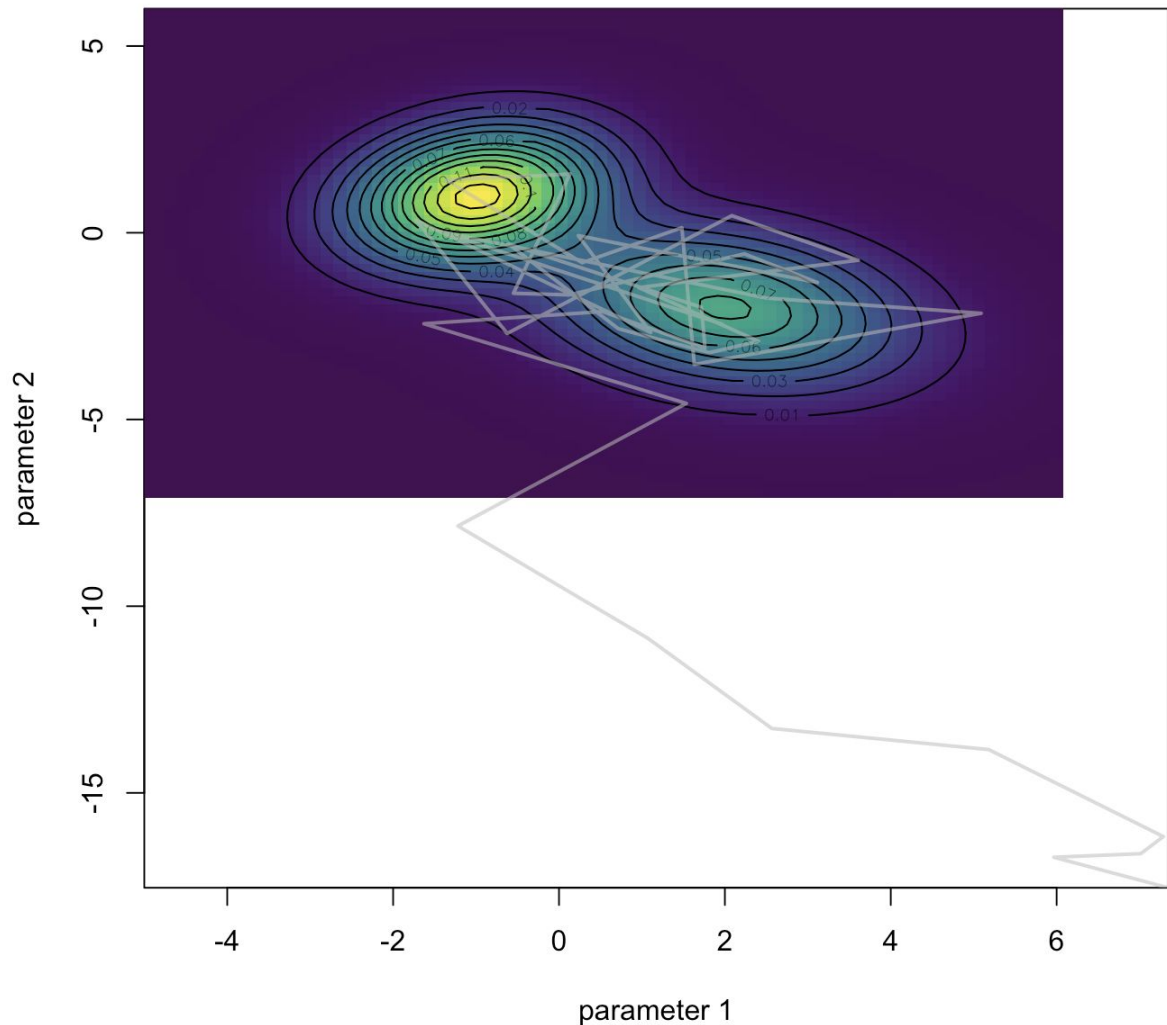
Suppose two parameters,  
with posterior:



# Sample path

We start our parameter values at a random guess

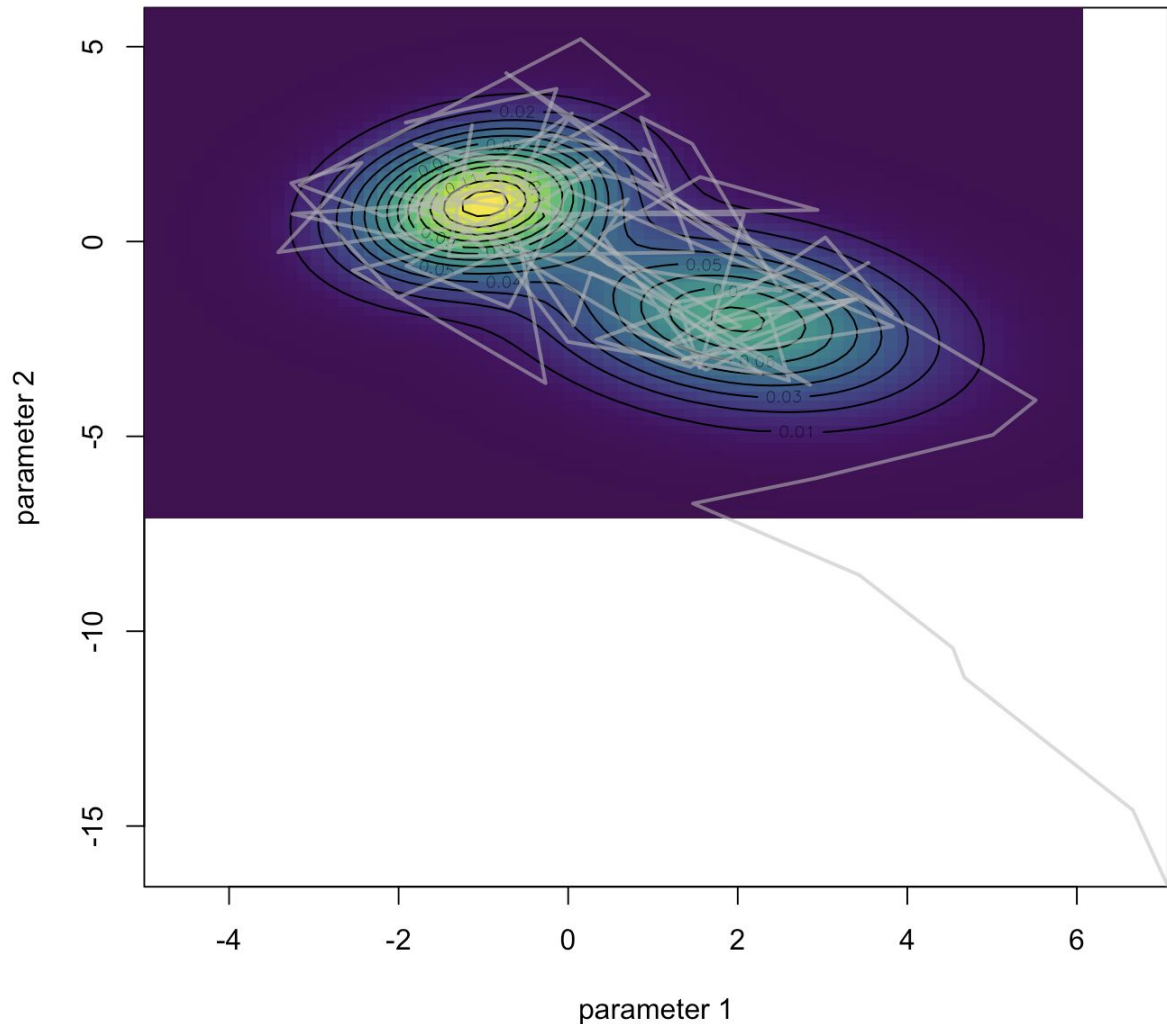
The random walk the MCMC traverses is shown as the grey line



# Sample path

We start our parameter values at a random guess

The random walk the MCMC traverses is shown as the grey line

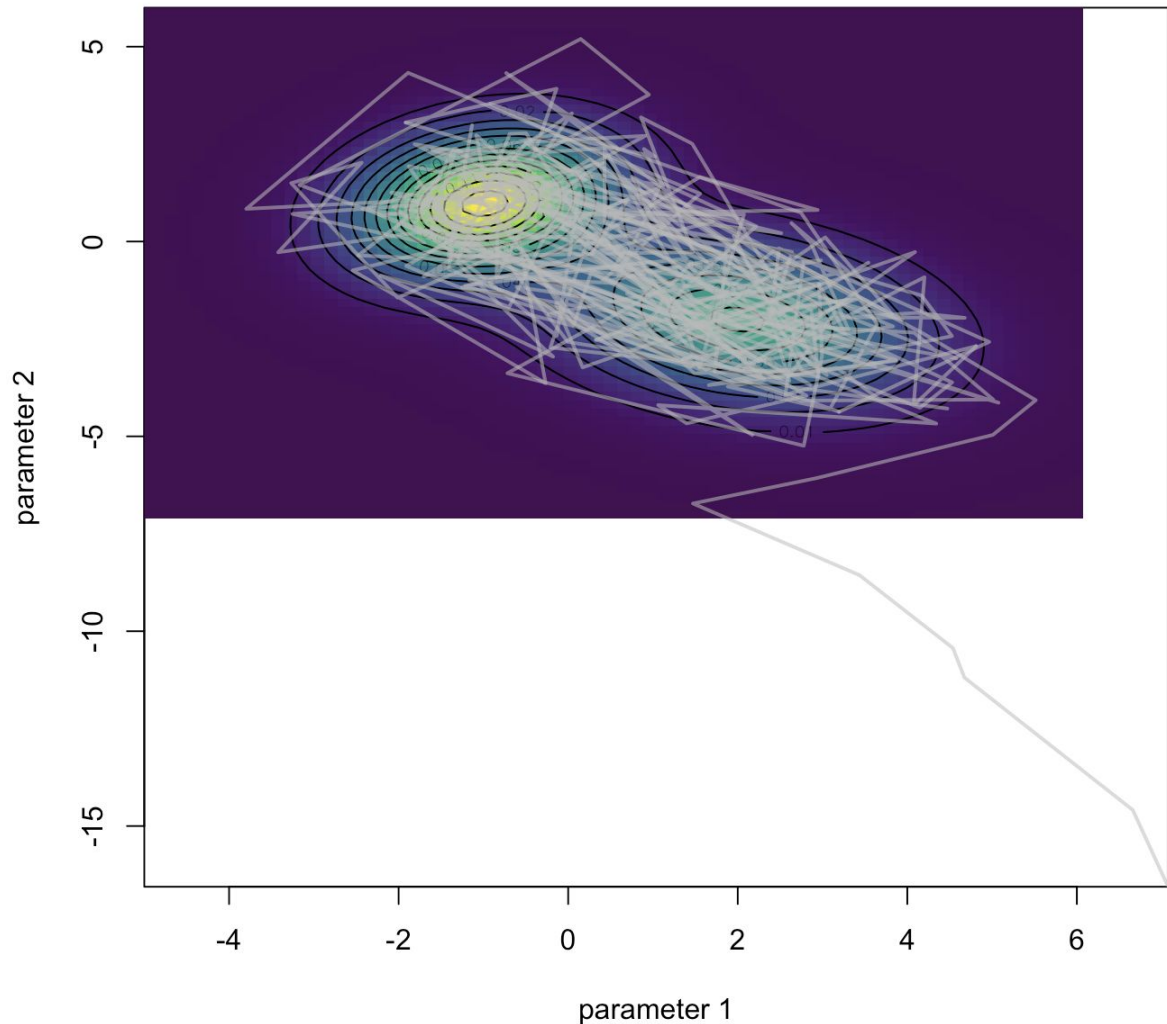




# Sample path

Over time, the random walk accrues samples of the posterior distribution, proportional to the probability of those values

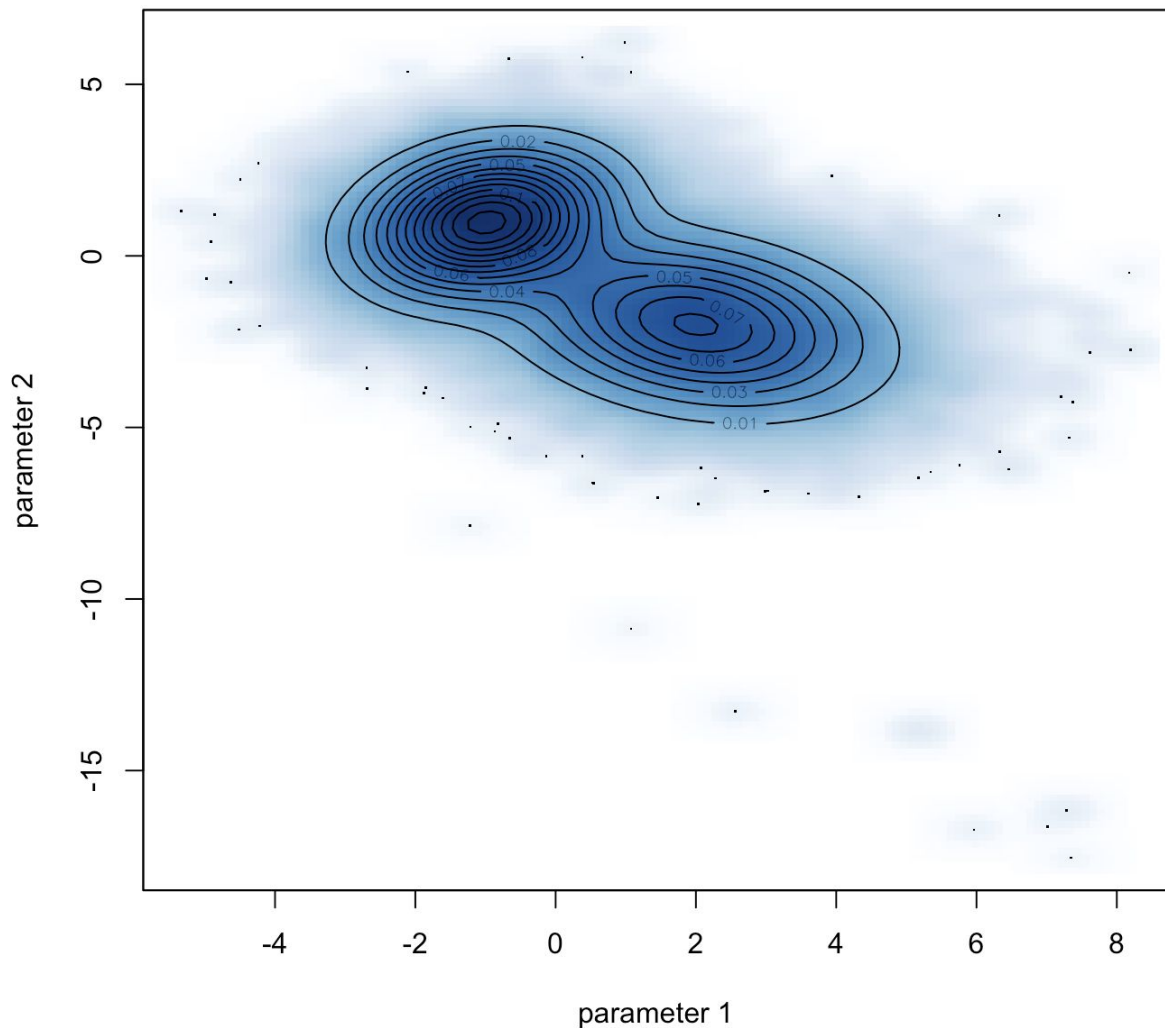
In other words, we get more samples from higher probability regions



# Sampled density

Eventually, the sampled values recreate the posterior distribution!

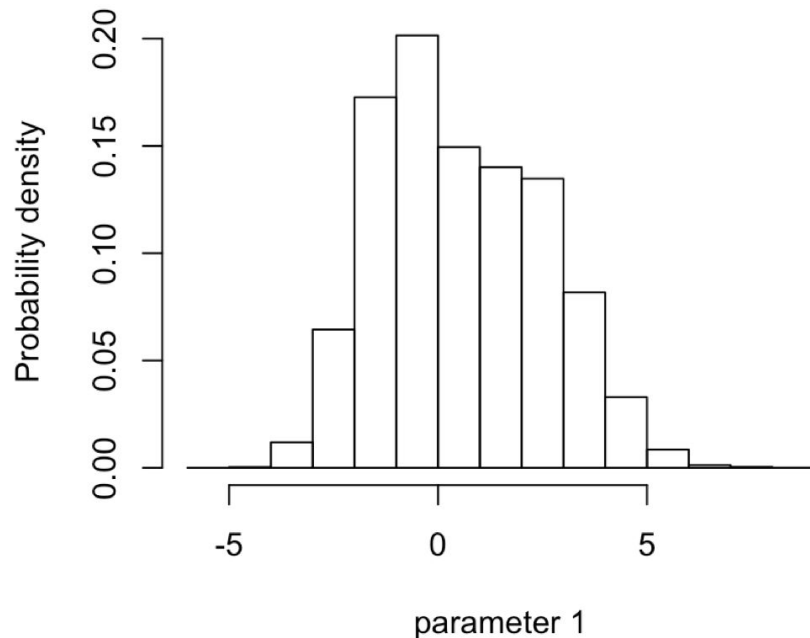
And we didn't need the denominator term, only the numerator term, so these are relatively easy to calculate



# Sampled parameter distributions

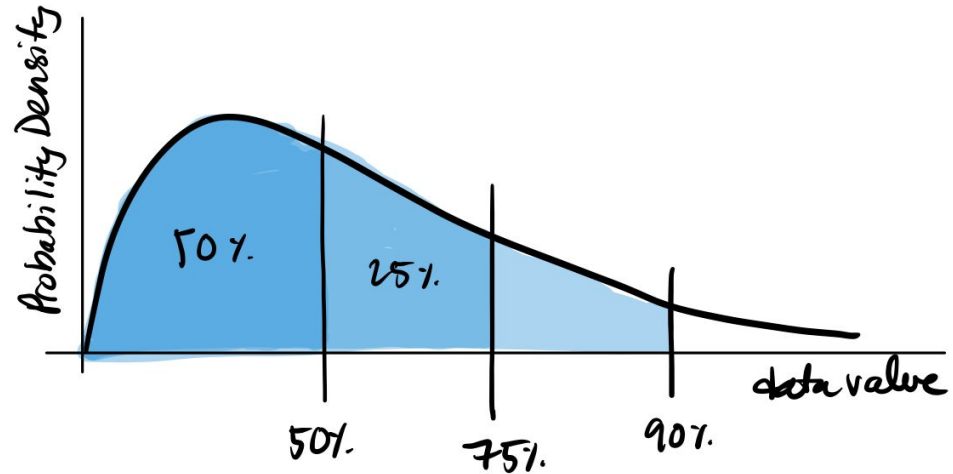
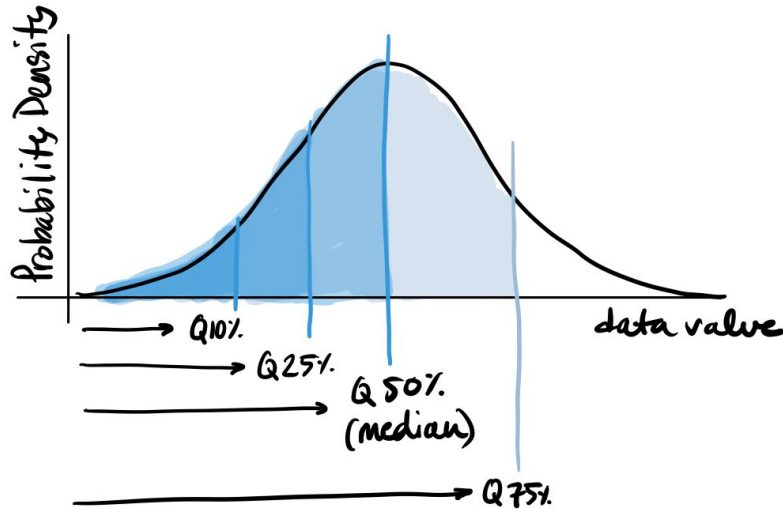
We can use the sampled parameter values (posterior distribution) to examine distributions of the parameters given the data

Use quantiles to calculate uncertainty ranges (confidence intervals or credible intervals, as they're usually called when we're using Bayesian methods)



# Quantiles

**Quantiles** are cut points at which X% of the distribution is below that level (e.g. the 50% quantile is the median!). **Quartiles** are just a special name for the 0%, 25%, 50%, 75% quantiles.



To get uncertainty ranges (e.g. from MCMC), you can use the quantiles on either side—e.g. 90% uncertainty ranges can be generated from the 5% and 95% quantiles

# MCMC isn't perfect either

- MCMC improves many of the problems that other optimization methods face (getting trapped in local minima, etc.)
- However, those issues can still cause problems for MCMC too
- How to know when you've run the MCMC long enough and collected enough samples to reflect the distribution?
- How to know if you have explored the space sufficiently?
- More on this in the Appendix if we have time!

# Examining the goodness-of-fit for a model and data

## **The Eyeball Test!**

(i.e. plot the data and the model predictions on the same graph and see if they look sensible!)

## **Posterior/negative log likelihood/sum of squares/cost function value**

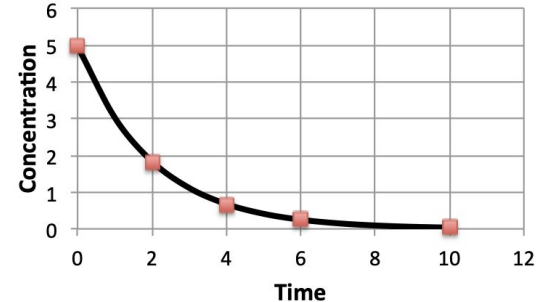
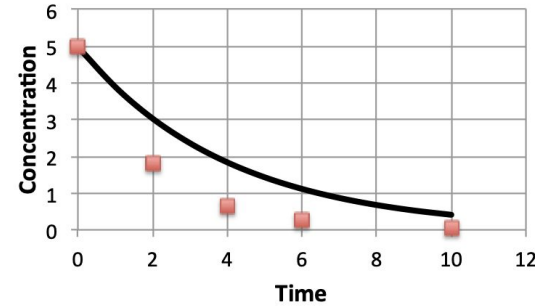
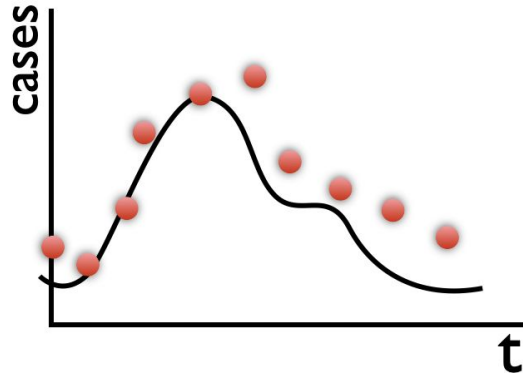
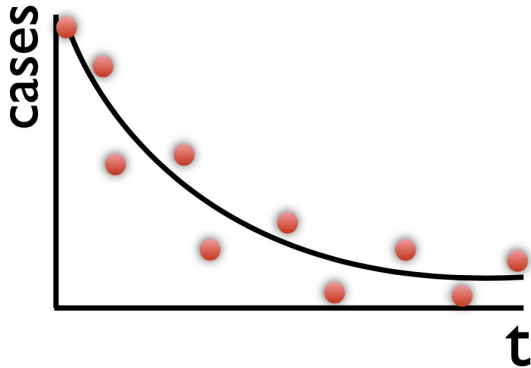
**Parameter uncertainties & correlations**—if you are seeing high uncertainty levels or strongly correlated parameter estimates or uncertainty, there may be unidentifiability problems

**Distribution of residuals** - should make sense based on data assumptions (e.g. if you assumed a normal distribution for your likelihood, do the residuals of your fit look like they match a normal distribution?)

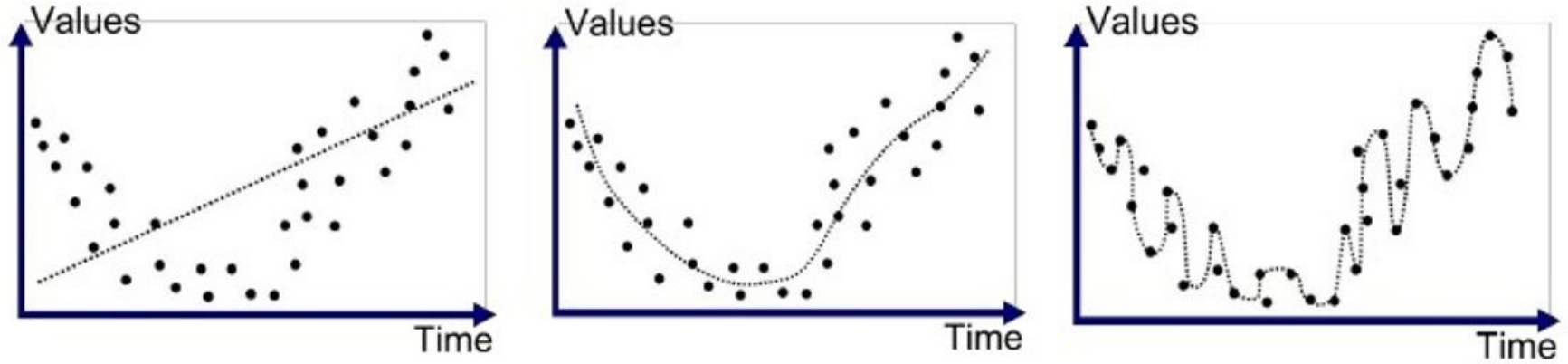
# Examining the goodness-of-fit for a model and data

**Correlation of residuals** (e.g serial correlation coefficient)

**Wald-Wolfowitz Runs Test** - a very fancy name for a simple idea: count how many data points in a row are on one side or the other of your model prediction



# Overfitting





# Model Misspecification

All models are misspecified to some degree

How to detect model misspecification? (Can you always?)

What to do about it?

When testing model goodness of fit, often tend to compare model versus other candidate models

So, how do we compare alternative models?

# Model Comparison

Many methods: F-test, likelihood ratio tests, simply comparing goodness of fit, etc.

One of the most common/popular—**Akaike Information Criterion (AIC)**

# Akaike Information Criterion (AIC)

More parameters means more degrees of freedom, more flexibility in the model

So, we expect models with more parameters may be able to fit data better

Danger of overfitting—need for parsimony

AIC accounts for goodness of fit (likelihood) and overparameterization

$$\begin{aligned} AIC &= -2 \ln(\max(L)) + 2n_P \\ &= 2 \min(-LL) + 2n_P \end{aligned}$$

Where  $n_P$  is the number of parameters we are estimating,  $L$  is the likelihood, and  $LL$  is the log likelihood

Smaller AIC is better (even if negative, i.e. more negative is better)

$AIC \sim -LL + \text{penalty term for parameters}$

Note: AIC only makes sense when fitting to the same data (i.e. different models, same data set)

# AIC

AIC can be derived from information theory - “information loss” when using one model versus another

One AIC has no real meaning by itself— generally need to compare AIC of competing models to say something with it

Many variations of the AIC

Bayesian Information Criterion (BIC) - stronger penalty on parameters

Corrected AIC (cAIC) - correction for small data sets

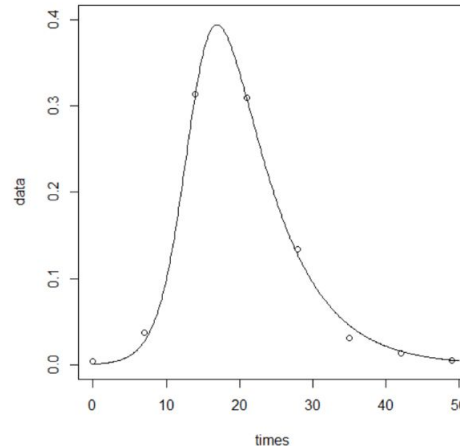
# Word of caution about AICs

A lower AIC just that it gives you the simplest model among those tested that fits well—it may be ‘more’ misspecified in some ways

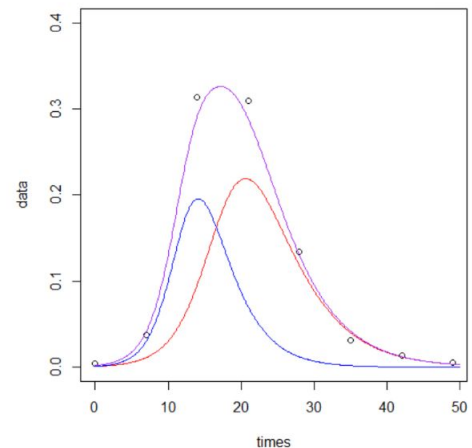
Ex: an epidemic that spreads across 2 towns

**Lower AIC  
because same fit  
but simpler**

**1 pop model**



**2 pop model**



# Some points when you're doing parameter estimation

1. **Be sure visualize the fit of your model!** I.e. plot the model and the data on the same plot and see how good your model is at matching the data
  - a. People will often just run the optimization and then accept the numbers that come out of it—these can be totally meaningless if the fit is bad! (and even if the fit is good...) The likelihood or cost function value often doesn't really tell you how well the model fits the data, you need to look at it and see if it makes sense
  - b. This can be difficult with high dimensional data sets! You may need to do some thinking about how to visualize this, but it's worth it to make sure you do
2. **Be careful about local minima or canyons (unidentifiability)!**
  - a. Optimization algorithms will not always know or warn you that this is a problem—you can be very misled by parameter estimates if this is the case
3. **Think carefully about how you choose your model!**
  - a. Assumptions, overfitting, and model selection
  - b. Try multiple models (and think carefully about how you will choose between them! More on this next week)

# A nice probability distribution cheatsheet

<https://web.cs.elte.hu/~mesti/valszam/kepletek>



## Appendix: Metropolis Algorithm (if extra time)

## Example: Metropolis Algorithm

---

- Idea is to ‘walk’ randomly through parameter space, spending more time in places that are higher probability—that way, the overall distribution draws more from higher probability spots
- Setup—we need
  - A function  $f(p)$  proportional to the distribution we want to sample, in our case  $f(p) = P(z|p) \cdot P(p)$
  - A proposal distribution (how we choose the next point from the current one) - more on this in a minute

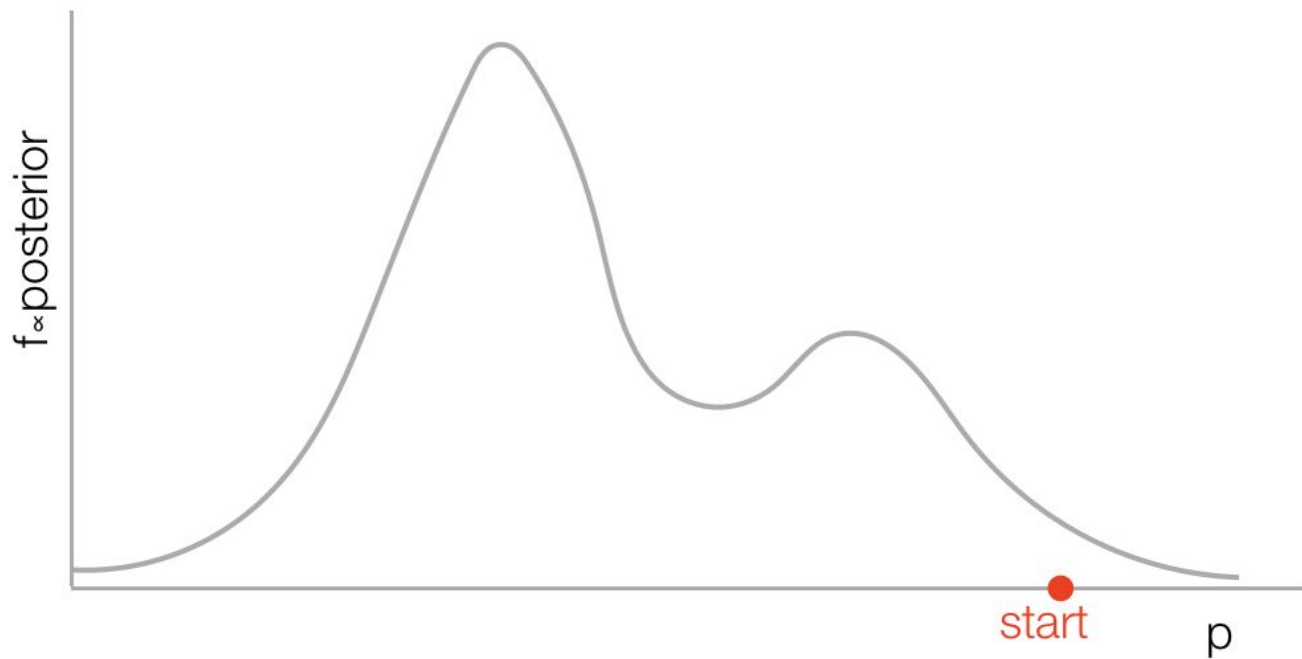
# Metropolis Algorithm

---

- Start at some point in parameter space
- For each iteration
  - Propose a new random point  $p_{next}$  based on the current point  $p_{curr}$  (using the proposal distribution)
  - Calculate the **acceptance ratio**,  $\alpha = f(p_{next})/f(p_{curr})$ 
    - If  $\alpha \geq 1$ , the new point is as good or better—accept
    - If  $\alpha < 1$ , accept with probability  $\alpha$

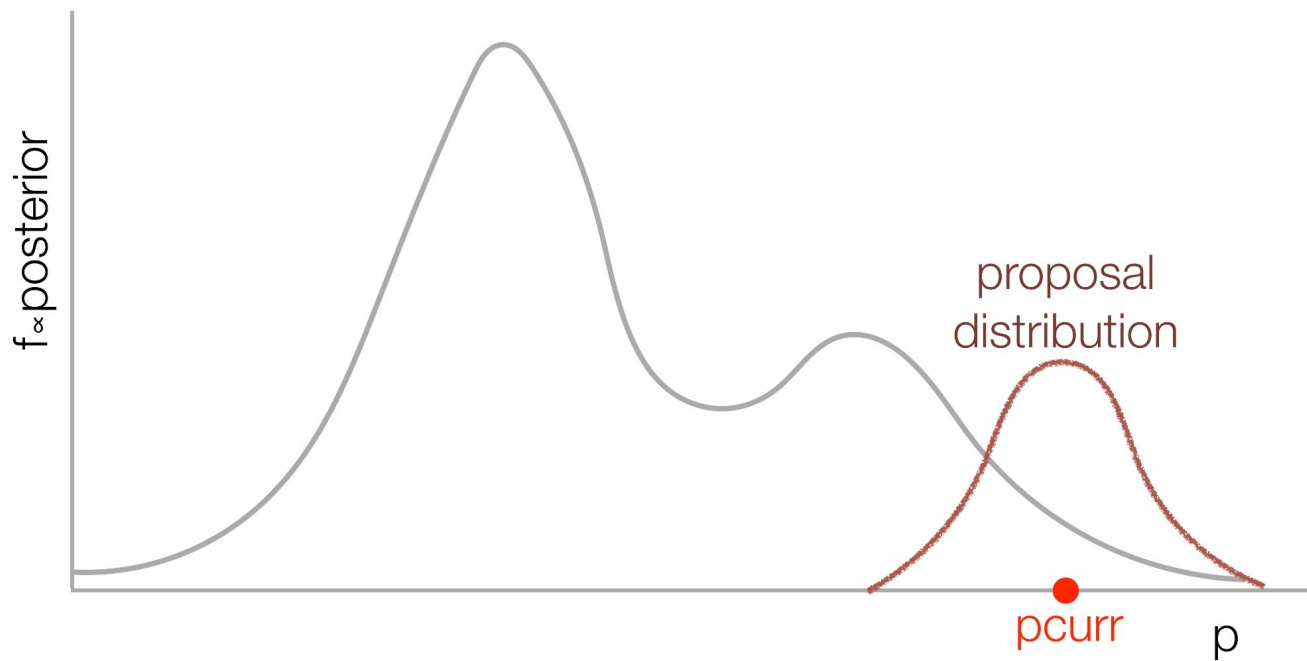
What does the metropolis algorithm do?

---



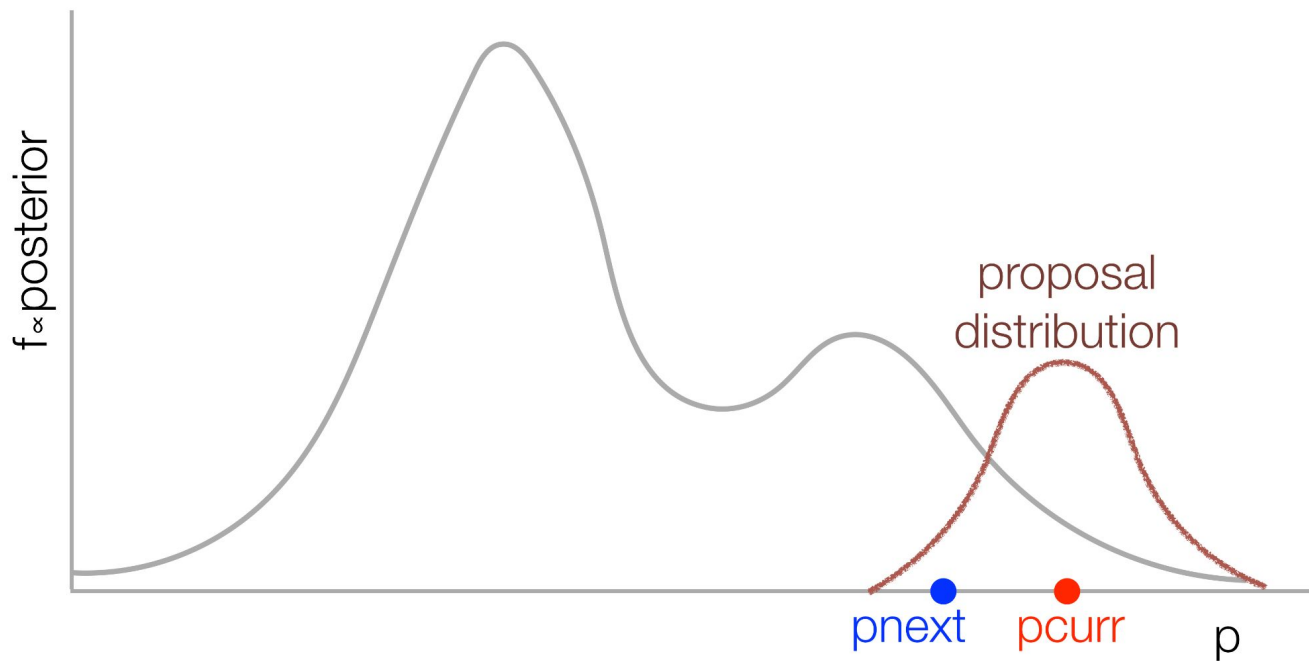
# What does the metropolis algorithm do?

---



# What does the metropolis algorithm do?

---



# What does the metropolis algorithm do?

---

