Math Lecture 6

Modeling and estimation from data

- What is a model?

Modeling & estimation from data

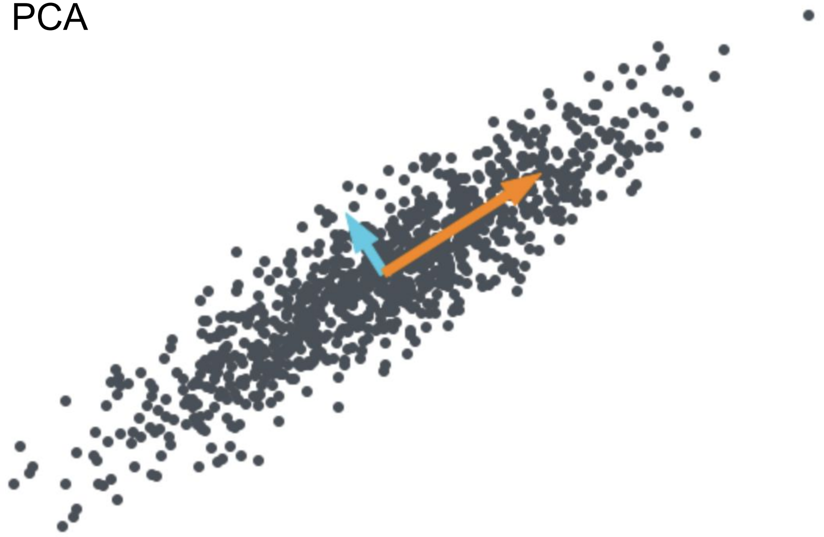
- What is a model?

- A model is a representation of the system we're interested in! Models can be
  - Physical (a diorama for example)
  - Animal (mouse models for testing new drugs)
  - Visual - diagrams, maps (org chart, google maps)

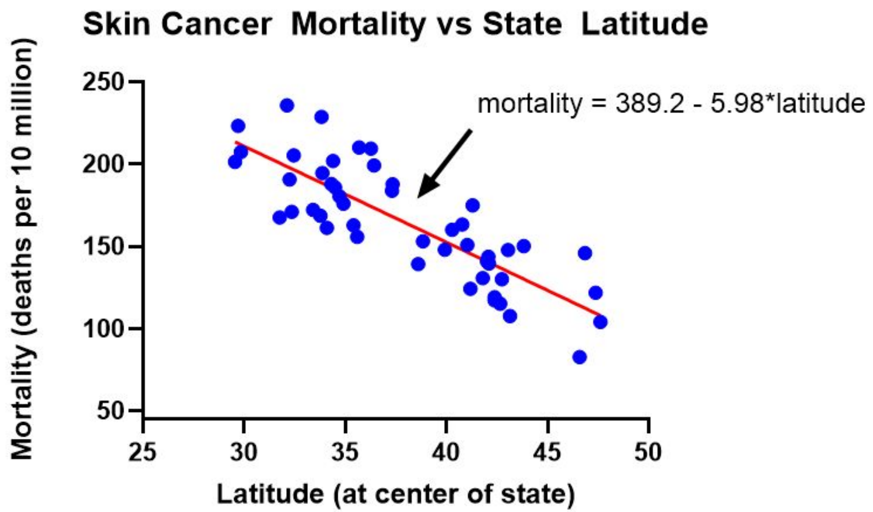
  - **Or they can be built from math and code! (data science)**
  - (And more!)

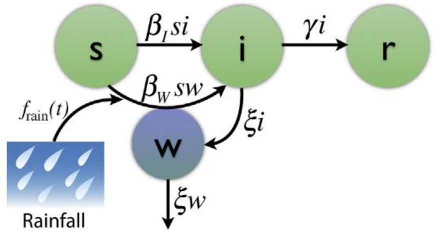
- We use models to represent the underlying structure of data and the mechanisms/processes that generated the data

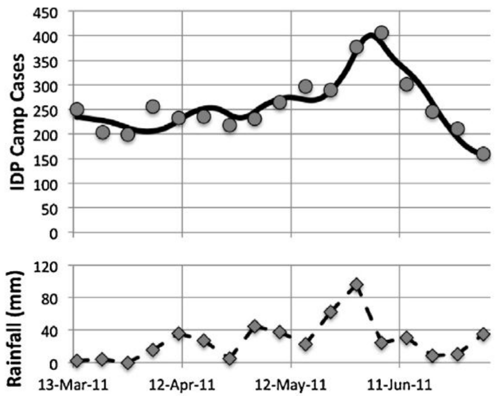
PCA



Linear regression







Transmission model
(in this case for cholera)

Modeling & estimation from data

- Models are abstractions of reality—thus they are in some sense always **wrong** (they necessarily simplify things), but they are also (hopefully) **useful**

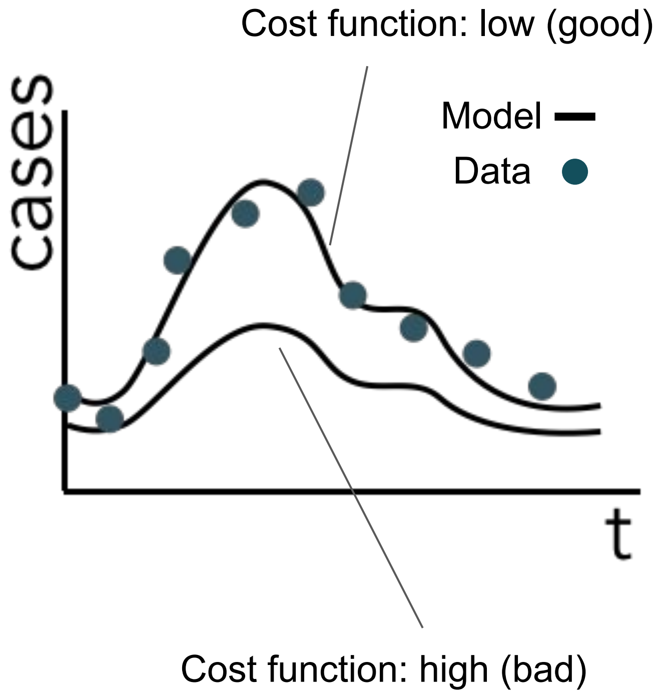
- Almost all statistics and data science approaches involve taking some kind of model, optimizing the parameters of that model to match some data, and then drawing conclusions from the model estimates

Parameter estimation

- **Basic idea/assumption**: parameters that give model behavior that more closely matches data are ‘best’ or ‘most likely’

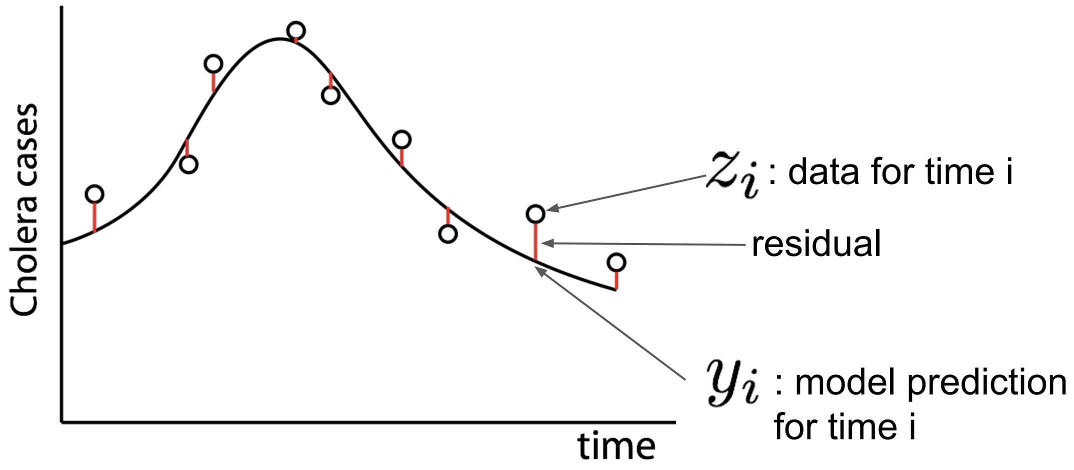
- How to find the ‘closest’ or ‘best’ match?
  - **Cost or objective function**: a way to say how close the model is to the data
  - **Optimization**: a way to adjust the model to get the best match, i.e. to minimize the cost function

- We want to frame this idea from a **statistical perspective** (inference, regression)
  - Can determine ‘most likely’ parameters or distribution, confidence intervals, etc.



Least squares - one of the most commonly used forms of parameter estimation and optimization

- Goal: adjust parameters to match the data as closely as possible
- Residual: distance between model and data at a given time point



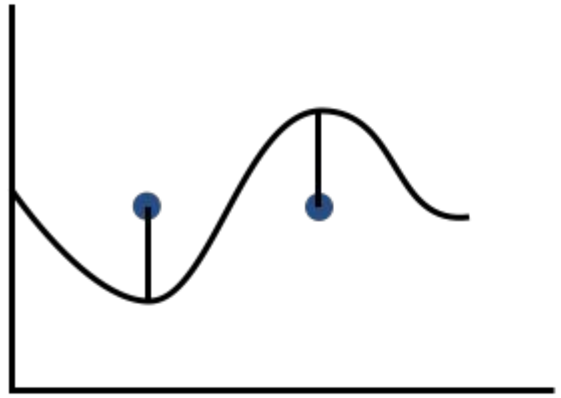
z_i : data for time i

y_i : model prediction for time i

Least squares estimation

- Minimize residuals? Problem is, sign issues

$$\min\left(\sum_{i=1}^{n} z_i - y_i\right)$$



$\longrightarrow$

Sum of residuals:
+1 - 1 = 0
even though model is far from data

Least squares estimation

z_i : data for time i

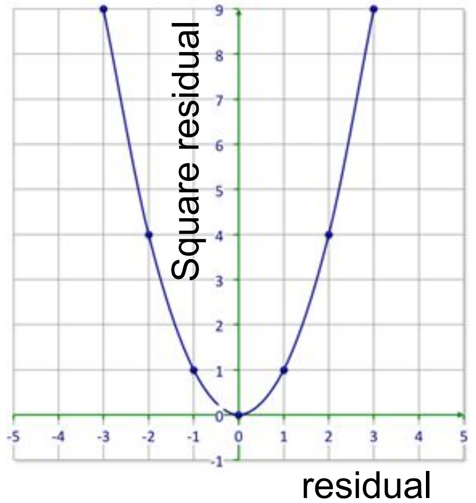
y_i : model prediction for time i

- Least Squares: Minimize square of residuals

$$\min\left(\sum_{i=1}^{n}(z_i - y_i))^2\right)$$

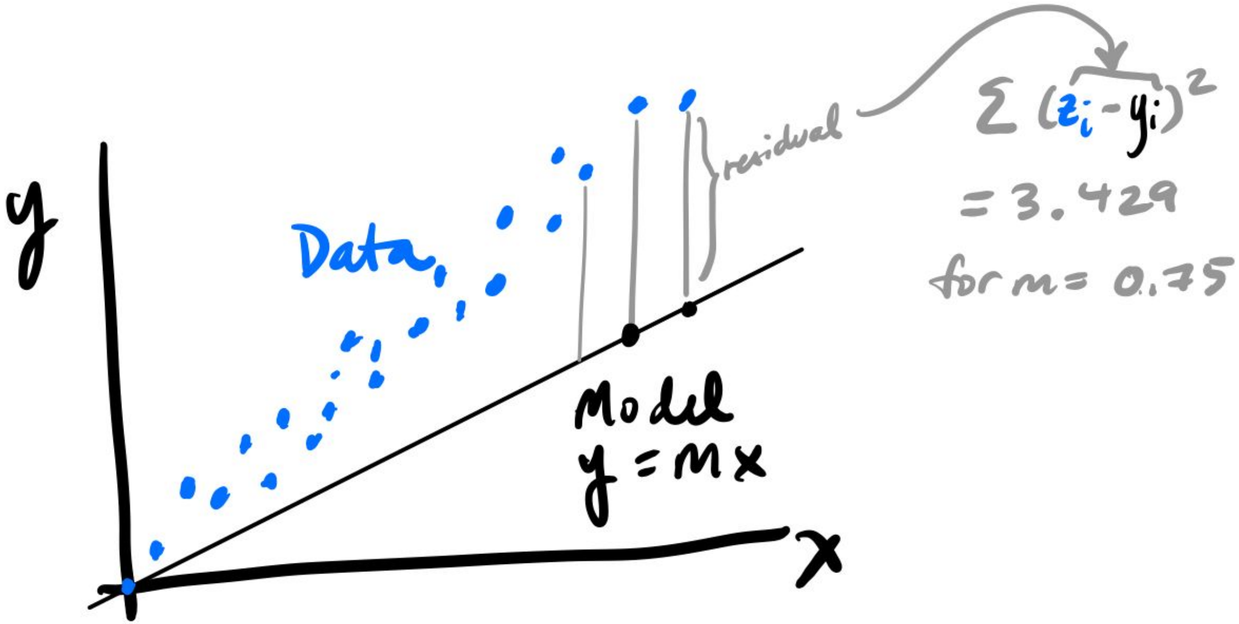
- All errors have same sign in summation

- Penalizes large errors—square term is less than one for small residuals (|zi-yi| < 1) but bigger than 1 for big residuals (|zi-yi| > 1)



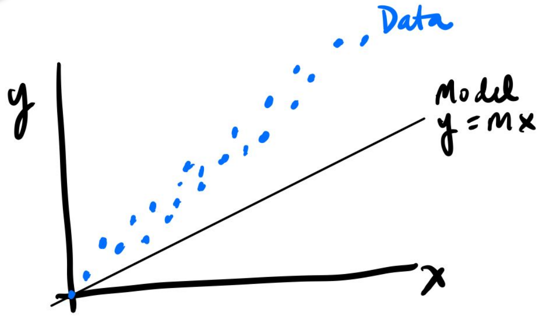
Mini Example

- Model: y = mx
  - 1 parameter, m
- Cost function f(m)
  - Sum of squares
- Initial guess: m = 0.75
- Cost function: 3.429

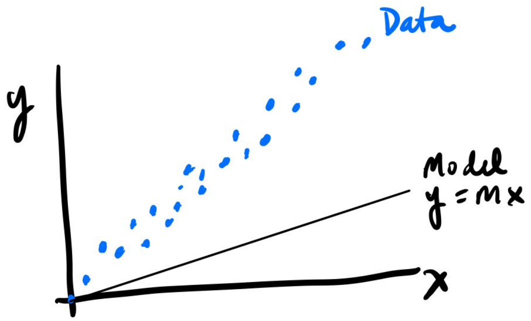


1 m = 0.75, Cost = 3.429

Start

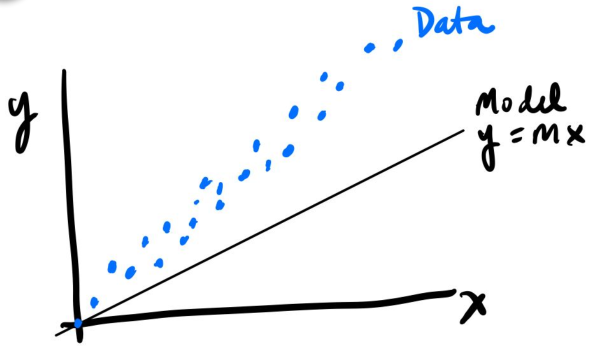


2 m = 0.5, Cost = 3.708

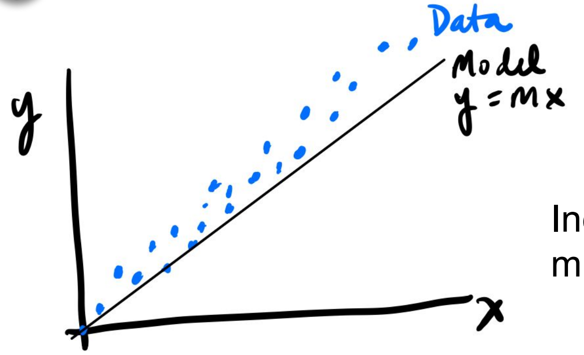


Worse!
Let's go back

3 m = 0.75, Cost = 3.429

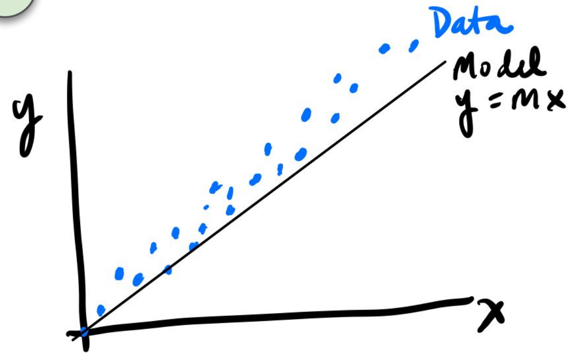


4 m = 1, Cost = 1.654

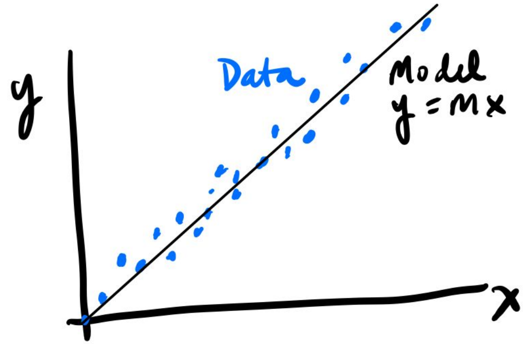


Increasing
m is better

4 m = 1, Cost = 1.654

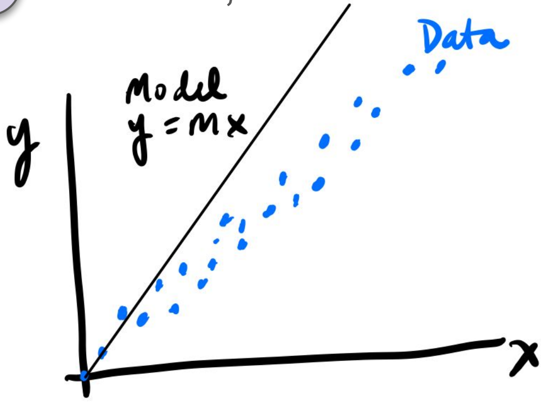


5 m = 1.2, Cost = 0.0016



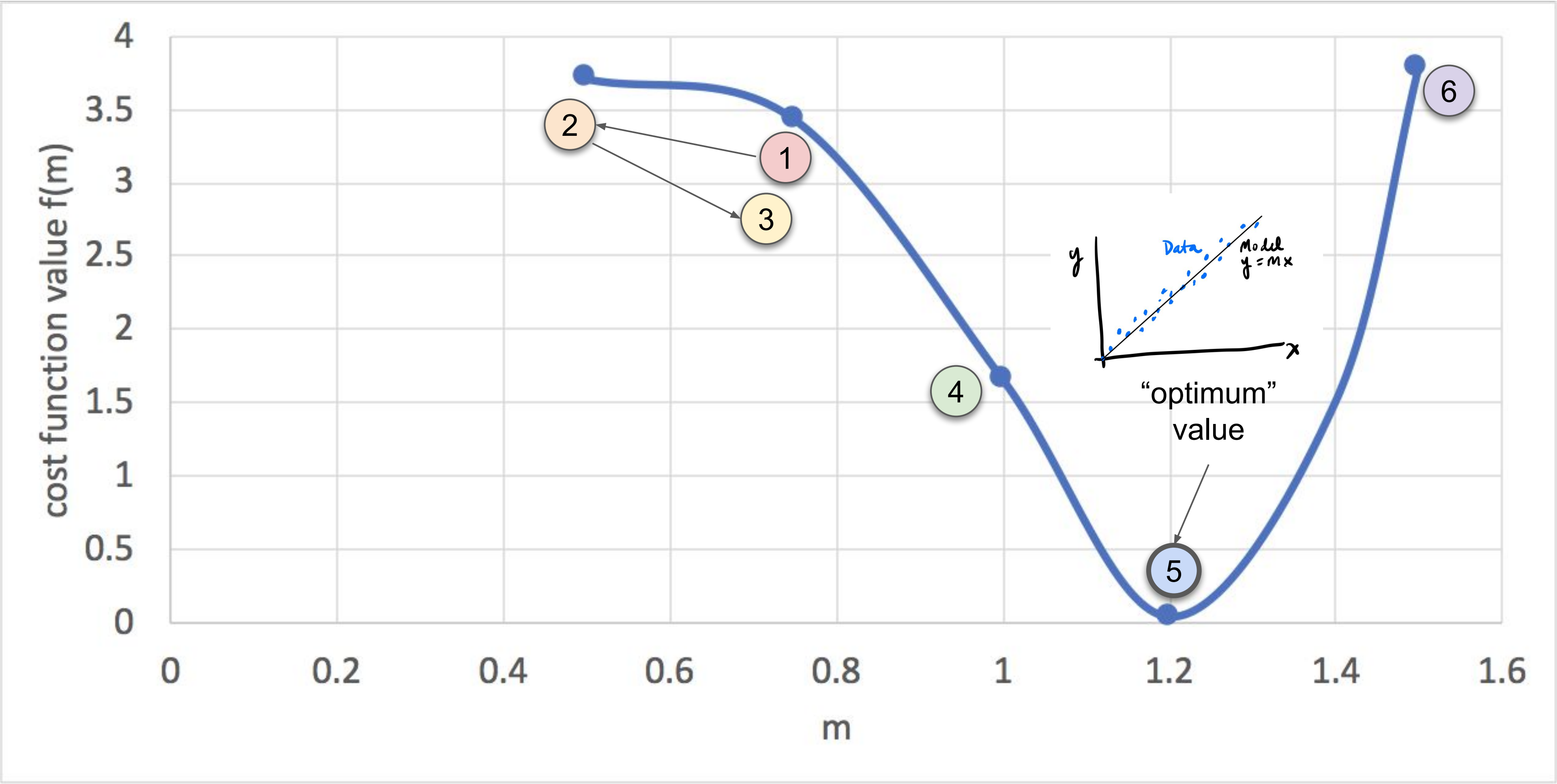
This looks pretty good! Let's try a little further

6 m = 1.5, Cost = 3.781



Went too far!

Okay, so m = 1.2 seems the best of the guesses we've tried! Let's plot our progress


cost function value f(m)
4
3.5
3
2.5
2
1.5
1
0.5
0
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
m
1
2
3
4
5
6
y
x
Data
Model
y = mx
"optimum"
value

Okay, but guessing and checking is probably not the best way to do this…

Optimization

Basic idea: use mathematical and computational methods to make something optimal (best)

In our case, usually what we mean is to choose a function or formula that represents what we want to optimize (e.g. how close or far is our model from the data, or how well a control strategy is working, or the actual cost of something), and find the minimum or maximum value!

This function is called our **cost or objective function**

Vocabulary

- **Objective function or cost function, f(x)**: the function we are trying to minimize or maximize. Can be a function of a range of different variables!
  - Usually a function of our data and the parameters and variables of our model
  - Historically, *convention is to minimize f(x)* (note if you want to maximize just minimize -f(x) !)
- **Variables/inputs/parameters**: inputs to the cost function that we can adjust/change/control
- **Constraints**: any restrictions to our optimization (e.g. find the biggest number, but our constraints are that it has to be even and less than 11)

In a data science context:

Goodness-of-fit

Model parameters

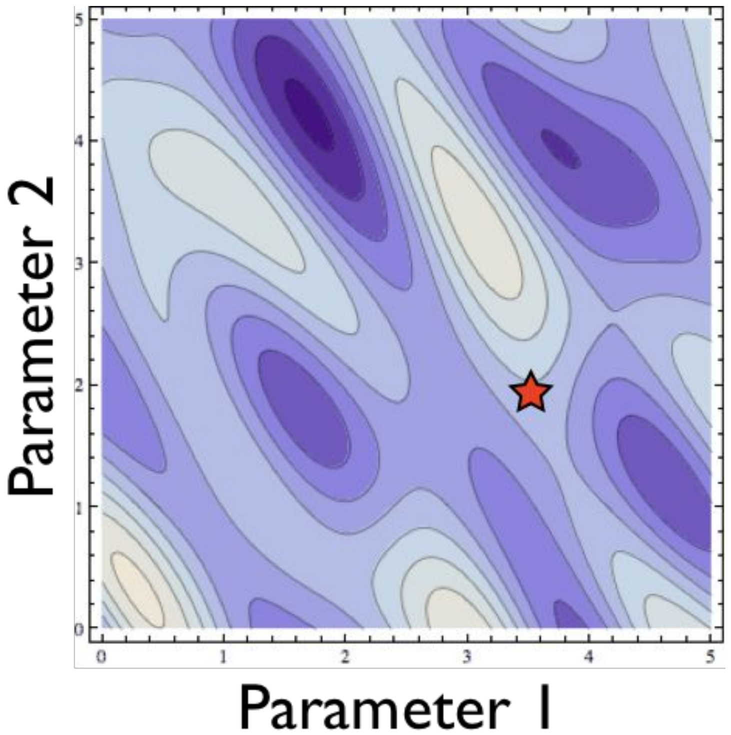
Restrictions on parameters or fit

Optimization methods

Many methods do something like:

- Take in a set of starting parameter values (initial values)
- Evaluate the nearby cost function landscape (i.e. what are nearby cost function values)
- Pick a new set of parameter values
- Repeat until no further improvement can be made, or enough parameter values have been sampled

Color = Cost function

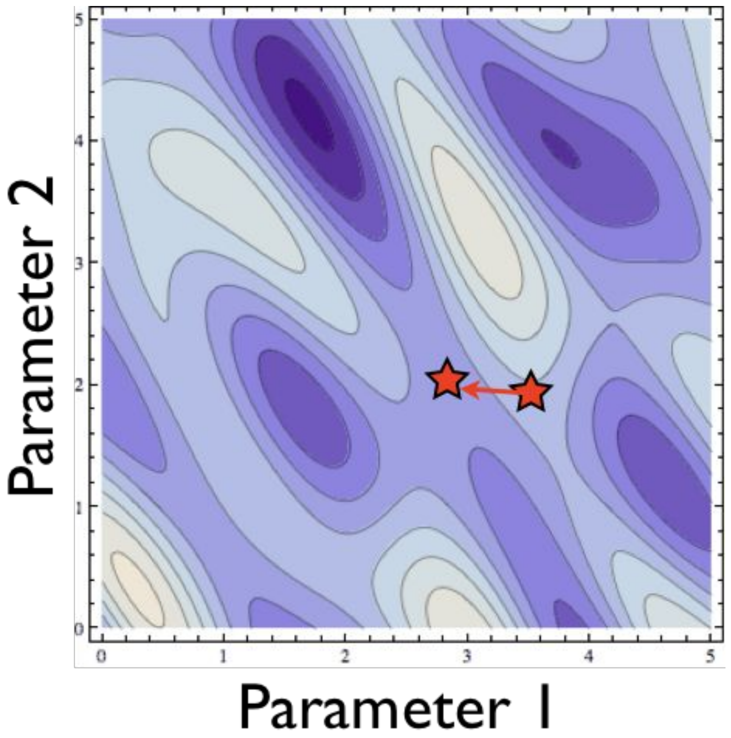


Optimization methods

Many methods do something like:

- Take in a set of starting parameter values (initial values)
- Evaluate the nearby cost function landscape (i.e. what are nearby cost function values)
- Pick a new set of parameter values
- Repeat until no further improvement can be made, or enough parameter values have been sampled

Color = Cost function

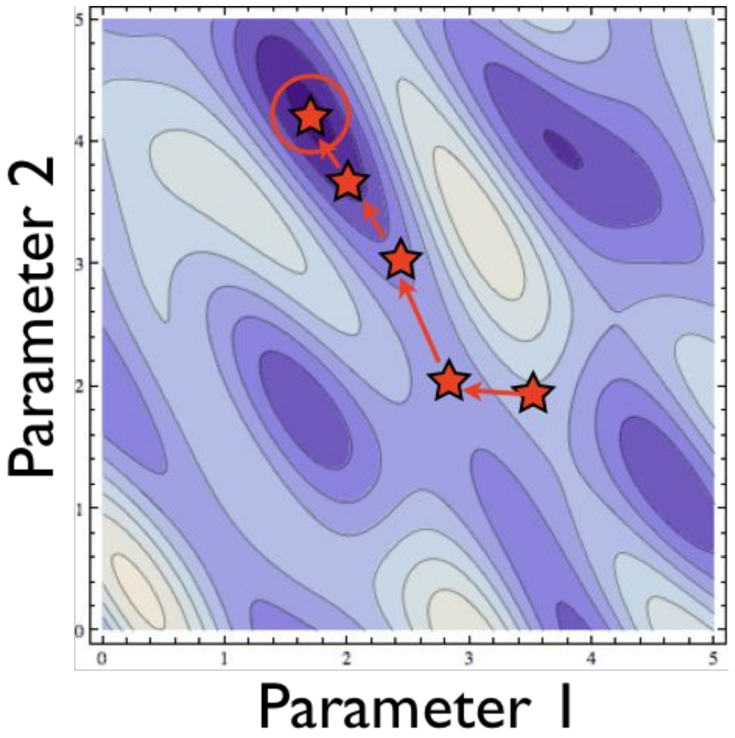


Optimization methods

Many methods do something like:

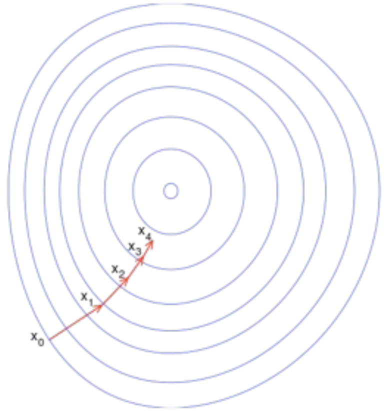
- Take in a set of starting parameter values (initial values)
- Evaluate the nearby cost function landscape (i.e. what are nearby cost function values)
- Pick a new set of parameter values
- Repeat the previous two steps until no further improvement can be made, or enough parameter values have been sampled

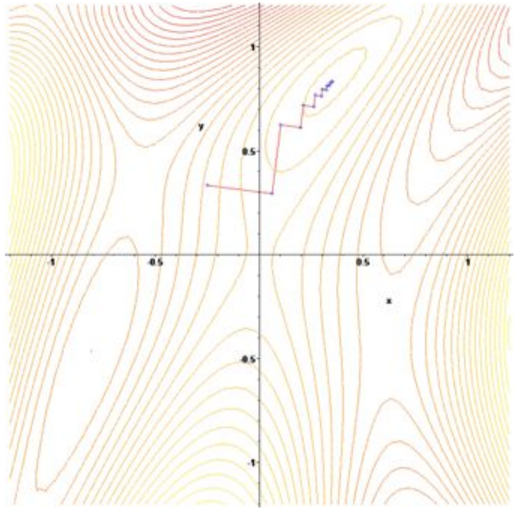
Color = Cost function



Optimization methods: gradient descent

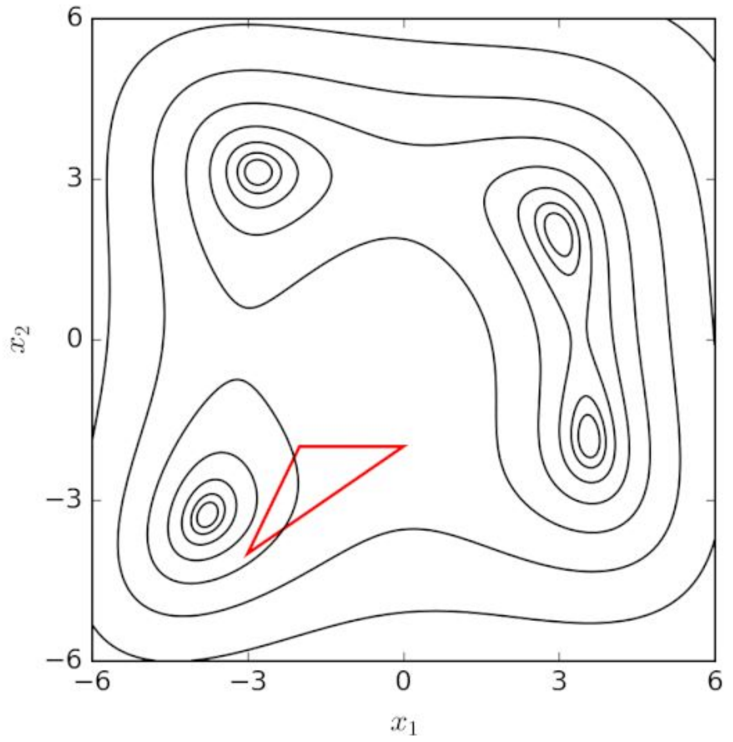
- Fast!
- Relatively easy to use and implement
- Can get stuck in local minima
- Some surfaces can cause weird/problematic behavior (zig-zag issue, soft serve ice cream example)





Optimization methods: simplex algorithms

- Can be similar to gradient search
    - Easy to implement
    - Fast, efficient to compute
- Gets around some of the issues of gradient descent, but can still get caught in local minima
- E.g. Nelder-Mead algorithm

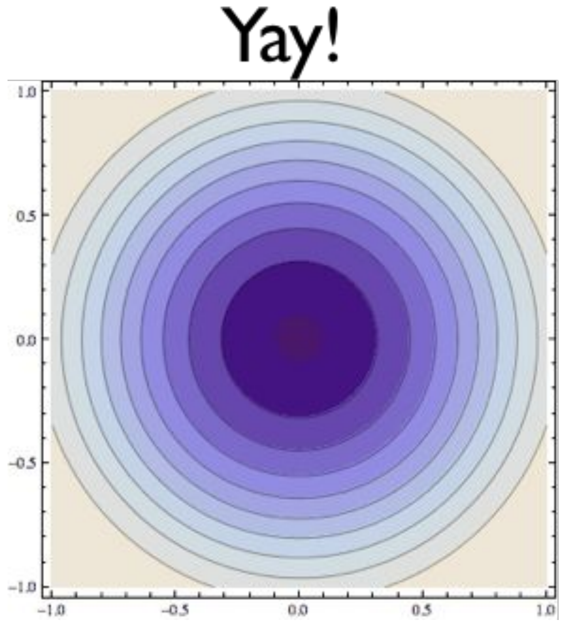


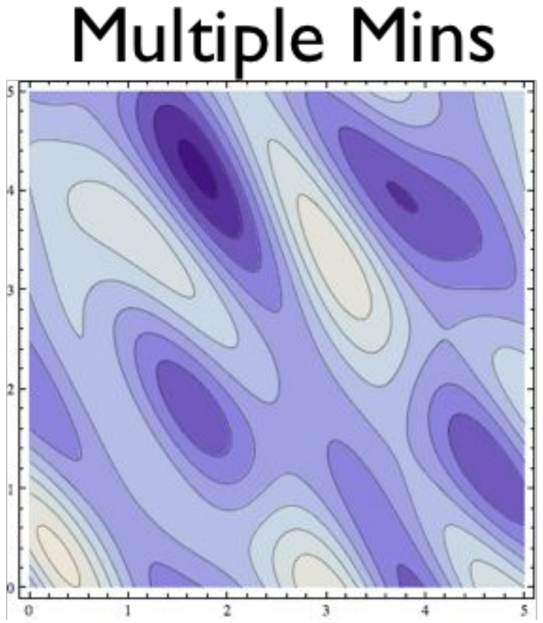
Global optimization methods

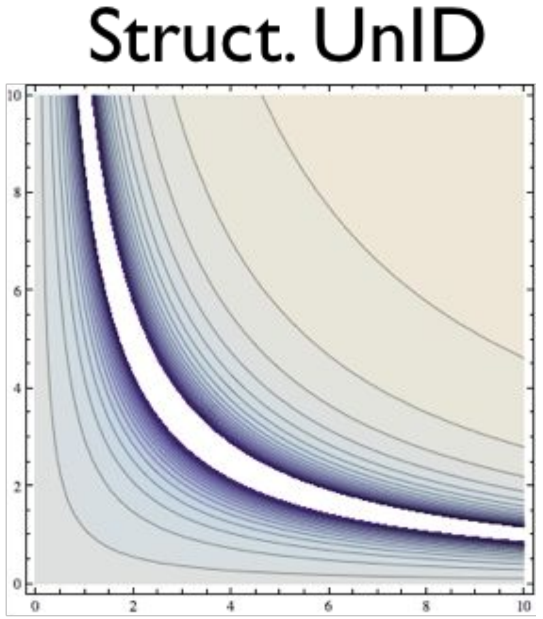
- Many options! Simulated annealing, evolutionary algorithms, etc
    - Markov Chain Monte Carlo (MCMC) methods can also be considered in this category

- These approaches typically allow for some acceptance of worse cost function values, to allow the optimizer to escape local minima
- Also often involve examining many different initial parameter values or many parameter samples, to more fully explore the cost function surface
- Gives a better view of the cost function surface—but often very(!) slow compared to local optimization methods (gradient descent, simplex methods, etc)

Optimization methods are often formulated for convex, single minimum surfaces—multiple minima and canyons (sometimes called unidentifiability) can cause problems







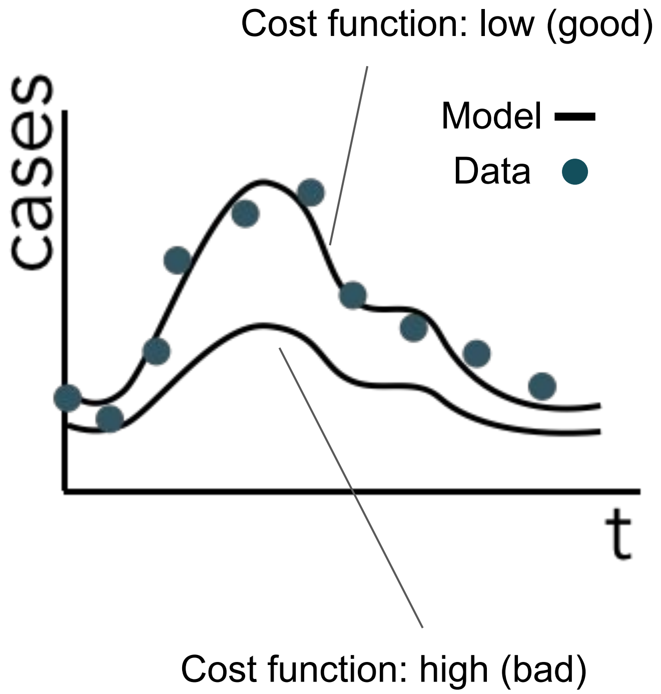
Code examples parts 1 and 2

Parameter estimation

- **Basic idea/assumption**: parameters that give model behavior that more closely matches data are ‘best’ or ‘most likely’

- How to find the ‘closest’ or ‘best’ match?
  - **Cost or objective function**: a way to say how close the model is to the data
  - **Optimization**: a way to adjust the model to get the best match, i.e. to minimize the cost function

- We want to frame this idea from a **statistical perspective** (inference, regression)
  - Can determine ‘most likely’ parameters or distribution, confidence intervals, etc.



How to frame this statistically?

- Maximum Likelihood

- **Basic idea**
  - View our model as a probability distribution, where we suppose we know the general form of the probability density function but not the parameter values
  - Then figure out which parameter values make the model most likely to generate the data we see

Maximum Likelihood

Given the parameters, we can usually work out the probability of observing a particular data set $P(z \mid p)$

For example, suppose we have a potentially biased coin, and we want to estimate the probability that a coin flip results in heads—call this parameter p.

- Our data (call this z): we measure 3 coin flips: H, T, H.
- What's the probability that p = 0.5 given this data? Hard to say!
- But, if we knew p, we can definitely calculate the probability of seeing this data. Assume the coin flips are independent, then:

$$P(z|p) = p*(1-p)*p$$

This P(z|p) is the likelihood! The main idea of maximum likelihood is to choose p to maximize this

Maximum likelihood estimate?

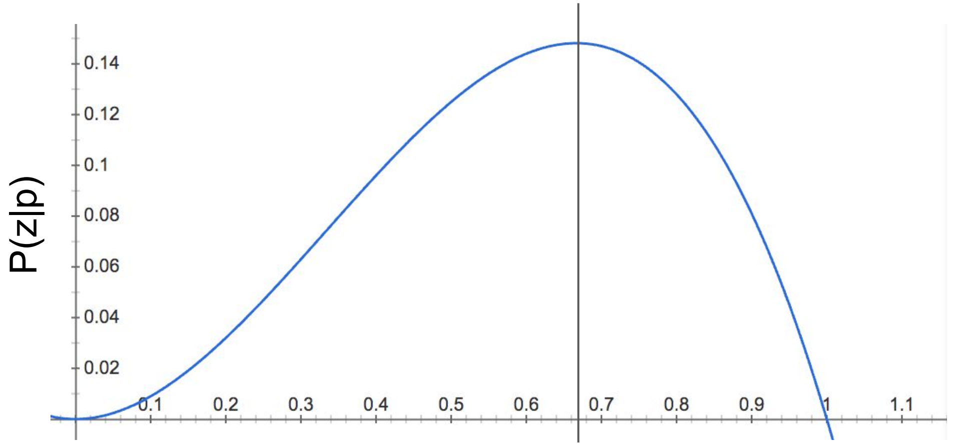
Data (z) is 3 coin flips: H, T, H
Likelihood function: P(z|p) = p*(1-p)*p

What value of p maximizes this?

$P(z|p) = p*(1-p)*p = p^2 - p^3$

(Note this is actually a constrained optimization problem—p must be between 0 and 1)

We could code it up—but we can also just plot it!



Maximum likelihood estimate!
p = 2/3

p

Maximum likelihood

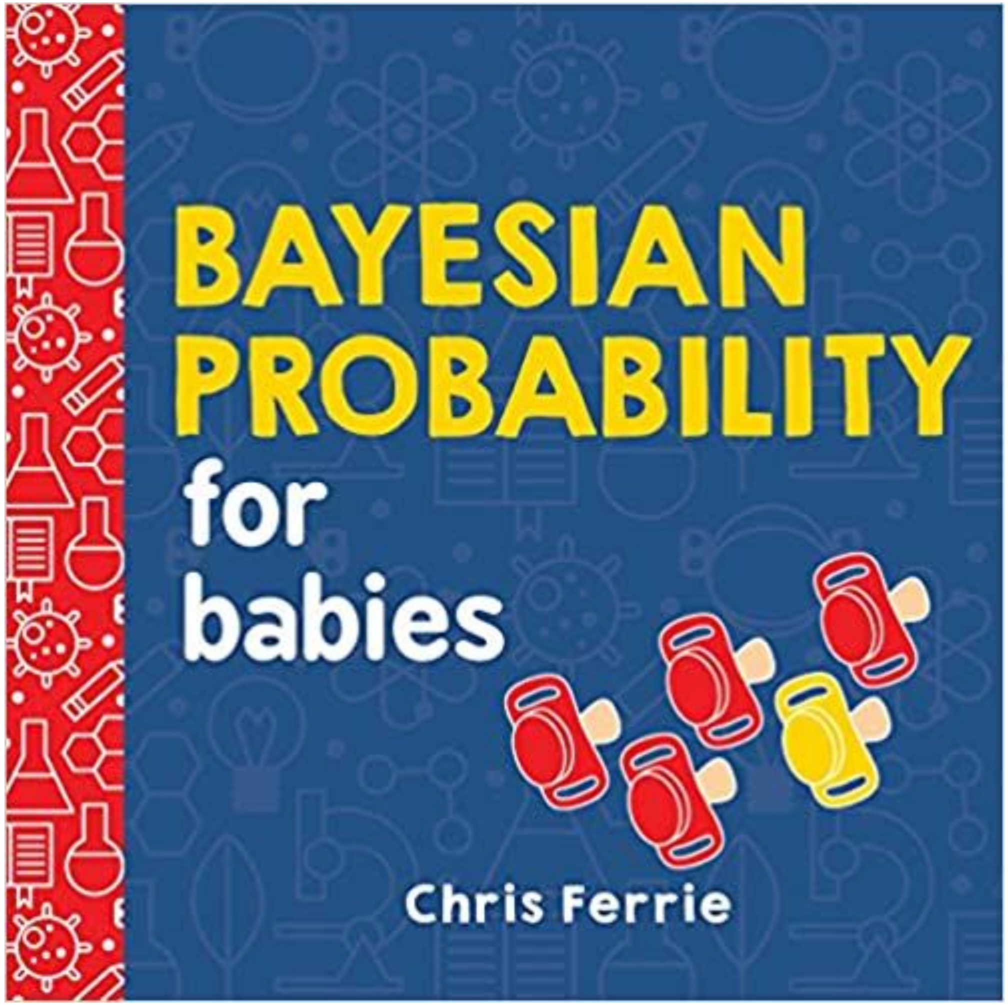
Likelihood function: the probability of seeing the data we have, assuming that we knew the parameter values.

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

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

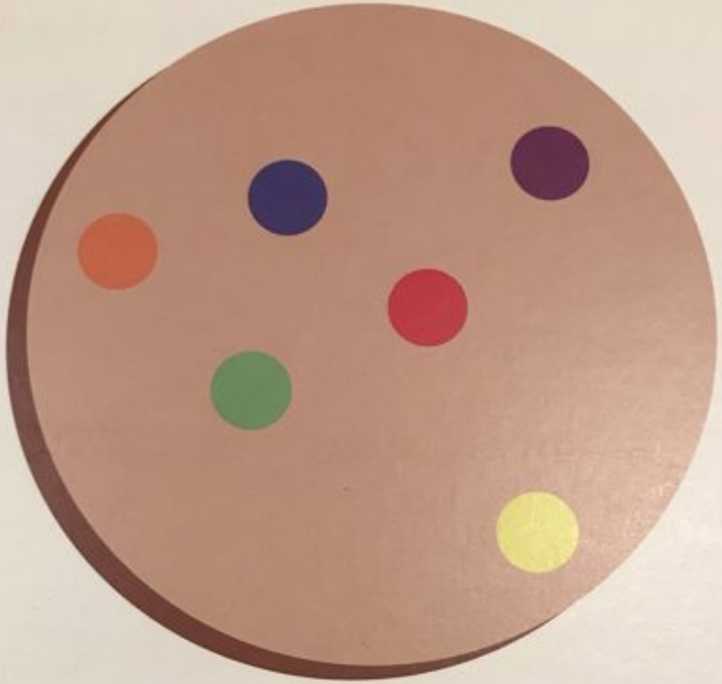
E.g. normal distribution: $f(z \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{(z-\mu)^2}{2\sigma^2} \right) = L(\mu, \sigma^2 \mid z)$

Find the value of p that maximizes L(p|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!)

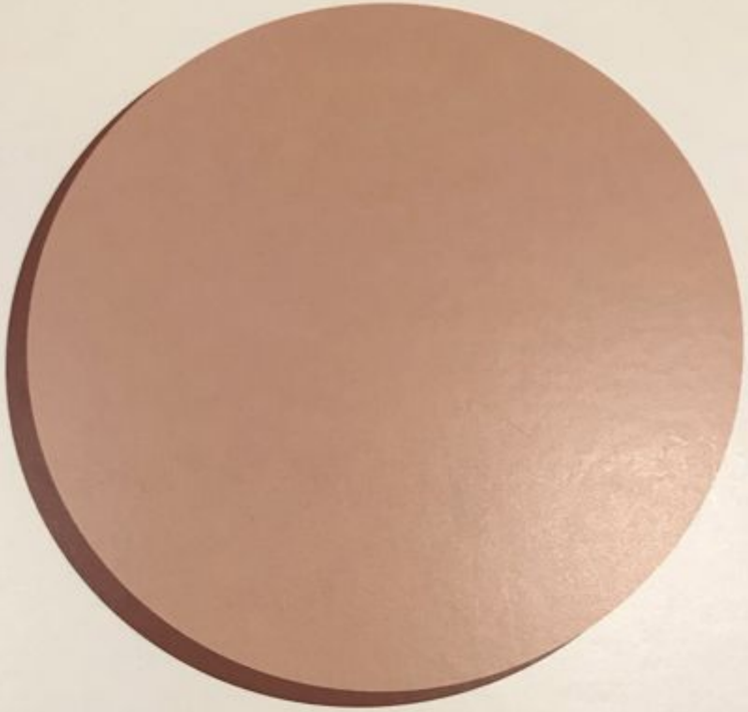


BAYESIAN PROBABILITY for babies

Chris Ferrie



Some cookies have candy.



Some don't.

Our data:

Take a bite. It has no candy.

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

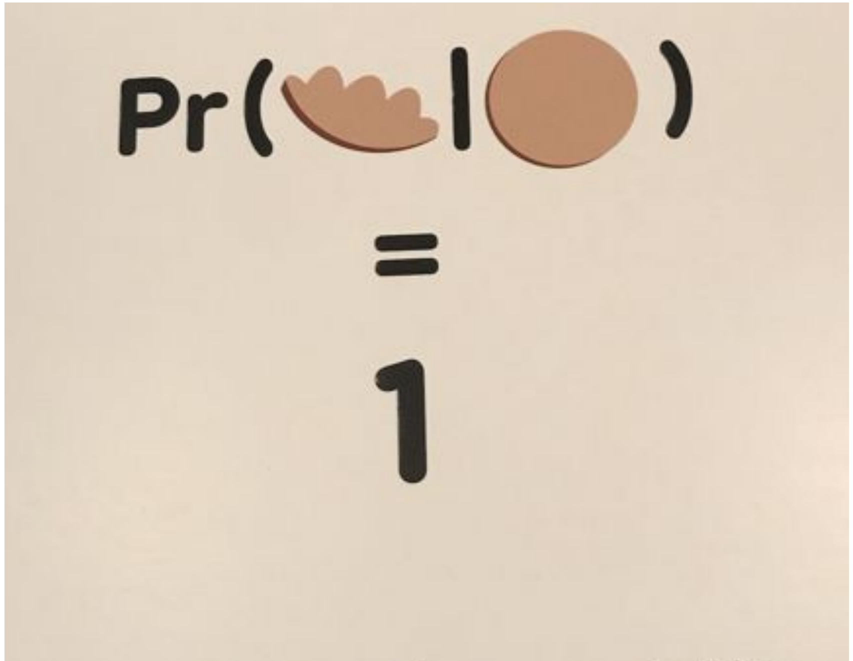
In other words, our parameter estimation problem is to estimate c!

Does c = 0 or 1?

What is the likelihood?

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

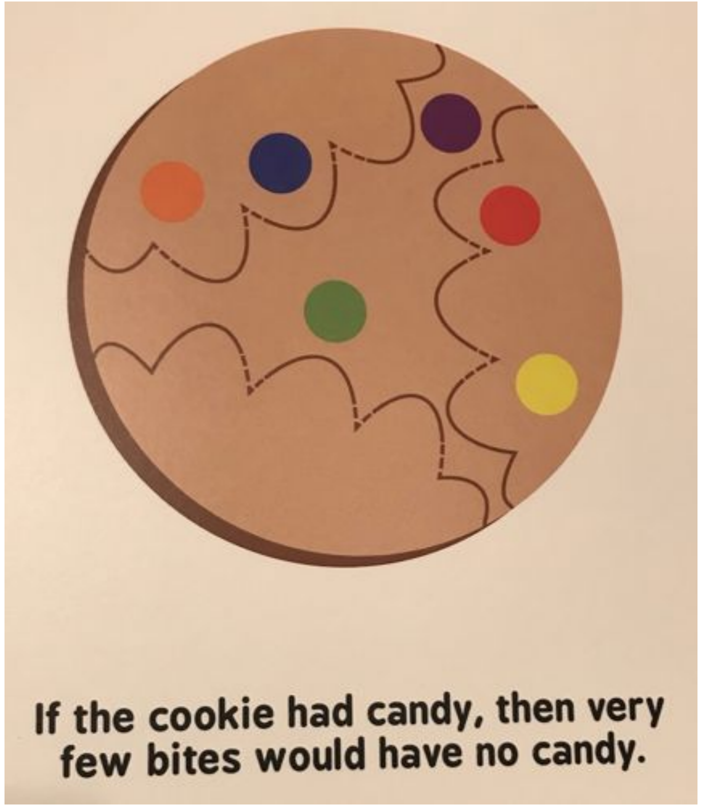
If the cookie had no candy,
then every bite would have no candy.



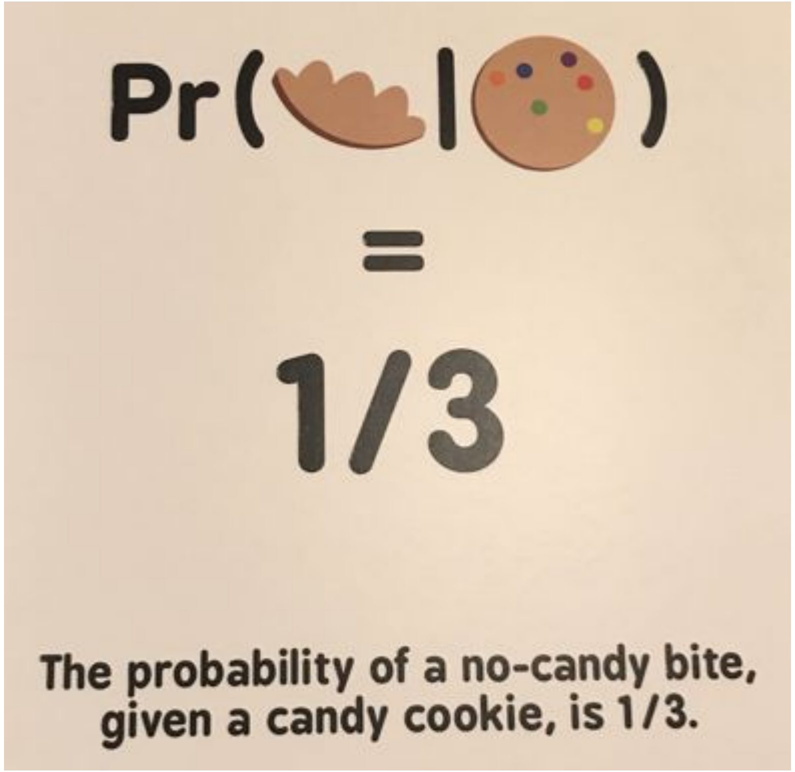
The probability of a no-candy bite,
given a no-candy cookie, is 1.

What is the likelihood?

L(c = 1 | NC bite) = P(NC bite | c = 1) = 1/3



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



Pr(|)

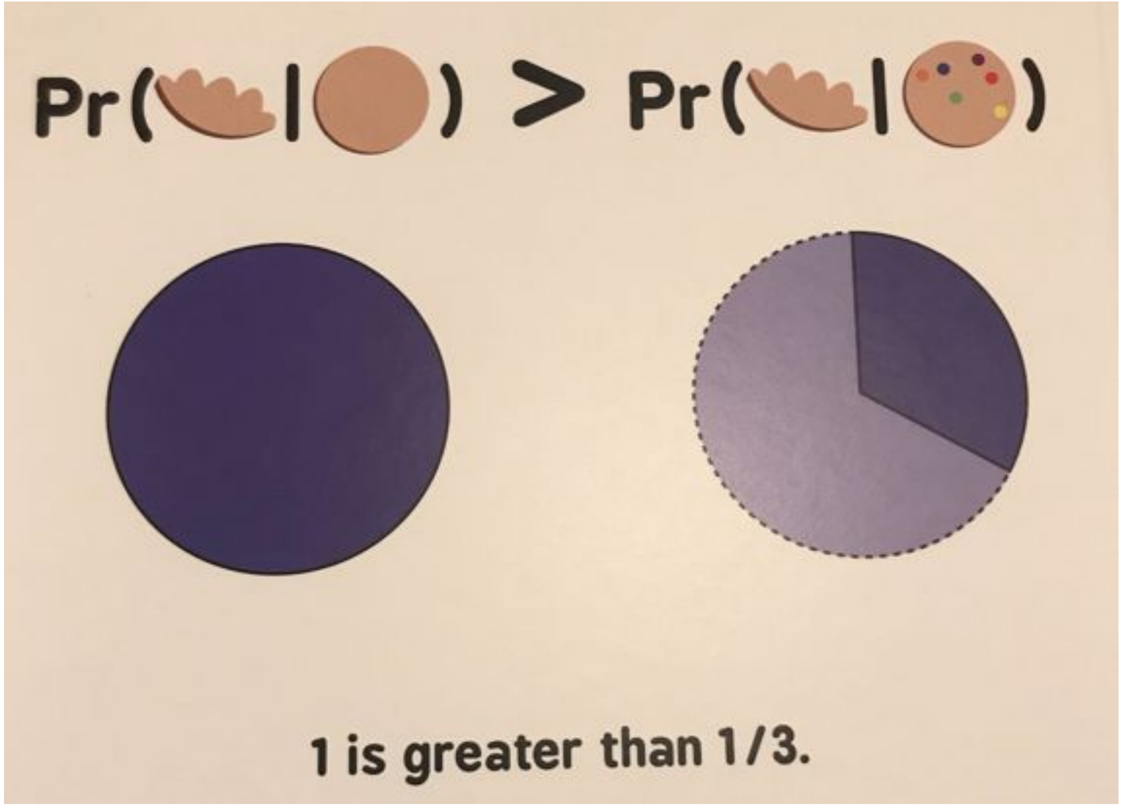
=

1/3

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

What is the maximum likelihood estimate?

1 > ⅓, so we estimate that c = 0



1 is greater than 1/3.

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

Maximum likelihood

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

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

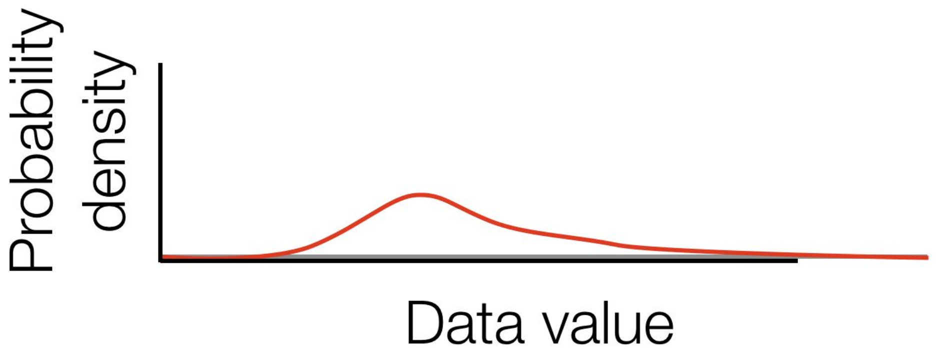
Main idea: Re-think of the probability distribution as a function of the data instead of the parameters

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|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

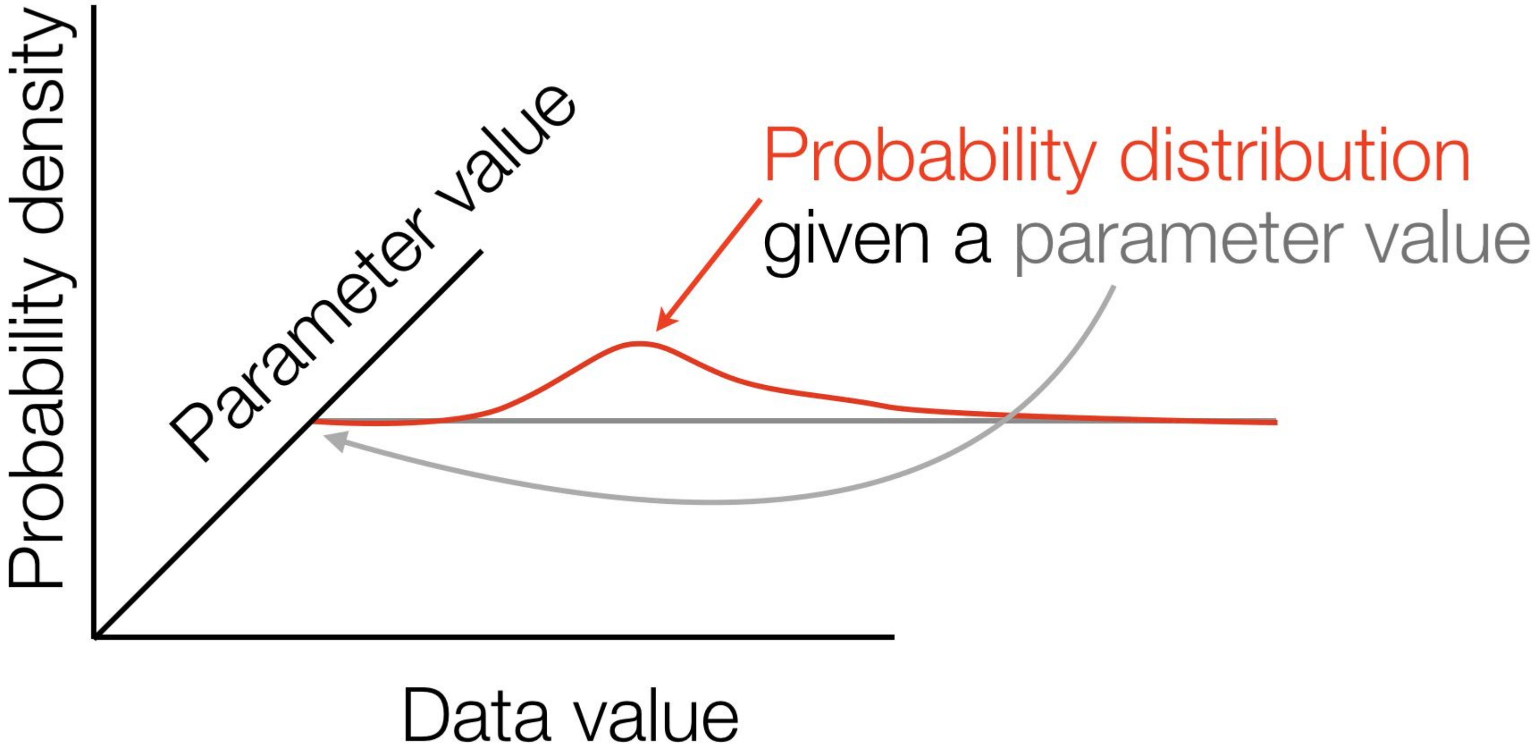
We usually plot a probability distribution something like this:



But, there are parameters that control the shape of this distribution! The mean, standard deviation, etc.

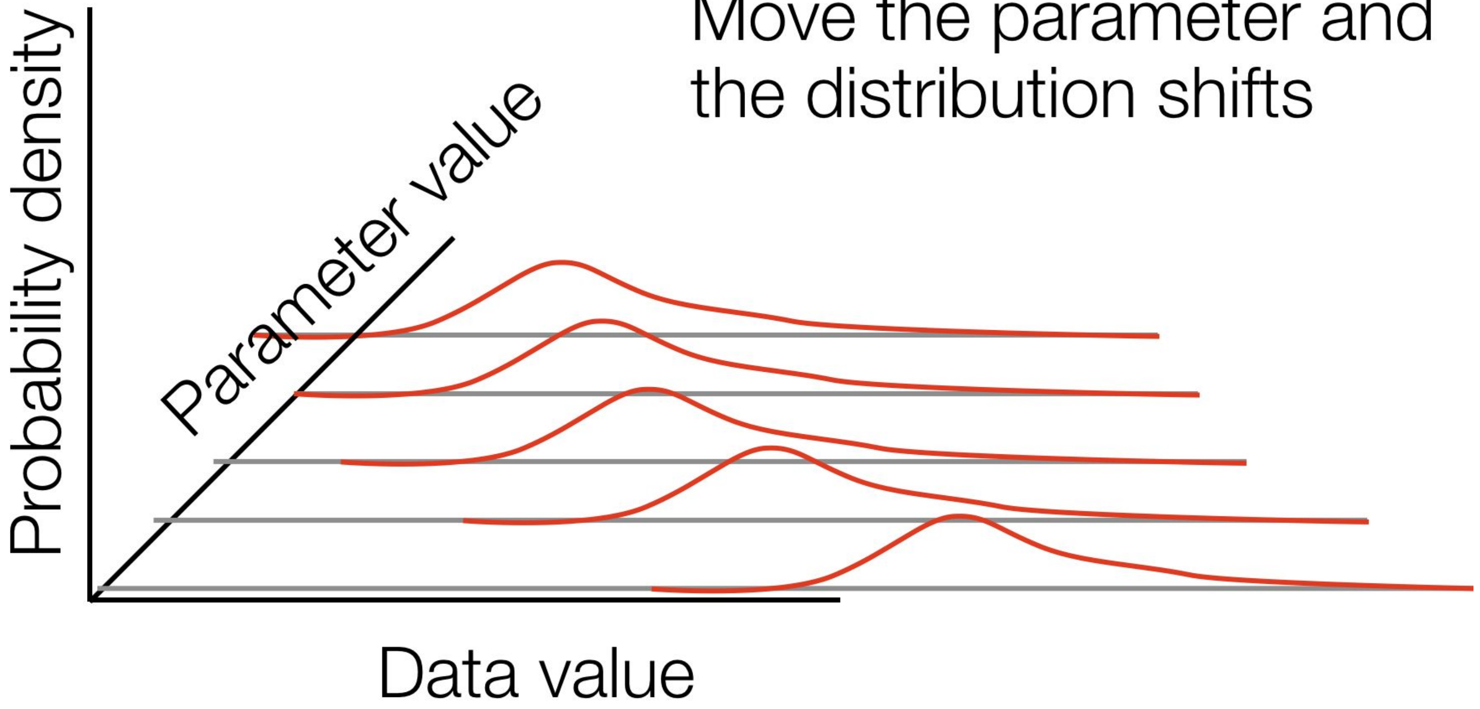
Really, we should maybe think of it more like:

Likelihood function

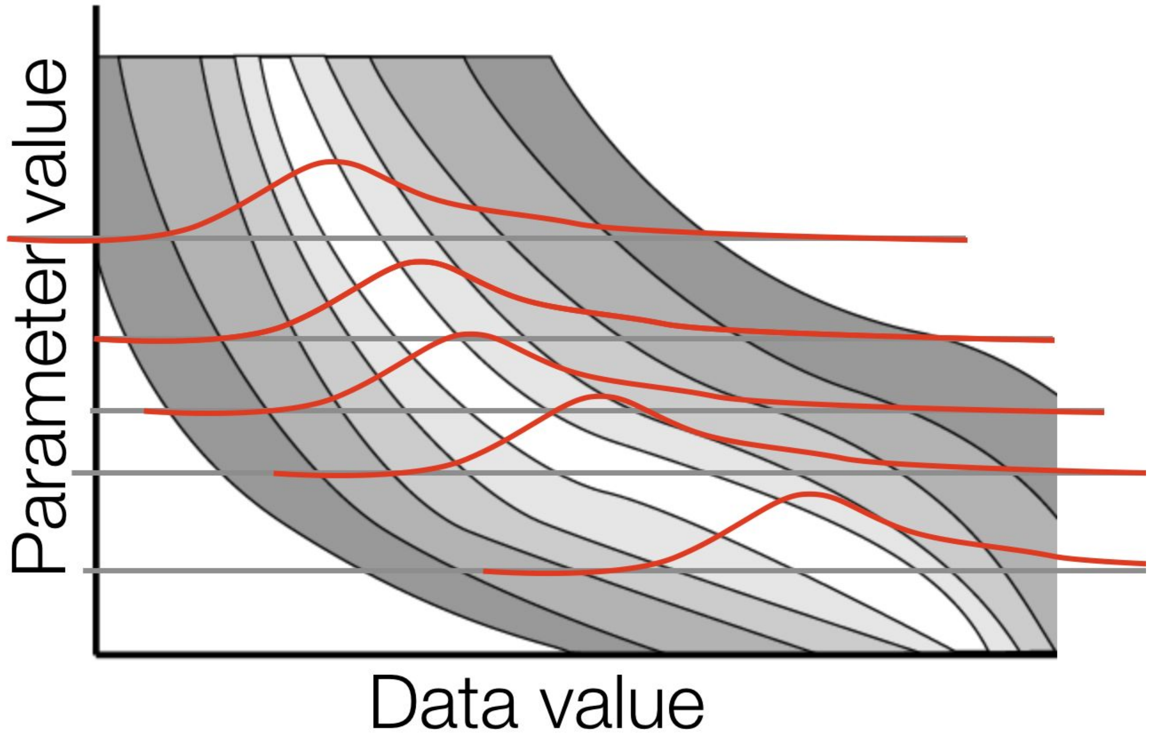


Likelihood function

Move the parameter and the distribution shifts



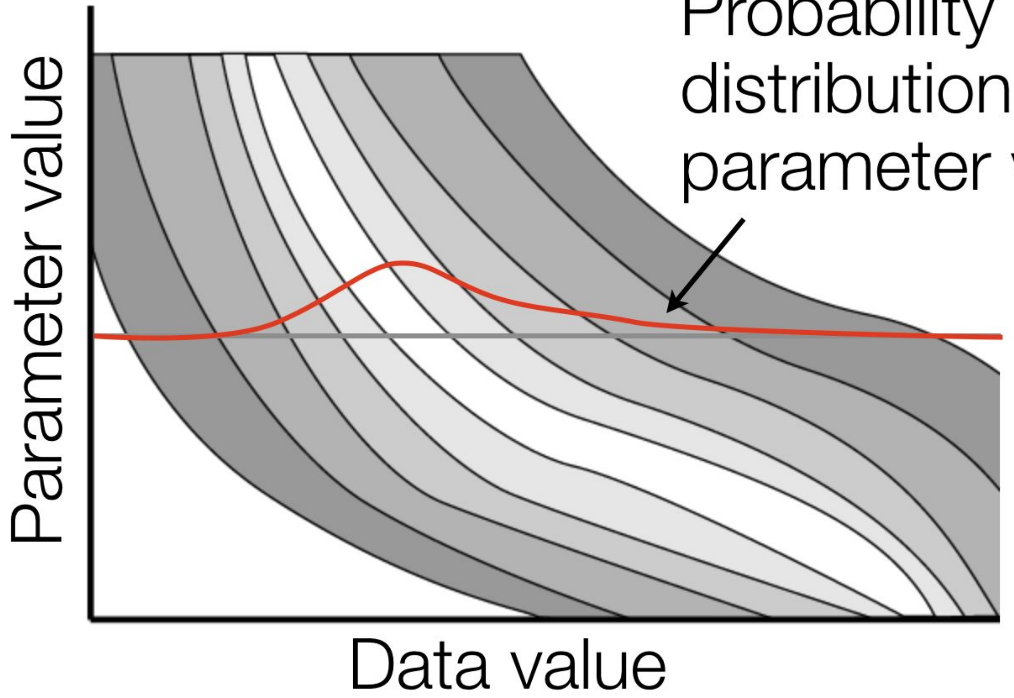
Likelihood function



Let's think of this as a heat map, where the shading shows the probability density

Think of the density as a function of both the data value and the parameter value

Likelihood function

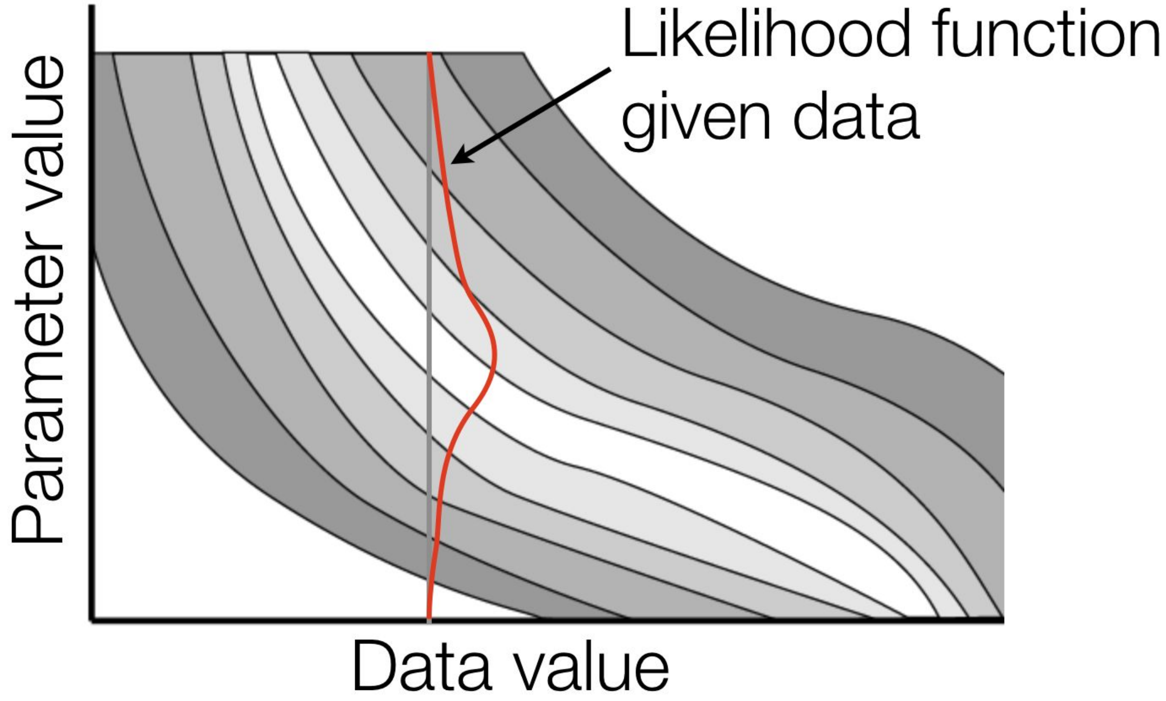


$P(z \mid p)$

Probability distribution given a parameter value

Likelihood function

$L(p \mid z)$



Maximum Likelihood

Consistency - with sufficiently large number of observations n, it is possible to find the value of p with arbitrary precision (i.e. converges in probability to p) (assuming the model is correct in the first place! A big assumption)

Normality - as the sample size increases, the distribution of the MLE tends to a Gaussian distribution with mean and covariance matrix equal to the inverse of the Fisher information matrix

Efficiency - achieves CR bound as sample size$\longrightarrow\infty$ (no consistent estimator has lower asymptotic mean squared error than MLE)

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

How does this work with more complicated models?

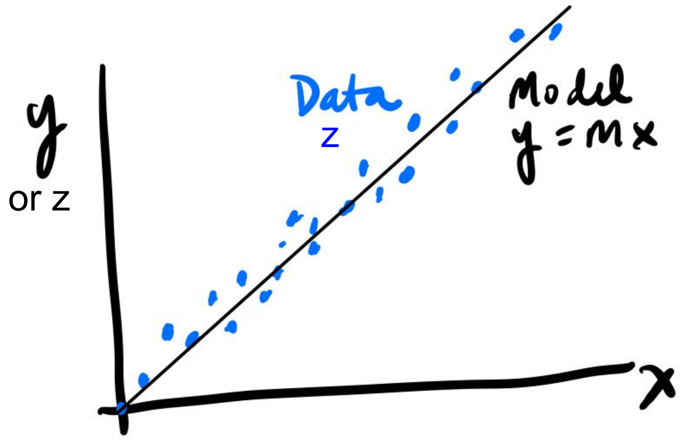
E.g. linear regression, etc?

Usually we view the model as describing one of the parameters or features/moments (e.g. the mean) of the distribution.

Revisit our least squares example—this is actually maximum likelihood!

Suppose we view the data (z) as coming from a normal distribution, but where the mean is given by a linear model like y = mx, and suppose we know the standard deviation

$$
\begin{aligned}
P(z_i|m) &= \mathcal{N}(\mu, \sigma) \\
&= \mathcal{N}(y_i, \sigma) \\
&= \mathcal{N}(mx_i, \sigma)
\end{aligned}
$$



If you work out the algebra (do if we have time, otherwise next time), it turns out this maximum likelihood cost function is actually just least squares!

Code Part 3: Adapt our previous code example and compare

Examples: what might a model and likelihood function be for the following situations?

1. **Data**: 5 coin tosses, HTTHH
   Parameter to estimate: coin bias (i.e. probability to show heads)

2. **Data**: a sample of 100 people's ages and estimated happiness level (between 0-10 say)
   Question of interest: How does age (input) affect happiness level (output)?
   Parameters to estimate: effect of age on happiness level, and maybe some sort of baseline happiness level

Many of these have multiple right answers! Also, we will talk more about different distributions and what they're used for in a moment

Probability distribution cheatsheet

Examples: what might a model and likelihood function be for the following situations?

1. **Data**: incidence of bicycle accidents each year
   Parameter to estimate: rate of bicycle accidents

2. **Data**: incidence of bicycle accidents last year by state (suppose all states have the same number of people), number of bike lanes in each state
   Question of interest: What is the impact of bike lanes on reducing bicycle accidents?
   Parameters to estimate: effect of bike lanes on bicycle accident rate, baseline bicycle accident rate without bike lanes
   What problems might this model/data/likelihood have?

Many of these have multiple right answers!

Code Part 3: More detailed example from start to finish

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 mislead by parameter estimates if this is the case (more on this next week)
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)

Next week

Introduce basic ideas for:

- Connecting least squares and maximum likelihood
- Assessing goodness of fit
- Uncertainty, confidence intervals, etc
- Exploration of models, regression
- Bayesian estimation and MCMC