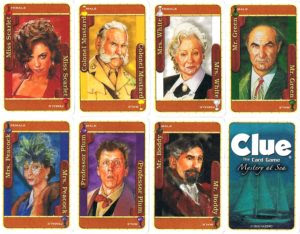
**tl;dr** -I don’t remember how many games of Clue I’ve played but I do remember being surprised by Mrs White being the murderer in only 2 of those games. Can you give an estimate and an upper bound for the number of games I have played?  
We solve this problem by using Bayes theorem and discussing the data generation mechanism, and illustrate the solution with R.

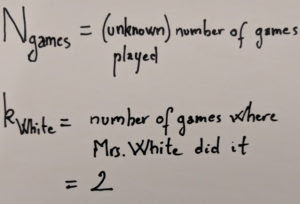
[](https://i2.wp.com/nc233.com/wp-content/uploads/2019/04/characters.jpg)The characters in the original game of Clue. Mrs White is third from the left on the first row (and is now retired!)

**Making use of external information with Bayes theorem**

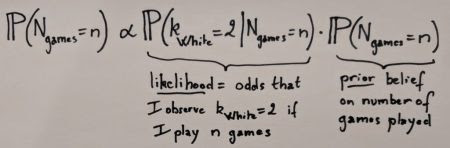
Having been raised a [frequentist](http://rebecq.fr/research.html), I first tried to solve this using a max likelihood method, but quickly gave up when I realized how intractable it actually was, especially for the upper bound.

This is a problem where conditioning on external knowledge is key, so the most natural way to tackle it is to use [Bayes theorem](https://en.wikipedia.org/wiki/Bayes'_theorem). This will directly yield an interpretable probability for what we’re looking for (most probable number of and [uncertainty interval](https://statmodeling.stat.columbia.edu/2016/11/26/reminder-instead-confidence-interval-lets-say-uncertainty-interval/))

Denote an integer n>3 and:

[](https://i1.wp.com/nc233.com/wp-content/uploads/2019/04/notations.jpg)Our notations for the problem

What we want writes as a simple product of quantities that we can compute, thanks to Bayes:

[](https://i1.wp.com/nc233.com/wp-content/uploads/2019/04/bayes.jpg)Bayes theorem gives us directly the probability we are looking for

Notice that there is an “proportional to” sign instead of an equal. This is because the denominator is just a normalization constant, which we can take care of easily after computing the likelihood and the prior.

**Likelihood**

The likelihood indicates the odds of us observing the data (in this case, that k\_Mrs\_White = 2) given the value of the unknown parameter (here the number of games played). Since at the beginning of each game the murderer is chosen at uniform random between 6 characters, the number of times Mrs White ends up being the culprit can be modeled as a [binomial distribution](https://en.wikipedia.org/wiki/Binomial_distribution) with parameters n and 1/6.

This will be easily obtained using the [dbinom](https://stat.ethz.ch/R-manual/R-patched/library/stats/html/Binomial.html) function, which gives directly the [exact value of P(x = k)](https://en.wikipedia.org/wiki/Probability_mass_function), for any x and a binomial distribution of parameters n and p. Let’s first import a few useful functions that I put in [our GitHub repo](https://github.com/haroine/nc233_blog) to save some space on this post, and set a few useful parameters:

library(tidyverse)

source("clue/clue\_functions.R")

## Parameters

k\_mrs\_white <- 2 # Number of times Mrs. White was the murderer

prob <- 1/6 # Probability of Mrs. White being the murderer for one game

Note that we can’t exactly obtain the distribution for any game from 1 to infinity, so we actually compute the distribution for 1 to 200 games (this doesn’t matter much in practice):

x <- 1:200 # Reduction of the problem to a finite number of games

## Likelihood

dlikelihood <- dbinom(k\_mrs\_white, x, prob)

easy enough 

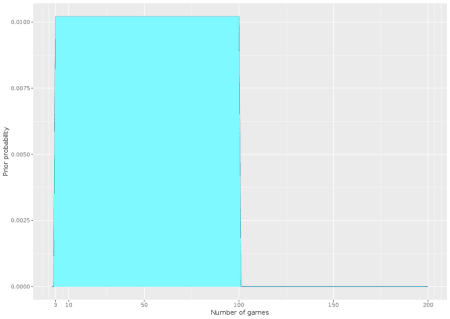
Side note: when I was a student, I kept forgetting that the distribution functions existed in R and whenever I needed them I used to re-generate them using the random generation function (rbinom in this case) 

**Prior**

There are a lot of possible choices for the prior but here I’m going to consider that I don’t have any real reason to believe and assume a uniform probability for any number of games between 3 and 200:

dprior1 <- dunifdisc(x,3,100)

plot\_clue\_prior(x, dprior1)

[](https://i2.wp.com/nc233.com/wp-content/uploads/2019/04/uniform_prior_linear.png)Uniform prior for all number of games between 3 and 100

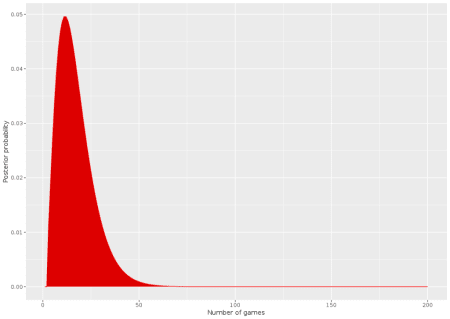
**First posterior**

Using the likelihood and the prior, we can easily compute the posterior, normalize it and plot it:

dposterior1 <- dlikelihood \* dprior1

dposterior1 <- dposterior1 / sum(dposterior1)

plot\_clue\_posterior(x, dposterior1)

[](https://i2.wp.com/nc233.com/wp-content/uploads/2019/04/posterior_1.png)Plot of the first posterior computed

We can also compute directly the estimates we’re looking for. The most probable number of games played is 11:

> which.max(dposterior1)

[1] 11

And there is a 95% chance that the number of games is less than 40:

> threshold\_val <- 0.975

> which(cumsum(dposterior1) > (threshold\_val))[1]

[1] 40

**A more realistic data generation mechanism**

I find this result very unsatisfying. It doesn’t “feel” right to me that someone would be surprised by only 2 occurrences of Mrs White being guilty in such a small number of games! For example, I simulated 40 games, a number that was supposedly suspiciously high according to the model:

set.seed(9)

N\_sim\_games <- 40

sim\_murderer <- runifdisc(N\_sim\_games, 6)

plot\_murderer <- ggplot(tibble(x=1:N\_sim\_games, y=sim\_murderer), aes(y, stat(count))) +

geom\_histogram(aes(y =..count..),

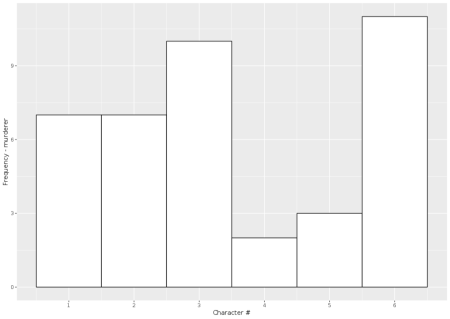
bins=6, fill="white",colour="black") +

ylab("Frequency - murderer") +

xlab("Character #") +

scale\_x\_continuous(breaks = 1:6)

print(plot\_murderer)

[](https://i0.wp.com/nc233.com/wp-content/uploads/2019/04/simulations_clue.png)Simulating 40 games of Clue. Character #4 and #5 were the murderer in respectively only 2 and 3 games

We observe that characters #4 and #5 are the murderers in respectively only 2 and 3 games!

In the end I think what really counts is not the likelihood that **\*Mrs White\*** was the murderer 2 times, but that the **\*minimum\*** number of times one of the characters was the culprit was 2 times!

I think it’s a cool example of a problem where just looking at the data available is not enough to make good inference – you also have to think about **\*how\*** the data was generated (in a way, it’s sort of a twist of the Monty Hall paradox, which is one of the most famous examples of problems where the data generation mechanism is critical to understand the result).

I wrote a quick and dirty function based on simulations to generate this likelihood, given a certain number of games. I saved the distribution directly in the GitHub (and called it [Gumbel](https://en.wikipedia.org/wiki/Gumbel_distribution) since it kinda looks like an extreme value problem) so that we can call it and do the same thing as above:

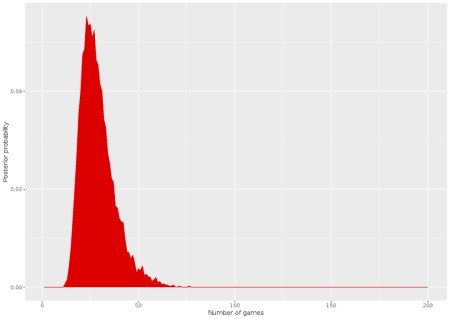
gumbelclue\_2 <- readRDS("clue/dcluegumbel\_2.rds")

gumbelclue\_2 <- gumbelclue\_2[x]

dposterior\_gen <- gumbelclue\_2 \* dprior1

dposterior\_gen <- dposterior\_gen / sum(dposterior\_gen)

plot\_clue\_posterior(x, dposterior\_gen)

[](https://i1.wp.com/nc233.com/wp-content/uploads/2019/04/posterior_2.png)Posterior with the new data generation mechanism

The new posterior has the same shape but appears shifted to the right. For example N\_games = 50 seems much more likely now! The estimates are now **23** for the number of games:

> which.max(dposterior\_gen)

[1] 23

And **51** for the max bound of the uncertainty interval

> threshold\_val <- 0.975

> which(cumsum(dposterior\_gen) > (threshold\_val))[1]

[1] 51