Technische Universiteit Delft
Faculteit Elektrotechniek, Wiskunde en Informatica
Delft Institute of Applied Mathematics

A Coin tossing strategy for a High-Low guessing game

Verslag ten behoeve van het
Delft Institute for Applied Mathematics
als onderdeel ter verkrijging

van de graad van

BACHELOR OF SCIENCE
in
TECHNISCHE WISKUNDE

door

Jacob de Zoete

Delft, Nederland
Juni 2010

BSc verslag TECHNISCHE WISKUNDE

"Coin tossing strategies"

Jacob de Zoete

Technische Universiteit Delft

Begeleider

Dr.ir. R.J. Fokkink

Overige commissieleden

Dr. J.A.M. de Groot

Dr. J.A.M. van der Weide

Dr. J.G. Spandaw/Drs. A. Verweij

...

Juni, 2010

Delft

Contents

Abstract

Consider the following guessing game: Lucy thinks of a number that is in between 0 and 100 and James tries to guess it as fast as possible. Lucy says 'Higher' when the guess is too low and she says 'Lower' when the guess is too high. A good strategy for James is tossing 100 fair coins where the first guess is equal to the number of coins that show heads. If this guess is too low he retosses all the tail coins and if the guess is too high he retosses all the head coins. What is the expected number of tosses in this strategy? This leads to all kinds of coin tossing problems. Most notably a problem by Råde. I discuss this problem and I consider how James may improve his strategy by adapting the coins. This leads to recursive problems.

1 Introduction

This report is, just like a lot of other reports that concern probability, about tossing coins.

The guessing game in which one player takes a secret number in mind and another player tries to guess it, is a well known game. After each guess the first player, from now on referred as the 'Hider' tells whether the other player's guess is lower or higher than the secret number he has in mind. Knowing this, the other player, from now on referred as the 'Guesser' makes a new guess. This happens a few times until the guess is the same as the secret number. Using your intuition the best way to play this game for the Guesser is to take the number that's right in between the two boundaries the Hider used to pick a number. For instance, when the Hider picked a number between 0 and 100 most 'Guessers' would pick 50 as a first guess. And as a second guess 25 or 75, which depends on the outcome of the first guess (higher of lower). This way of searching is known as a Binary Search in computer science. Searching this way has a disadvantage. Knowing that the Guesser will use this way of searching the Hider would always pick numbers like 0, 49, 51 or 100 as a secret number cause these will take the most guesses. A guessing strategy should randomize, for instance by tossing 100 coins and take the number of coins that show heads as an initial guess. This way it's most likely to take a number that's in the middle as a first guess, but the probability of tossing 49, 50 or 51 are almost equal. When the secret number is higher, the Guesser tosses all the tail-coins and takes as a new guess the new number of coins that show heads. When the secret number is lower, the Guesser tosses all the head-coins. An example of the guessing strategy is presented in section 1.1.

Suppose the secret number is 0. This means that you always need to retoss the head-coins until al coins are tails. A question that could arise is the following:

If you use the coin tossing strategy, what is the probability that you guess 1 *before guessing* 0*?*

This question is equivalent to a problem that was proposed by Lennart Råde in 1991[2]:

Problem 1.1. *Suppose we have n identical coins for each of which heads occurs with probability p. Suppose we first toss all the coins, then toss those which show tails after the first toss, then toss those which show tails after the second toss, and so on until all the coins show heads. Let X be the number of coins involved in the last toss.*

(a) *Find $P(X = i)$ for $i = 1, 2, \cdots, n$ and $E[X]$.*
(b) *Let $p_n = P(X = 1)$. Analyze the behavior of p_n as $n \to \infty$.*

In other words, we look at the probability that the last toss has i coins and analyze the behavior of the probability to toss with 1 coin in the final toss as the number of coins go to infinity.

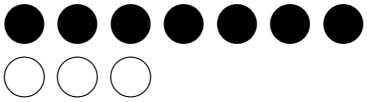
In this report we first reconsider results from a paper by Steutel et al.[1] Then we consider the expected number of guesses in the coin tossing strategy. Finally we consider what happens if we take biased coins.

1.1 Example of the guessing strategy

To illustrate this guessing strategy we present an example:

Suppose that John, the Hider, picked a secret number between 0 and 10, say 4. Paul, the Guesser, tries to guess it by tossing 10 coins and take the number of coins that land on heads as an initial guess. We represent coins that land on heads with black circles and coins that land on tails with white circles.

1. The first toss results in;

   Paul tosses 7 head coins so his first guess is 7. John tells him that the secret number is lower so Paul puts the 3 coins that show tails aside;

2. Paul now retosses all the head coins, this results in:

3. The second guess of Paul is 3 and now John tells him that his secret number is higher. Paul puts the 3 head coins aside and tosses the 4 remaining coins again;

4. Paul's third guess is 5 so John tells him that his secret number is lower. Now there is a inefficiency in our strategy: It is clear that the secret number is 4 (it is less then 3 and it is above 1), but our strategy tells us to toss another time. Paul puts the 'tails' coin aside and tosses the remaining 2 coins once more:

   The final guess is 4: Paul tosses 1 head coin and adds the 3 head coin that he tossed earlier. The total number of guesses is 4.

2 A first look at Råde's Problem

In this chapter we'll be working on a solution for Råde's Problem, especially part (b). We follow the approach of Brands, Steutel and Wilms in their paper "On the number of maxima in a discrete sample"[1]. I would like to thank Prof. Steutel for the personal communication on a topic of the paper. Almost all results in this section are taken from that paper but there are a few results of my own. First we'll do Råde's problem for general probability distributions and later on for a geometric distribution that agrees with the tossing of coins.

2.1 Råde for general distributions

Råde's question about the number of coins in the final toss is not so easy to manage. That is why we reformulate this question the following way. Suppose that we're tossing one coin at a time, and we keep tossing it until it becomes heads. We say that N_i is the number of tosses needed for the individual coin i to become heads. We'll do this n times (i.e. with n coins). The maximum number of tosses a single coin required will be denoted by:

$$M_n = \max(N_1, \ldots, N_n)$$

The number of coins that required the maximum number of tosses will be denoted by:

$$K_n = \#\{j \in \{1, 2, \ldots, n\} : N_j = M_n\},$$

Now we see that the number of coins in the final toss in Råde's question is actually the same as the value K_n. This variable K_n is easier to manage.

From now one we'll be making use of the following notation:

$$p_j = P(N_1 = j), \quad P_0 = 0, \quad P_j = \sum_{k=1}^{j} p_k, \quad \bar{P}_j = 1 - P_{j-1} \quad (j = 1, 2, \ldots). \tag{1}$$

So p_j is the probability that a coin needs j tosses until it shows heads. P_j is the probability that a coin needs at most j tosses. And $\bar{P}_j$ is the probability that a coin needs a minimum of j tosses to become heads. The probability that K_n equals a certain value is given by the sum of every possible outcome:

Lemma 2.1.

$$P(K_n = k) = \binom{n}{k} \sum_{j=1}^{\infty} p_j^k P_{j-1}^{n-k} \qquad (k = 1, 2, \ldots, n; n = 1, 2, \ldots). \tag{2}$$

Proof. We'll want to compute the probability of k successes and $n - k$ failures, where a success is defined by tossing the maximum number of coins. On a total of n coins this can be done in $\binom{n}{k}$ ways. Because the different coin tosses are independent of each other and using symmetry between the coins:

$$\begin{aligned} P(K_n = k) &= \binom{n}{k} \sum_{j=1}^{\infty} P(N_1 = \ldots = N_k = j, N_{k+1} \leq j - 1, \cdots, N_n \leq j - 1) \\ &= \binom{n}{k} \sum_{j=1}^{\infty} p_j^k P_{j-1}^{n-k} \end{aligned}$$

□

To obtain $E[K_n]$ we'll use that the expectation of a discrete variable is the sum of all possible outcomes times the probability for that particular outcome.

$$\begin{aligned} E[K_n] = \sum_{k=1}^{\infty} kP(K_n = k) &= \sum_{k=1}^{\infty} k \sum_{j=1}^{\infty} P(K_n = k, M_n = j) \\ &= \sum_{k=1}^{\infty} k \binom{n}{k} \sum_{j=1}^{\infty} p_j^k P_{j-1}^{n-k} \end{aligned}$$

And because $k\binom{n}{k} = k\frac{n(n-1)\cdots(n-k+1)}{k(k-1)\cdots 1} = n\frac{(n-1)(n-2)\cdots(n-k+1)}{(k-1)(k-2)\cdots 1} = n\binom{n-1}{k-1}$

$$\sum_{k=1}^{\infty} k \binom{n}{k} \sum_{j=1}^{\infty} p_j^k P_{j-1}^{n-k} = \sum_{k=1}^{\infty} n \binom{n-1}{k-1} \sum_{j=1}^{\infty} p_j^k P_{j-1}^{n-k}$$

Because we're only summing positive terms we can interchange the order in which we sum:

$$\sum_{k=1}^{\infty} n \binom{n-1}{k-1} \sum_{j=1}^{\infty} p_j^k P_{j-1}^{n-k} = n \sum_{j=1}^{\infty} p_j \sum_{k=1}^{\infty} \binom{n-1}{k-1} p_j^{k-1} P_{j-1}^{n-k}$$

Now notice that $\binom{n}{m} = 0$ for all $m > n$.

$$n \sum_{j=1}^{\infty} p_j \sum_{k=1}^{\infty} \binom{n-1}{k-1} = n \sum_{j=1}^{\infty} p_j \sum_{k=1}^{n} \binom{n-1}{k-1} p_j^{k-1} P_{j-1}^{n-k}$$

By interchanging $k-1$ by k and $n-1$ by n in the last sum we'll get:

$$n \sum_{j=1}^{\infty} p_j \sum_{k=1}^{n} \binom{n-1}{k-1} p_j^{k-1} P_{j-1}^{n-k} = n \sum_{j=1}^{\infty} p_j \sum_{k=0}^{n} \binom{n}{k} p_j^{k} P_{j-1}^{n-k}$$

And now we're able to use the Binomium of Newton

$$n \sum_{j=1}^{\infty} p_j \sum_{k=0}^{n} \binom{n}{k} p_j^{k} P_{j-1}^{n-k} = n \sum_{j=1}^{\infty} p_j (p_j + P_{j-1})^{n-1}$$

And this gives the following Lemma:

Lemma 2.2.

$$E[K_n] = n \sum_{j=1}^{\infty} p_j P_j^{n-1} \tag{3}$$

If we fill in $k = 1$ in formula (2) we have an expression for the probability that there is only a single coin that became heads after the maximum number of tosses. This is the probability that the number of maximum values is equal to one, i.e. the probability that the last toss involves a single coin which is one part of Råde's problem:

$$P(K_n = 1) = n \sum_{j=1}^{\infty} p_j P_{j-1}^{n-1} \tag{4}$$

The reader should compare the formulas for $P(K_n = 1)$ and $E[K_n]$. The expressions are very similar. In Corollary 2.3 we show that the expressions can be asymptotically the same.

To avoid trivial cases we'll put a constraint on the distribution of p_j. If $p_j = 0$ for $j > M$, in other words; it is impossible to toss more then $j-1$ times, then the number of coins that will be using the maximum number of tosses, M, will grow larger and larger and so $K_n \to \infty$ as $n \to \infty$. To avoid this we'll be assuming that the N_i's are unbounded; $P_j < 1$ for all j.

At this point we'll make our problem a stochastic process. Instead of $P(K_n = k)$ for fixed n we'll be working with $P(K_n = k)$ for increasing n. So after we've determined the value of K_i for i coins we'll toss another coin (i.e. coin number $i+1$) which means we can determine the new value K_{i+1}.

Now we would like to say something about the event that there is a multiple number of coins that need the maximum number of tosses to become heads. Which condition do we need on the probability space to be certain that there will be one coin in the final toss if we increase n, the total number of coins? If we can be sure that when n increases the probability $P(K_n = 2)$ becomes 0 then we're also sure that the same happens for $P(K_n = 3), P(K_n = 4), ...$ To prove a theorem concerning this question we'll need a part of the Borel-Cantelli Lemma[4].

Lemma 2.3. [Borel – Cantelli]
Let $E_1, E_2, ...$ denote a sequence of events. If

$$\sum_{i=1}^{\infty} P(E_i) < \infty,$$

then,

$$P\{\textit{an infinite number of the } E_i \textit{ occur}\} = 0$$

Using this Lemma we're able to prove the following theorem:

Theorem 2.1. *If,*

$$\sum_{j=2}^{\infty} p_j^2 \bar{P}_j^{-3} < \infty$$

then,

$$P\{\textit{an infinite number of } K_n = 2 \textit{ occur}\} = 0$$

Proof. We use Borel-Cantelli with $E_i = \{K_n = 2\}$. To prove our theorem we need that

$$\sum_{n=1}^{\infty} P(K_n = 2) < \infty$$

So if we fill in $k = 2$ in our equation for $P(K_n = k)$ (2):

$$\sum_{n=1}^{\infty} \binom{n}{2} \sum_{j=1}^{\infty} p_j^2 P_{j-1}^{n-2} < \infty$$

Now we ignore the right part of the equation and work with the left part:

$$\sum_{n=1}^{\infty} \frac{1}{2}(n^2 - n) \sum_{j=1}^{\infty} p_j^2 P_{j-1}^{n-2}$$

Because we're only summing positive terms we can interchange the order of summation. And because $P_0 = 0$ by definition (see (1)) we can start summing from $j = 2$:

$$\sum_{j=2}^{\infty} \frac{p_j^2}{2} \sum_{n=1}^{\infty} (n^2 - n) P_{j-1}^{n-2} \tag{5}$$

Now we use that, for $r < 1$:

$$\sum_{n=1}^{\infty} r^n = \frac{1}{1-r}$$

If we differentiate twice on both sides:

$$\begin{aligned} \sum_{n=1}^{\infty} n r^{n-1} &= -\frac{1}{(1-r)^2} \\ \sum_{n=1}^{\infty} (n^2 - n) r^{n-2} &= \frac{2}{(1-r)^3} \end{aligned}$$

So, if we interchange r by P_{j-1}, which is less than 1 for all j, in (5):

$$\sum_{j=2}^{\infty} \frac{p_j^2}{2} \sum_{n=1}^{\infty} (n^2 - n) P_{j-1}^{n-2} = \sum_{j=2}^{\infty} p_j^2 \bar{P}_j^{-3}$$

Which converges due to our assumption. □

An example of a distribution that satisfies the condition of this theorem is the following:

Corollary 2.1. *If $p_j = cj^{-\alpha}$, where c is some constant and $1 < \alpha < 2$, then:*

$$P\{\textit{an infinite number of } K_n = 2 \textit{ occur}\} = 0$$

Proof. By using Theorem 2.1 we know that it's enough to show that

$$\sum_{j=2}^{\infty} p_j^2 \bar{P}_j^{-3} < \infty$$

For $\bar{P}_j$ we know that:

$$\bar{P}_j = 1 - P_{j-1} = 1 - \sum_{k=1}^{j-1} p_k = \sum_{k=j}^{\infty} p_k$$

We know that p_j is decreasing in j because $\alpha > 1$. We can use this in the next inequality:

$$p_k \leq \int_{k-1}^{k} p(y) dy$$

So:

$$\bar{P}_j = \sum_{k=j}^{\infty} p_k \leq \sum_{k=j}^{\infty} \int_{k-1}^{k} p(y) dy = \int_{j-1}^{\infty} p(y) dy$$

Now we're able to fill in the assumption of Theorem 2.1:

$$\sum_{j=2}^{\infty} p_j^2 \bar{P}_j^{-3} \leq \sum_{j=2}^{\infty} p_j^2 \left(\int_{j-1}^{\infty} p(y) dy \right)^{-3}$$

Now we fill in our definition for p_j, because we're proving that the above sum is less then infinite we can ignore our constant c:

$$\sum_{j=2}^{\infty} j^{-2\alpha} \left(\int_{j-1}^{\infty} y^{-\alpha} dy \right)^{-3} = \sum_{j=2}^{\infty} j^{-2\alpha} \left(\frac{(j-1)^{1-\alpha}}{\alpha - 1} \right)^{-3}$$

Using the same argument as above we can ignore the constant term $\alpha - 1$:

$$\sum_{j=2}^{\infty} j^{-2\alpha}(j-1)^{3\alpha-3}$$

To show that this sequence converges we compare it with the sequence $\sum j^{\alpha-3}$ which is convergent for $\alpha < 2$. For these two sequences we use the ratio test:

$$\lim_{j\to\infty} \frac{j^{-2\alpha}(j-1)^{3\alpha-3}}{j^{\alpha-3}} = \lim_{j\to\infty} \frac{(j-1)^{3\alpha-3}}{j^{3\alpha-3}} = 1$$

And with this result we've proven that the sequence converges for $1 < \alpha < 2$. □

Brands et al. obtained a stronger theorem. To get this we'll be making use of Lebesgue's dominated convergence theorem [5]:

Theorem 2.2. [Lebesgue′s dominated convergence theorem]
Let $f_1, f_2, \ldots, f_n, \ldots$ denote a sequence of integrable functions that converges almost everywhere to a function f. If g is a nonnegative integrable function with the property that for any n:

$$|f_n| \leq g \qquad \textit{almost everywhere}$$

Then f is integrable and

$$\lim_{n\to\infty} \int f_n = \int f$$

We will use this theorem on the counting measure $\mathbb{N}$, which means that $\int f_n = \sum f_n$. We define 'almost everywhere' as:

$$P(|f_n| \leq g) = 1$$

Brands et al. come up with this 'stronger' theorem about our probability concerning the event $K_n = 2$ by using a variation on the Borel-Cantelli lemma, which was made by Barndorff-Nielsen in 1961 [3]:

Lemma 2.4. [Barndorff − Nielsen]
Let $A_n (n \geq 1)$ be a sequence of events satisfying $P\{A_n\} \to 0$. Let also

$$\sum_{n=1}^{\infty} P\{\bar{A}_n \cap A_{n+1}\} < \infty \tag{6}$$

then,

$$P\{\textit{an infinite number of the } E_i \textit{ occur}\} = 0$$

Now we're able to prove the following theorem:

Theorem 2.3. *If,*

$$\sum_{j=1}^{\infty} p_j^2 \bar{P}_j^{-2} < \infty$$

then,

$$P\{\textit{an infinite number of } K_n = 2 \textit{ occur}\} = 0$$

Proof. We make use of the variation on the Borel-Cantelli lemma by Barndorff-Nielsen (Lemma 2.4). We'll take $A_n = \{K_n = 2\}$. The first thing we need to prove is $P\{A_n\} \to 0$:

$$P(K_n = 2) = \binom{n}{2} \sum_{j=1}^{\infty} p_j^2 P_{j-1}^{n-2}$$

Because $\bar{P}_j^{-2}(1 - P_{j-1})^2$ is equal to 1, we can multiply each term with it:

$$P(K_n = 2) = \binom{n}{2} \sum_{j=1}^{\infty} p_j^2 P_{j-1}^{n-2} = \sum_{j=1}^{\infty} p_j^2 \bar{P}_j^{-2} \binom{n}{2} P_{j-1}^{n-2}(1 - P_{j-1})^2$$

We easily have,

$$\binom{n}{2} P_{j-1}^{n-2}(1 - P_{j-1})^2 < \sum_{k=0}^{\infty} \binom{n}{k} P_{j-1}^{n-k}(1 - P_{j-1})^k$$

Because the left hand side is only one term ($k = 2$) of the right hand sum, and all the terms on the right hand side are nonnegative. Because $\binom{n}{k} P_{j-1}^{n-k}(1 - P_{j-1})^k$ is the probability mass function of a binomial distribution we have:

$$\sum_{k=0}^{\infty} \binom{n}{k} P_{j-1}^{n-k}(1 - P_{j-1})^k = 1$$

Which gives us:

$$\binom{n}{2} P_{j-1}^{n-2}(1 - P_{j-1})^2 < \sum_{k=0}^{\infty} \binom{n}{k} P_{j-1}^{n-k}(1 - P_{j-1})^k = 1 \tag{7}$$

Now we take:

$$f_n(j) = p_j^2 \bar{P}_j^{-2} \binom{n}{2} P_{j-1}^{n-2}(1 - P_{j-1})^2$$

And,

$$g(j) = p_j^2 \bar{P}_j^{-2}$$

To use Lebesgue's dominated convergence theorem we need that $f_1, f_2, \ldots, f_n, \ldots$ converges almost everywhere to a function f and that g is a nonnegative integrable function with the property that for any n:

$$|f_n| \leq g \qquad \text{almost everywhere}$$

We use the measure on $\mathbb{R}$ that has mass 1 in $\mathbb{Z}$, i.e. the counting measure on the integers. In that case:

$$\int g = \sum g$$

This means that we van write:

$$\int g = \sum g = \sum_{j=1}^{\infty} p_j^2 \bar{P}_j^{-2} < \infty$$

This proves that g is integrable.

Now we'll need that $f_1, f_2, \ldots, f_n, \ldots$ denote a sequence that converges almost everywhere to a function f. It is clear that $f_1, f_2, \ldots, f_n, \ldots$ is a sequence but does it converge almost everywhere to a function f? For fixed j we'll have that

$$\lim_{n\to\infty} f_n(j) = \lim_{n\to\infty} p_j^2 \bar{P}_j^{-2} \binom{n}{2} P_{j-1}^{n-2}(1 - P_{j-1})^2 \to 0$$

Because $\binom{n}{2}$ is equal to a polynomial function $\frac{1}{2}(n^2-n)$ and $P_{j-1}<1$. This is a standard limit that says that the exponential function goes faster to zero then a polynomial to infinity. This proves a second condition. Now we only need to prove the last condition.

By using equation (7) we directly see that,

$$f_n(j) = p_j^2 \bar{P}_j^{-2} \binom{n}{2} P_{j-1}^{n-2}(1-P_{j-1})^2 < p_j^2 \bar{P}_j^{-2} = g(j)$$

And because all terms of f_n are non negative we have $|f_n| = f_n$. Which gives us:

$$|f_n| \leq g$$

Lebesgue's dominated convergence theorem tells us that we are now allowed to interchange limit and sum:

$$\lim_{n\to\infty} \int f = \int \lim_{n\to\infty} f_n = \int f = 0$$

Which implies that

$$P(K_n = 2) \to 0 \qquad \text{if } n \to \infty$$

Now we need to prove the second statement of Barndorff-Nielsen:

$$\sum_{n=1}^{\infty} P\{\bar{A_n} \cap A_{n+1}\} < \infty :$$

Where the first part is equal to:

$$\sum_{n=2}^{\infty} P(K_n = 2, K_{n-1} \neq 2) = \sum_{n=2}^{\infty} P(K_n = 2, K_{n-1} = 1)$$

In words, the n-th coin needs the same amount of tosses as the current record $M_{n-1} = M_n$. The first $n-1$ coins have a single maximum that can arise from one of the $n-1$ coins:

$$\sum_{n=2}^{\infty} P(K_n = 2, K_{n-1} = 1) = \sum_{n=2}^{\infty} (n-1) P(N_{n-1} = N_n = M_n, N_1 < M_n, \ldots, N_{n-2} < M_n)$$

Using that coin n has to be one of the two maxima (cause after coin $n-1$ we've only one maximum and after coin n there are two) and by formula (2):

$$\sum_{n=2}^{\infty} (n-1) P(N_{n-1} = N_n = M_n, N_1 < M_n, \ldots, N_{n-2} < M_n) = \sum_{n=2}^{\infty} (n-1) \sum_{j=2}^{\infty} p_j^2 P_{j-1}^{n-2}$$

Now we interchange the order of summation. We are allowed to do this cause we're only adding positive terms:

$$\sum_{n=2}^{\infty} (n-1) \sum_{j=2}^{\infty} p_j^2 P_{j-1}^{n-2} = \sum_{j=2}^{\infty} p_j^2 \sum_{n=2}^{\infty} (n-1) P_{j-1}^{n-2}$$

Now by using that $\sum_{n=1}^{\infty} r^n = \frac{1}{1-r}$, and thus: $\sum_{n=2}^{\infty} (n-1) r^{n-2} = \frac{1}{(1-r)^2}$:

$$\sum_{j=2}^{\infty} p_j^2 \sum_{n=2}^{\infty} (n-1) P_{j-1}^{n-2} = \sum_{j=2}^{\infty} p_j^2 \bar{P}_j^{-2}$$

And with this result we've proven that

$$P\{\text{an infinite number of } K_n = 2 \text{ occur}\} = 0$$

. □

We can use this 'stronger' theorem to obtain a 'stronger' corollary. Corollary 2.1 said that a probability distribution with $p_j = cj^{-\alpha}$, where $1 < \alpha < 2$, gives new records in the number of tosses when we increase the total number of coins. Using this new theorem we're also able to improve the corollary.

Corollary 2.2. *If $p_j = cj^{-\alpha}$, where c is a constant and $\alpha > 1$, then:*

$$P\{\textit{an infinite number of } K_n = 2 \textit{ occur}\} = 0$$

Proof. This proof will be similar to the proof of corollary 2.1. By using theorem 2.3 we know that we have to show that:

$$\sum_{j=2}^{\infty} p_j^2 \bar{P}_j^{-2} < \infty$$

By following the exact same steps as in the proof of corollary 2.1 we come to the following equation (we take c=1):

$$\sum_{j=2}^{\infty} p_j^2 \bar{P}_j^{-2} = \sum_{j=2}^{\infty} j^{-2\alpha}(j-1)^{2\alpha-2} \leq \sum_{j=2}^{\infty} j^{-2\alpha} j^{2\alpha-2} = \sum_{j=2}^{\infty} j^{-2}$$

And this sequence converges. □

We see that by using Barndorff-Nielsen we get a firmly stronger theorem. This corollary works for all values of α that are greater than 1, where corollary 2.1 only works for values of α that are in between 1 and 2.

The following theorem gives us a condition that makes certain that there will be new maximum numbers of tosses as $n \to \infty$:

Theorem 2.4. *If for the values of $p_1, p_2, \ldots$ it holds that*

$$\liminf_{j\to\infty}(p_j/p_{j-1}) = 1, \tag{8}$$

then,

$$\lim_{n\to\infty} P(K_n = 1) = 1.$$

Proof. We have

$$\begin{aligned} P_j^n - P_{j-1}^n &= \left(\sum_{k=1}^{j} p_k\right)^n - \left(\sum_{k=1}^{j-1} p_k\right)^n \\ &= (p_1 + \ldots + p_j)^n - (p_1 + \ldots + p_{j-1})^n \\ &= (P_{j-1} + p_j)^n - (P_{j-1})^n \end{aligned}$$

By using the Binomium of Newton we can simplify our expression:

$$(P_{j-1} + p_j)^n - (P_{j-1})^n = \sum_{k=0}^{n} \binom{n}{k} P_{j-1}^{n-k} p_j^k - P_{j-1}^n$$

We see that if we take $k = 0$ we find $P_{j-1}^n - P_{j-1}^n$. So, may we sum from $k = 1$ to n instead of summing from $k = 0$ to n:

$$\sum_{k=1}^{n} \binom{n}{k} P_{j-1}^{n-k} p_j^k$$

If we write out the expression:

$$p_j P_{j-1}^{n-1} + p_j^2 P_{j-1}^{n-2} + ... + p_j^n = p_j(P_{j-1}^{n-1} + p_j P_{j-1}^{n-2} + ... + p_j^{n-1})$$

Which implies,

$$P_j^n - P_{j-1}^n = p_j(P_{j-1}^{n-1} + p_j P_{j-1}^{n-2} + ... + p_j^{n-1})$$

and,

$$np_j P_{j-1}^{n-1} \leq P_j^n - P_{j-1}^n \leq np_j P_j^{n-1} \tag{9}$$

It is clear that $P(K_n = 1) \leq 1$. We now want to prove that $\lim_{n\to\infty} P(K_n = 1) \geq 1 - \epsilon$ for any $\epsilon > 0$. If we choose m in such a way that $p_j/p_{j-1} \geq 1 - \epsilon$ if $j \geq m$, which we can do because of the condition that $\liminf_{j\to\infty}(p_j/p_{j-1}) = 1$, then:

$$1 \geq P(K_n = 1) = n\sum_{1}^{\infty} p_j P_{j-1}^{n-1} \geq n \sum_{j=m+1}^{\infty} p_j P_{j-1}^{n-1} = n \sum_{j=m+1}^{\infty} \frac{p_j}{p_{j-1}} p_{j-1} P_{j-1}^{n-1}$$

$$\geq n(1-\epsilon) \sum_{j=m}^{\infty} p_j P_j^{n-1} \geq (1-\epsilon) \sum_{j=m}^{\infty} (P_j^n - P_{j-1}^n)$$

$$= (1-\epsilon)(1 - P_{m-1}^n),$$

And for $(1 - P_{m-1}^n)$ holds that if $n \to \infty$ then $(1 - P_{m-1}^n) \to 1$ for each fixed m. And with this we've proven our theorem. □

Corollary 2.3. *If we have that p_j is such that $\liminf_{j\to\infty}(p_j/p_{j-1}) = 1$ then,*

$$\lim_{n\to\infty} EK_n = 1.$$

Proof. We first note that:

$$E[K_n] = \sum_{k=1}^{\infty} kP(K_n = k) \geq P(K_n = 1)$$

And because we have the same assumptions as in theorem 2.4 that stated that $P(K_n = 1) = 1$ we see that:

$$\lim_{n\to\infty} E[K_n] \geq \lim_{n\to\infty} P(K_n = 1) = 1$$

Now we use that the equations for $E[K_n](3)$ and $P(K_n = 1)(4)$ are very similar, so they can be manipulated in the same way. We carry out a similar computation as for $P(K_n = 1)$. As follows;

$$E[K_n] = n\sum_{j=1}^{\infty} p_j P_j^{n-1}$$

$$= n\left(\sum_{j=1}^{m-1} p_j P_j^{n-1} + \sum_{m}^{\infty} p_j P_j^{n-1}\right)$$

Now we can take the limit on both sides:

$$\lim_{n\to\infty} E[K_n] = \lim_{n\to\infty} n\left(\sum_{j=1}^{m-1} p_j P_j^{n-1} + \sum_{j=m}^{\infty} p_j P_j^{n-1}\right)$$

We see that:

$$\lim_{n\to\infty} \sum_{j=1}^{m-1} p_j P_j^{n-1} = 0$$

Because we can interchange limit and sum (our sum is finite) and:

$$\lim_{n\to\infty} n P_j^{n-1} = 0$$

So to compute $\lim E[K_n]$ we only need to compute

$$\lim_{n\to\infty} n \sum_{j=m}^{\infty} p_j P_j^{n-1}$$

But we already saw that

$$n\sum_{j=m}^{\infty} p_j P_j^{n-1} \leq \frac{1}{1-\epsilon} n \sum_{m+1}^{\infty} p_j P_{j-1}^{n-1} \leq \frac{1}{1-\epsilon} P(K_n = 1)$$

in the proof above. So,

$$\lim_{n\to\infty} E[K_n] \leq \frac{1}{1-\epsilon} \lim_{n\to\infty} P(K_n = 1) = \frac{1}{1-\epsilon}$$

for every $\epsilon > 0$. So we may conclude that

$$\lim_{n\to\infty} E[K_n] = 1$$

□

2.2 Råde for a Geometric distribution

The tossing of a coin until heads arise could be represented by a Geometric distribution with parameter p, the probability of tossing heads.

We need some inequalities to prove a theorem concerning the probability that the final toss will be done with one coin.

Lemma 2.5. *If* $0 \leq s \leq 1$ *and* $k \geq 1$*:*

$$-ks^2e^{-ks} \leq (1-s)^k - e^{-ks} \leq 0 \qquad 0 \leq s \leq 1, k \geq 1 \tag{10}$$

Proof. We'll prove this inequality in two steps. First we'll prove the right part,

$$\begin{array}{ll} (1-s)^k - e^{-ks} & \leq 0 \\ (1-s)^k & \leq e^{-ks} \\ (1-s) & \leq e^{-s} \end{array}$$

Now we notice that e^{-s} is a convex function, and $(1-s)$ is the tangent to e^{-s} in $s = 0$. Because a tangent line to a convex function is always below the graph. We can also illustrate this in the following figure.

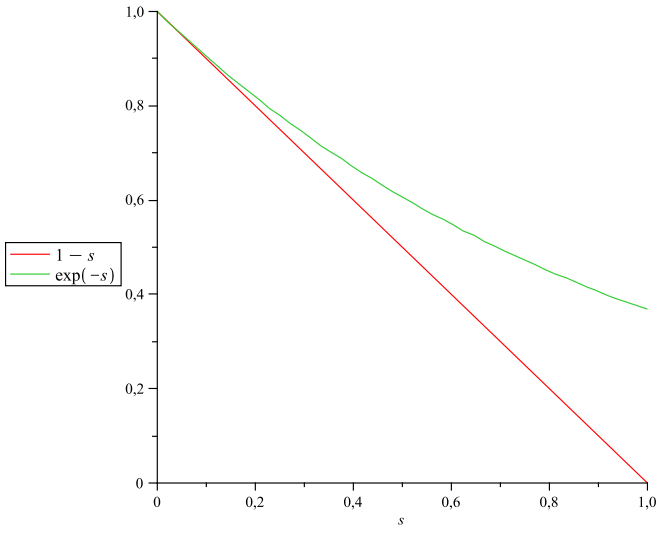


Figure 1: $1 - s$ versus e^{-s}

Now we'll have to prove the second (left) part:

$$\begin{array}{ll} -ks^2e^{-ks} & \leq (1-s)^k - e^{-ks} \\ (1-ks^2)e^{-ks} & \leq (1-s)^k \\ (1-ks^2) & \leq (1-s)^k e^{ks} \\ (1-ks^2)^{\frac{1}{k}} & \leq (1-s)e^s \end{array}$$

To make this expression easier we'll show that it is enough to check the case that $k = 1$ because of the following inequality:

$$(1-ks^2)^{\frac{1}{k}} \leq (1-s^2)$$

Which we rewrite as

$$(1-kx) \leq (1-x)^k \qquad (x = s^2)$$

We'll be using convexity once more. The left part, $1-kx$ is linear. The right part, $(1-x)^k$, has second derivative $k(1-k)(1-x)^{k-2}$ with respect to k. Because $x = s^2$, and $0 \leq s \leq 1$, we'll see that $0 \leq x \leq 1$. By using that $k \geq 1$ we'll see that:

$$k(1-k)(1-x)^{k-2} \leq 0$$

The slope of the function $(1-x)^k$ is $-k$ in $x = 0$ which is the same as the slope of the linear function $1-kx$ in $x = 0$. Using this and the fact that $(1-x)^k$ is convex we've shown that only the case $k = 1$ is relevant. So we have to check that

$$(1-s^2) \leq e^s(1-s)^2$$

Which is equivalent to

$$(1+s) \leq e^s$$

We use (again) that e^s is an convex function and that $(1+s)$ is a tangent line to e^s when $s = 0$. We can illustrate this in the following figure:

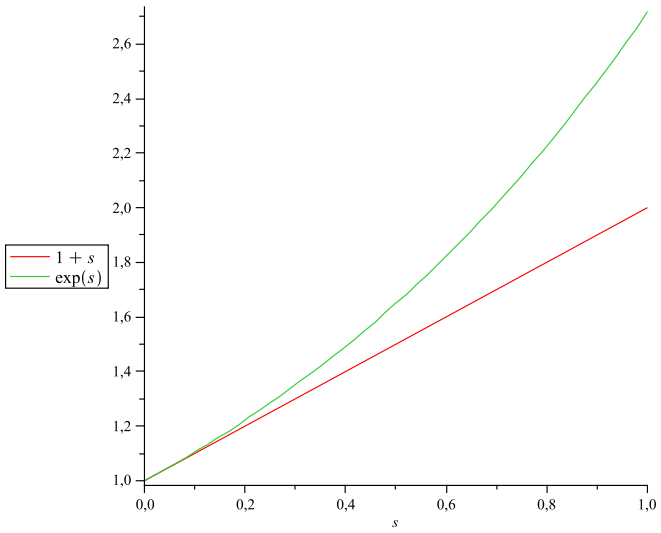


Figure 2: $1+s$ versus e^s

And with this result the inequality is proven. □

We want a formula that describes the probability (and its periodicity) for tossing with one coin in the final toss. The equation we want to prove is:

$$P(K_n = 1) = p \sum_{l=-\infty}^{\infty} e^{-\lambda(l-\theta_n)} e^{-e^{-\lambda(l-\theta_n)}} + O(n^{-1})$$

The next lemma is an inequality which we need to to prove this.

Lemma 2.6. *If we'll take* $u_j(t) = e^{-jt}\exp(-e^{-t})$ *and* $-\lambda(l-\theta_n) \leq 1$, *then:*

$$-n^{-1}u_2(\lambda(l-\theta_n)) \leq (1-n^{-1}e^{-\lambda(l-\theta_n)})^{n-1} - \exp(-e^{-\lambda(l-\theta_n)}) \leq en^{-1}u_2(\lambda(l-\theta_n)). \quad (11)$$

Proof. By taking $s = n^{-1}\exp(-\lambda(l-\theta_n))$ in (10) we'll get:

$$-k(n^{-1}\exp(-\lambda(l-\theta_n)))^2 e^{-k(n^{-1}\exp(-\lambda(l-\theta_n)))} \leq (1-n^{-1}\exp(-\lambda(l-\theta_n)))^k - e^{-k(n^{-1}\exp(-\lambda(l-\theta_n)))} \leq 0$$

Now we'll take $k = n$:

$$-n^{-1}\exp(-2\lambda(l-\theta_n))e^{-\exp(-\lambda(l-\theta_n)))} \leq (1-n^{-1}\exp(-\lambda(l-\theta_n)))^n - e^{-\exp(-\lambda(l-\theta_n))} \leq 0$$

By using that $u_j(t) = e^{-jt}\exp(-e^{-t})$:

$$-n^{-1}u_2(\lambda(l-\theta_n)) \leq (1-n^{-1}\exp(-\lambda(l-\theta_n)))^n - e^{-\exp(-\lambda(l-\theta_n))} \leq 0$$

By subtracting one term in the second part of the equation:

$$-n^{-1}u_2(\lambda(l-\theta_n)) \leq (1-n^{-1}\exp(-\lambda(l-\theta_n)))(1-n^{-1}\exp(-\lambda(l-\theta_n)))^{n-1} - e^{-\exp(-\lambda(l-\theta_n))} \leq 0$$

Now we'll bring one term to the other side, and forget about the most left part for a while:

$$(1-n^{-1}\exp(-\lambda(l-\theta_n)))^{n-1} - e^{-\exp(-\lambda(l-\theta_n))} \leq n^{-1}\exp(-\lambda(l-\theta_n))(1-n^{-1}\exp(-\lambda(l-\theta_n)))^{n-1}$$

To prove inequality (11) we'll need that:

$$n^{-1}\exp(-\lambda(l-\theta_n))(1-n^{-1}\exp(-\lambda(l-\theta_n)))^{n-1} \leq en^{-1}u_2(\lambda(l-\theta_n))$$

This can be simplified to:

$$\begin{aligned} n^{-1}\exp(-\lambda(l-\theta_n))(1-n^{-1}\exp(-\lambda(l-\theta_n)))^{n-1} &\leq en^{-1}e^{-2(\lambda(l-\theta_n))}\exp(-e^{-\lambda(l-\theta_n)}) \\ e^{\lambda(l-\theta_n)}n^{-1}\exp(-2\lambda(l-\theta_n))(1-n^{-1}\exp(-\lambda(l-\theta_n)))^{n-1} &\leq en^{-1}e^{-2(\lambda(l-\theta_n))}\exp(-e^{-\lambda(l-\theta_n)}) \end{aligned}$$

From the definition of the exponential function we know that:

$$\exp(x) = \lim_{n\to\infty}\left(1+\frac{x}{n}\right)^n$$

So if we'll take $x = -e^{-\lambda(l-\theta_n)}$ and we'll use the fact that $\left(1+\frac{x}{n}\right)^n$ is a monotone increasing function, we'll find:

$$e^{\lambda(l-\theta_n)}n^{-1}e^{-2\lambda(l-\theta_n)}(1-n^{-1}\exp(-\lambda(l-\theta_n)))^{n-1} \leq e^{\lambda(l-\theta_n)}n^{-1}e^{-2\lambda(l-\theta_n)}\exp(-e^{-\lambda(l-\theta_n)})$$

To prove inequality (11), the last thing we'll need is that:

$$e^{-\lambda(l-\theta_n)} \leq e$$

Which is true because of the assumption we made in the lemma.

□

With this inequality we're able to give an 'exact' formula for the probability that there is exactly 1 coin that need the most tosses to become heads. Because we're tossing coins by a geometric distribution we can slightly change our notation of (1):

$$p_j = p(1-p)^{j-1}, \quad \bar{P}_j = (1-p)^{j-1}, \quad P_{j-1} = 1-\bar{P}_j \quad (j = 1, 2, \cdots). \tag{12}$$

We also define $\{a\}$ as the fractional part of a. The following theorem shows that there is a **certain periodicity** in the probability for ending up with 1 coin in the last toss as function of n, the total number of coins.

Theorem 2.5.

$$P(K_n = 1) = p \sum_{l=-\infty}^{\infty} e^{-\lambda(l-\theta_n)} e^{-e^{-\lambda(l-\theta_n)}} + O(n^{-1}) \tag{13}$$

as $n \to \infty$, *where* $\lambda = -\log(1-p)$ *and* $\theta_n = \{\lambda^{-1} \log n\}$

Proof. If we substitute $j-1$ for j in (4), take $m = [\lambda^{-1} \log n]$ and we'll take $p = 1 - e^{-\lambda}$:

$$\begin{aligned} P(K_n = 1) &= np \sum_{j=0}^{\infty} e^{-\lambda j}(1 - e^{\lambda j})^{n-1} = p \sum_{j=0}^{\infty} e^{-\lambda j + \log n} \left(1 - \frac{1}{n} e^{-\lambda j + \log n}\right)^{n-1} \\ &= p \sum_{l=-m}^{\infty} e^{-\lambda(1-\theta_n)} \left(1 - \frac{1}{n} e^{-\lambda(l-\theta_n)}\right)^{n-1} \end{aligned}$$

We now define $u_j(t) = e^{-jt} \exp(-e^{-t})$, as in Lemma 2.6 and

$$F_j(n, \lambda) = \sum_{l=\infty}^{\infty} u_j(\lambda(l - \theta_n)). \tag{14}$$

We would like to say something about the value of:

$$\sum_{l=-\infty}^{-m-1} u_1(\lambda(l - \theta_n))$$

So we're able to extend the sum to a sum from minus infinity to infinity by only adding something that is of order $O(n^{-1})$.

Because $u_1(t)$ is a monotone increasing positive function on $(-\infty, 0)$:

$$\sum_{l=-\infty}^{-m-1} u_1(\lambda(l - \theta_n)) \leq \int_{-\infty}^{-m} u_1(\lambda(s - \theta_n)) ds$$

$$\begin{aligned} \int_{-\infty}^{-m} u_1(\lambda(s - \theta_n)) ds &= \frac{e^{-e^{-\lambda(s-\theta_n)}}}{\lambda} \Big|_{s=-\infty}^{s=-m} \\ &= \frac{e^{-e^{\lambda(m+\theta_n)}}}{\lambda} - \frac{e^{-e^{-\lambda(\infty-\theta_n)}}}{\lambda} \end{aligned}$$

And because $m + \theta_n = \lambda^{-1} \log n$:

$$\sum_{l=-\infty}^{-m-1} u_1(\lambda(l - \theta_n)) \leq \lambda^{-1} e^{-n} \tag{15}$$

Now we can use equation (11) from Lemma (2.6):

$$-n^{-1} u_2(\lambda(l - \theta_n)) \leq (1 - n^{-1} e^{-\lambda(l-\theta_n)})^{n-1} - \exp(-e^{-\lambda(l-\theta_n)}) \leq e n^{-1} u_2(\lambda(l - \theta_n))$$

If we multiply this equation by $p \exp(-\lambda(l - \theta_n))$ we'll get:

$$-pn^{-1} u_3(\lambda(l-\theta_n)) \leq p \exp(-\lambda(l-\theta_n))(1-n^{-1} e^{-\lambda(l-\theta_n)})^{n-1} - pu_1(\lambda - \theta_n) \leq pen^{-1} u_3(-\lambda(l-\theta_n))$$

Now we'll sum all from $l = -m$ to ∞;

$$\sum_{l=-m}^{\infty} -pn^{-1} u_3(\lambda(l - \theta_n)) \leq P(K_n = 1) - \sum_{l=-m}^{\infty} pu_1(\lambda(l - \theta_n)) \leq \sum_{l=-m}^{\infty} pen^{-1} u_3(\lambda(l - \theta_n))$$

We want to add the sum of the $-\infty, -m-1$ part on the left and the right part of the equation, this is only valid if we'll subtract it as well. We immediately use that $F_j(\theta,\lambda) = \sum_{l=-\infty}^{\infty} u_j(\lambda(l-\theta))$:

$$-pn^{-1}F_3(\theta_n,\lambda) - pn^{-1}\sum_{l=-\infty}^{-m-1} u_3(\lambda(l-\theta_n)) \leq P(K_n=1) - pF_1(\theta_n,\lambda) + p\sum_{l=-\infty}^{-m-1} u_1(\lambda(l-\theta_n))$$

and

$$P(K_n=1) - pF_1(\theta_n,\lambda) + p\sum_{l=-\infty}^{-m-1} u_1(\lambda(l-\theta_n)) \leq pen^{-1}F_3(\theta_n,\lambda) - pen^{-1}\sum_{l=-\infty}^{-m-1} u_3(\lambda(l-\theta_n))$$

It's important to notice that for $u_j(t) = e^{-jt}\exp(-e^{-t})$ holds that:

$$u_1(t) > u_2(t) > u_3(t) > \cdots \qquad \text{for } t > 0$$

Using this fact, we can use our result in (15) which gives us:

$$-pn^{-1}F_3(\theta_n,\lambda) - p\lambda^{-1}n^{-1}e^{-n} \leq P(K_n=1) - pF_1(\theta_n,\lambda) \leq pen^{-1}F_3(\theta_n,\lambda)$$

Now we'll use the fact $F_j(\theta,\lambda)$ is bounded for each fixed λ which tells us that:

$$P(K_n=1) = p\sum_{l=-\infty}^{\infty} e^{-\lambda(l-\theta_n)}e^{-e^{-\lambda(l-\theta_n)}} + O(n^{-1})$$

□

3 Råde's Problem

We found a formula for the probability that the last toss has a single coin. We also found that this probability has a "certain periodic behavior". We expect some kind of periodicity in the probability if we change the value of θ_n, which is related to the value of n. If we take $p = \frac{1}{2}$, which is quite normal in real life, and we make a figure of the probability against the value of θ_n, where we should notice that $0 \leq \theta < 1$ for all n, and we ignore the order-fault, we'll get the following figure:

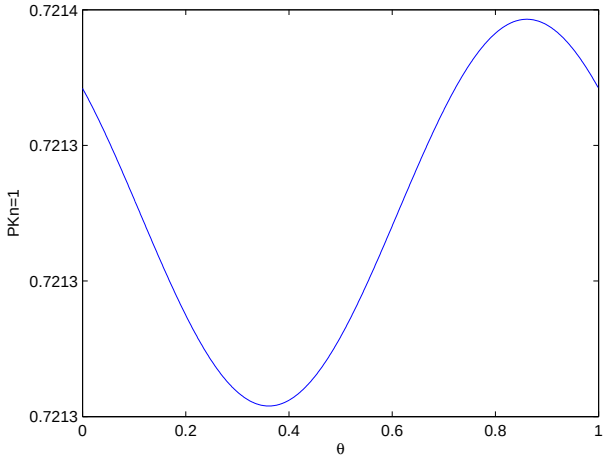


Figure 3: The probability of tossing with 1 coin in the final toss versus θ_n, where $p = \frac{1}{2}$

The maximum amplitude is of order 10^{-6}. This is very small which means that is would be very hard to find this result by simulating. To simulate this difference we could make confidence intervals for the probability that the final toss is with one coin for several values of n.

With Monte Carlo Methods we're able to compute these confidence intervals. To compute a 95% confidence interval of a large set of independent identically distributed random variables $X_1, X_2, \ldots, X_M$ we'll do:

$$\left[a_M - \frac{1.96 b_M}{\sqrt{M}}, a_M + \frac{1.96 b_M}{\sqrt{M}}\right]$$

Where a_M is the mean of the set and b_M the standard deviation.

In other words, to find the same result as in the figure by simulation, we need that $\frac{1.96 b_M}{\sqrt{M}}$ is of order 10^{-6}. This means that we'll need $M = 10^{12}$, i.e. a million times a million, simulations. This would take way to much time to compute.

Another way to find results by simulating that correspond to the formula we found is by choosing a different value of p. If we take, for instance, $p = \frac{3}{4}$ the figure of the probability of ending with one toss versus the value of θ_n can be found in figure 4 (remember that we're ignoring the order fault):

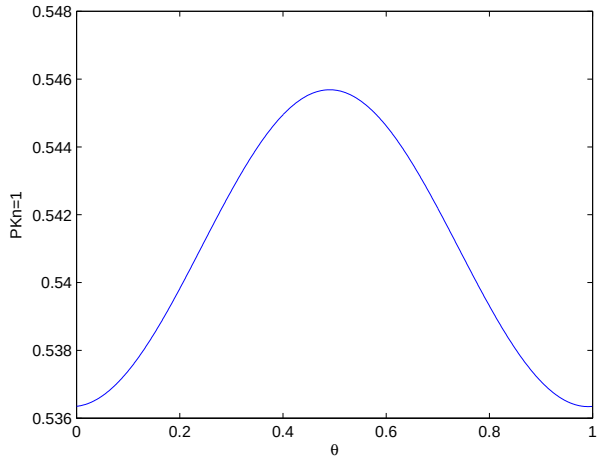


Figure 4: The probability of tossing with 1 coin in the final toss versus θ_n, where $p = \frac{3}{4}$

Now we'll have a amplitude of order 10^{-4}. If we simulate this tossing of coins in Matlab we can find the difference in the probability for the different values of theta. The minimum in the figure can be found at $\theta_n = .49$, the maximum at $\theta_n = 0.99$. We now need coinnumbers that give values of θ_n that are approximately 0.49 and 0.99. If we take $n = 32$ coins and $n = 63$ coins we find the following theta values:

$$\theta_{32} \approx 0.5000 \qquad \theta_{63} \approx 0.9886$$

The results that were find by simulating can be found in the following table:

Number of coins	Value of θ_n	$P(K_n = 1)$ by exact formula	95% Confidence interval
32	0.5000	0.54568	[0.54516 0.54764]
63	0.9886	0.53634	[0.53425 0.53672]

Table 1: Empirical results where $p = \frac{1}{3}$

This shows that there is a difference in the value of $p_n = P(K_n = 1)$ if we change the value of n. This difference grows if we take higher values for p.
As a last example we'll look at the the values of $P(K_n = 1)$ whenever $p = \frac{9}{10}$. The graph of θ_n against $P(K_n = 1)$ is as follows:

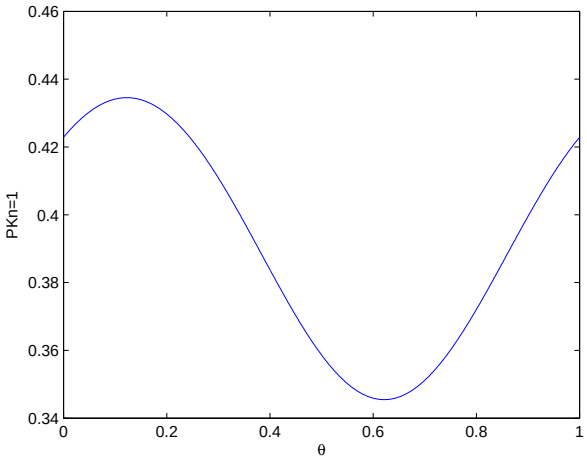


Figure 5: The probability of tossing with 1 coin in the final toss versus θ_n, where $p = \frac{9}{10}$

The absolute difference between the highest and the lowest value of $P(K_n = 1)$ is almost 0.1 which is equal to 10%.

4 Empirical results Råde's problem

The problem that Rade proposed can be solved using Matlab. The two questions were as follows:

1. Find $P(X = i)$ for $i = 1, 2, \cdots, n$ and $E(X)$.
2. Let $p_n = P(X = 1)$. Analyze the behavior of p_n as $n \to \infty$.

The second question is answered in the previous section. We'll only answer the first question here.

4.1 Råde's first question

This can be solved as follows. By simulating the tossing of n coins m times, m outcomes of K_n will be found. With this result it's easy to compute $P(X = i)$ and $E(X)$.
$P(X = i)$ is given by the number of times the outcome i was found divided by the total number of repetitions:

$$P(X = i) = \frac{\#\{K_n = i\}}{m}$$

$E(X)$ is given by the sum of all possible outcomes times the probability on that outcome:

$$E(X) = \sum_{j=1}^{\infty} P(X = j) \cdot \#\{K_n = j\}$$

Because there are finite repetitions, m, there is also a maximum value of K_n, which is finite because the total number of coins, n, is finite. We'll denote the maximum value of K_n with M. Now it is clear that for $i > M$:

$$P(X = i) = \frac{\#\{K_n = i\}}{m} = \frac{0}{m} = 0$$

And for $E(X)$ follows:

$$E(X) = \sum_{j=1}^{M} P(X = j) \cdot \#\{K_n = j\}$$

Which is a finite sum that can be computed numerically.

In table 2 the various outcomes for the number of coins in the last toss can be found. The computations were done 100000 times with 100 (fair) coins. We also included 95% confidence intervals. Some of the left boundaries of our confidence intervals are less then zero. Because we are sure that all probabilities are non negative we take 0 as a left boundary whenever our computations tells us that they are less then zero.

X	$\#\{K_n = X\}$	$P(K_n = x)$	95% Confidence interval	P by exact formula
1	71993	0.71993	[0.7144 , 0.7255]	0.7215
2	18150	0.18150	[0.1759 , 0.1871]	0.1804
3	6042	0.06042	[0.0548 , 0.0660]	0.0601
4	2200	0.02200	[0.0164 , 0.0276]	0.0225
5	937	0.00973	[0.0038 , 0.0149]	0.0090
6	383	0.00383	[0.0000 , 0.0094]	0.0038
7	164	0.00164	[0.0000 , 0.0072]	0.0016
8	73	0.00073	[0.0000 , 0.0063]	0.0007
9	32	0.00032	[0.0000 , 0.0059]	0.0003
10	14	0.00014	[0.0000 , 0.0057]	0.0001
11	9	0.00009	[0.0000 , 0.0057]	0.0001
12	1	0.00001	[0.0000 , 0.0056]	0.0000
13	1	0.00001	[0.0000 , 0.0056]	0.0000
14	1	0.00001	[0.0000 , 0.0056]	0.0000
15	0	0.00000	[0.0000 , 0.0056]	0.0000

Table 2: Results of the probability to do the final toss with x coins

The first question of Råde is also about the expected number of coins in the last toss. Because we know that the expected value should converge to 1 whenever the total number of coins, n, goes to infinity we first look at relative low total numbers of coins. After that we increase the total number of coins more rapidly to see that it converges.
We did $M = 100000$ repetitions. The results can be found in table 3:

Number of coins	E(X)	95% Confidence interval
16	1.4432	[1.4376,1.4488]
23	1.4396	[1.4340,1.4451]
32	1.4416	[1.4360,1.4471]
45	1.4478	[1.4421,1.4534]
64	1.4416	[1.4361,1.4472]
91	1.4442	[1.4386,1.4499]
128	1.4438	[1.4382,1.4493]
181	1.4403	[1.4347,1.4458]
10^4	1.4345	[1.4289,1.4401]
10^5	1.4298	[1.4241,1.4355]
10^6	1.4102	[1.4046,1.4158]
10^7	1.4102	[1.3896,1.4008]

Table 3: Results for the expected number of tosses for various number of coins

We'll see that for small numbers of n the expected number of tosses is approximately the same. But when n increases we see that the expected value decreases. In corollary 2.3 we saw that $E[K_n] = 1$ as $n \to \infty$.

5 Further questions

5.1 A guessing strategy due to Råde's Problem

Our interpretation of Råde's Problem is that of a high-low guessing game where the secret number is zero. The question we'll asked ourself was:

If you use the coin tossing strategy, what is the probability that you guess 1 *before guessing* 0*, and how does this relate to the number of coins?*

We can slightly modify this question into the following question:

If we use the coin tossing strategy, which secret number takes the most guesses, 0 *or* 1*, and how does this relate to the number of coins?*

The guessing game and strategy can be summarized in the following way (and example of it is presented in section 1.1):

Suppose someone (say Hider) chooses a fixed number between zero and n, $n > 0$. Another person (say Guesser) wants to guess the secret number by tossing n coins and takes the number of heads as a first guess. Now there are 3 possible outcomes:

1. The number of heads is equal to the secret number; the game ends
2. The number of heads is lower than the secret number; Guesser tosses all of the coins that show tails again. The new guess is the total number of heads (the old heads added with the new coins that show heads)
3. The number of heads is higher than the secret number; Guesser tosses all of the coins that show heads again. The new guess is the new total number of heads.

The strategy for the event that the secret number is 0 is as in Råde's Problem. First you toss all the coins. The first number you guess is equal to the number of coins that show heads. After that you retoss all the coins that show heads and take the new number of coins that show heads as a second guess. This goes on until all coins show tails which means that you guess 0. An example is: First we have 100 coins. We toss them all and end up with 48 coins that show heads. We retoss and get 27 coins that show heads. After that we guess 14, 6, 3, 1 and 0. This means that this time it took 7 guesses (48, 27, 14, 6, 3, 1, 0) to get to the secret number .

The strategy for the event that the secret number is 1 is slightly different. As long as the secret number 1 is lower then the number of coins that show heads (i.e. the number that is guessed) we do the same thing. However, it is possible to guess 0 before we guessed 1. For instance, if we toss 2 coins that both show tails we'll guess 0. When this happens we'll retoss all the coins in the last toss. An example for this case is: First we have 100 coins. We toss them all and end up with 50 coins that show heads. We retoss and get 22 coins that show heads. After that we guess 12, 6, 2, 2, 0, 1. This means that this time it took 8 guesses (50, 22, 12, 6, 2, 2, 0, 1) to guess the secret number.

5.1.1 With one coin

Our question is easily answered for low total numbers of coins. If we take $n = 1$ coin, so the possible secret numbers are 0 and 1, and we call X_i^j the number of tosses that are needed to guess secret number i while starting with j coins. This means that $E[X_i^j]$ is the expected number of tosses to guess i whenever we start with j coins, then we'll get the following equations for the expected number of tosses.

$$E[X_0^1] = 1/2 + 1/2\left(1 + E[X_0^1]\right) \qquad \rightarrow \qquad E[X_0^1] = 2$$

and,

$$E[X_1^1] = 1/2 + 1/2\left(1 + E[X_1^1]\right) \qquad \rightarrow \qquad E[X_1^1] = 2$$

We should notice that there is a recursion in these cases. If we are tossing with 1 coin there are two possibilities. Either you toss the required number of 'head-coins' at once or you toss the wrong number and you return to the first case where you are tossing with 1 coin. We'll see this recursion in both the equations and we conclude that, when tossing with 1 coin, both secret numbers take the same expected number of tosses. Now we'll do the same as above but with two coins:

5.1.2 With two coins

$$E[X_0^2] = 1/4 + 1/2(1 + E[X_0^1]) + 1/4(1 + E[X_0^2])$$

To understand this equation it is important to notice that with probability $(1/2)^2$ we toss two times tails. This means that with probability 1/4 we'll guess the secret number the first time. With probability $(1/2)^2+(1/2)^2$ we toss one time tails and one time heads (or one time heads and one time tails) which brings us back to the case where the secret number is 0 and we toss with 1 coin. We already now that it takes two tosses to guess the correct number in that case. This means that in this case the total expected number of tosses is 3 (1+2) tosses. With probability $(1/2)^2$ we toss two times heads which brings us back to the same problem, we still have 2 coins, so with probability 1/4 we need $(1 + E_2[X_0])$ tosses. Using this we get the following equation:

$$E[X_0^2] = 1/4 + 1/2(1 + 2) + 1/4(1 + E[X_0^2]) \qquad \rightarrow \qquad E[X_0^2] = 8/3$$

and,

$$E[X_1^2] = 1/4(1 + E[X_1^2]) + 1/2 + 1/4(1 + E[X_1^2]) \qquad \rightarrow \qquad E[X_1^2] = 2$$

We see that in this case it takes a little more guesses to find the secret number 0 then the secret number 1. We'll do the same for the next case, $n = 3$:

5.1.3 With three coins

$$\begin{aligned} E[X_0^3] &= 1/8 + 3/8(1 + E[X_0^1]) + 3/8(1 + E[X_0^2]) + 1/8(1 + E[X_0^3]) \\ &= 1/8 + 3/8(1 + 2) + 3/8(1 + 8/3) + 1/8(1 + E[X_0^3]) \qquad \rightarrow \qquad E[X_0^3] = 22/7 \end{aligned}$$

and,

$$\begin{aligned} E[X_1^3] &= 1/8(1 + E[X_1^3]) + 3/8 + 3/8(1 + E[X_1^2]) + 1/8(1 + E[X_1^3]) \\ &= 1/8(1 + E[X_1^3]) + 3/8 + 3/8(1 + 2) + 1/8(1 + E[X_1^3]) \qquad \rightarrow \qquad E[X_1^3] = 7/3 \end{aligned}$$

5.1.4 With n coins

We see a certain recursion for both expected number of tosses when the total number of coins increases. The recursion for the secret number 0 is the following:

$$E[X_0^n] = \frac{1}{2^n} + \sum_{k=1}^{n} \frac{\binom{n}{k}}{2^n}(1 + E[X_0^k]) = 1 + \sum_{k=1}^{n} \frac{\binom{n}{k}}{2^n} E[X_0^k] \tag{16}$$

And for the secret number 1 we get:

$$E[X_1^n] = \frac{1}{2^n}(1 + E[X_1^n]) + \frac{n}{2^n} + \sum_{k=2}^{n} \frac{\binom{n}{k}}{2^n}(1 + E[X_1^k]) = 1 + \frac{E[X_1^n]}{2^n} + \sum_{k=2}^{n} \frac{\binom{n}{k}}{2^n} E[X_1^k] \tag{17}$$

Using this two formulas we can easily find the solution for different values of n. Some results can be found in the following table:

n	$E[X_0^n]$	$E[X_1^n]$
1	2.0000	2.0000
2	2.6667	2.0000
3	3.1429	2.3333
4	3.5048	2.6667
5	3.7942	2.9556
6	4.0348	3.2000
7	4.2408	3.4085
8	4.4211	3.5894
9	4.5813	3.7494
10	4.7256	3.8932
50	6.9910	6.1582
100	7.9838	7.1511

Table 4: Expected number of tosses to find the secret numbers 0 and 1

The expected values are easy to find by doing a simulation in Matlab. If we for instance take $n = 100$ coins, do $M = 100000$ simulations and run a program that simulates the tossing of coins for the different secret numbers 0 and 1 we'll get the following results, we also include the answers found by our recursive formulas:

Secret number	Recursive formulas	Simulation	95% Confidence interval
0	7.9838	7.9831	[7.9715 , 7.9947]
1	7.1511	7.1486	[7.1377 , 7.1594]

Table 5: The number of guesses/tosses that are needed to guess the secret number

The results that we found by doing a simulation are (almost) the same as our results from our recursive formulas.

We see in table 5.1.4 that the expected number of tosses for the secret number 0 is consequently higher then the expected number of tosses for the secret number 1. An interesting question is how the relative difference between these two values behave. We could plot this relative difference $\frac{E[X_0^n]}{E[X_1^n]}$ for $n = 1 : 100$:

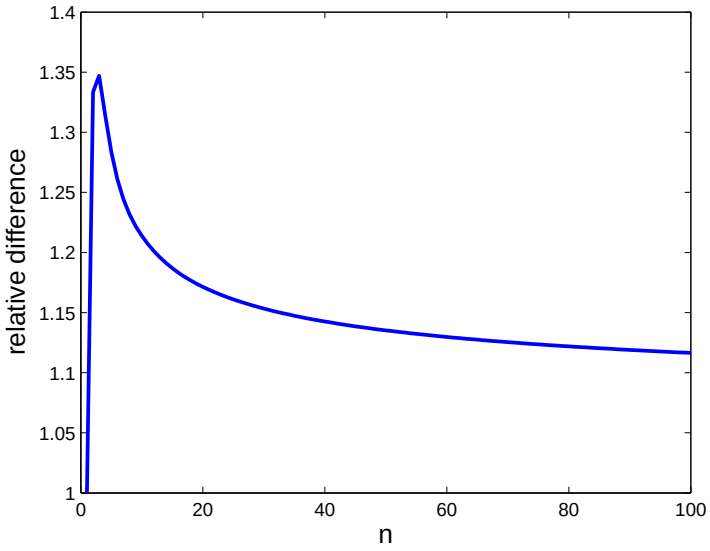


Figure 6: $E[X_0^n]$ divided by $E[X_1^n]$

It seems that the relative difference converge to a certain value. Interesting questions are; *At which rate does the relative difference converge?* and *What happens to the relative difference if we take* $n \to \infty$*?*.

To answer these questions we look at the grow rate of $E[X_0^n]$. We now that after each time you toss the coins approximately one half will be heads and one half will be tails. So it is interesting to compare the values of $E[X_0^n]$ with the values of $\log_2(n)$, **where** $\log_2(\mathbf{n})$ **is the base 2 logarithm of n**. There are two possible ways we could compare these two: We could look at the absolute and the relative difference in the growth of these 2 functions:

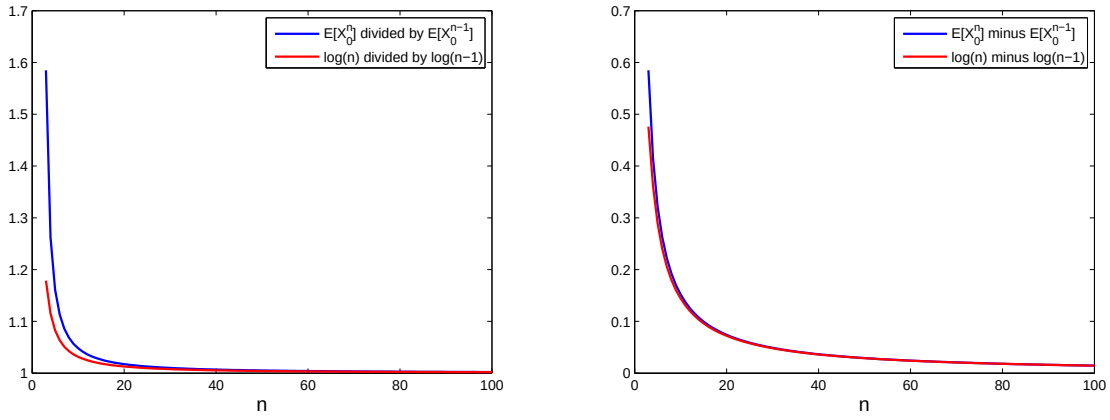


Figure 7: The difference between the growth of $E[X_0^n]$ and the growth of $\log_2(n)$

It seems that the two functions grow with the same rate when n increases. We could look at the absolute difference between $E[X_0^n]$ and $\log_2(n)$ as n increases. We'll do the same thing for $E[X_1^n]$. The absolute difference can be seen in the following figure:

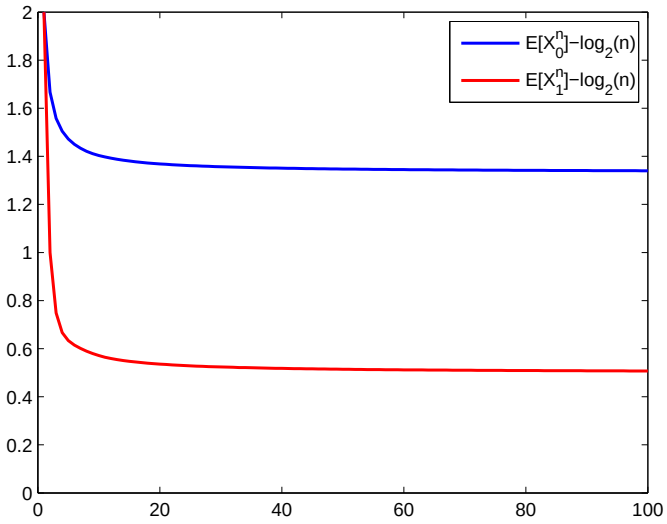


Figure 8: The absolute difference between $E[X_0^n]$ and $E[X_1^n]$ with $\log_2(n)$

From the figures it is clear that as n increases the absolute difference between $E[X_0^n]$ and $E[X_1^n]$ with $\log_2(n)$ converge to a certain value. These values are approximately: 1.3399 and 0.5072. This means that the absolute difference between $E[X_0^n]$ and $E[X_1^n]$ is:

$$E[X_0^n] - E[X_1^n] = (E[X_0^n] - \log_2(x)) - (E[X_1^n] - \log_2(x)) \approx 1.3399 - 0.5072 = 0.8327$$

This can be seen from the following figure:

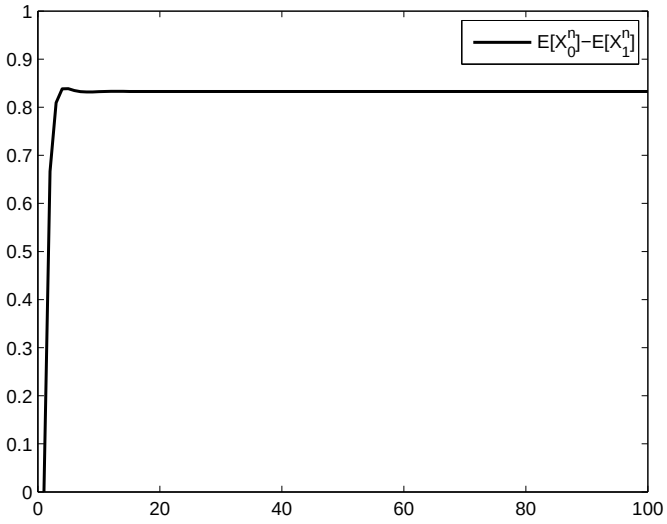


Figure 9: The absolute difference between $E[X_0^n]$ and $E[X_1^n]$

If we combine these two results (the result that says that for $E[X_0^n]$ and $E[X_1^n]$ holds that they have the same growth rate as $\log_2(n)$, and the result that gave us the absolute differences

between $E[X_0^n]$ and $E[X_1^n]$) we're able to answer our 2 questions. We know that:

$$\lim_{n\to\infty} \frac{E[X_0^n]}{E[X_1^n]} \approx \lim_{n\to\infty} \frac{\log_2(n) + 1.3399}{\log_2(n) + 0.5072} = 1$$

This means that when n increases the relative difference of $E[X_0^n]$ and $E[X_1^n]$ converge to 1. In other words; For big n the secret numbers 0 and 1 need the same expected number of guesses and are equally hard to guess. The rate of convergence is very low. If we for instance allow the relative difference to be 1.05 we'll need $n = 72556$ coins.

These results are found by looking at a graph. It is more convenient to find a actual proof. We'll try to proof this by using an inequality on the value $E[X_0^n]$:

Theorem 5.1.

$$\log_2(k) \quad \leq \quad E[X_1^k] \quad \leq \quad E[X_0^k] \quad \leq \quad \log_2(k) + 2 \tag{18}$$

Proof. We'll first proof the right part of equation 18:

To simplify our problem we'll denote $E[X_0^n]$ by $f(n)$, where

$$f(k) = 1 + E[f(X)] \qquad \text{where} \qquad X \sim Bin\left(k, \frac{1}{2}\right)$$

We define

$$f(0) = 0$$

This means that for $f(1) = E[X_0^1]$ holds that:

$$f(1) = 1 + E[f(X)] = 1 + \frac{1}{2}f(0) + \frac{1}{4}f(1) \qquad \rightarrow \qquad f(1) = 2$$

Our hypothesis is:

$$f(k) \leq \log_2(k) + 2$$

First we observe that this inequality holds for $k = 1$:

$$f(1) = 2 \leq \log_2(1) + 2 = 2$$

We assume that the statement holds for some k. Now we'll prove it for $k + 1$:

$$f(k+1) = 1 + E[f(X)] \qquad \text{where} \qquad X \sim Bin\left(k+1, \frac{1}{2}\right)$$

We can split our expectation into two conditional expectations, times the probability that the events happen:

$$f(k+1) = 1 + E[f(X)] = 1 + E[f(X)|X \leq k]P(X \leq k) + E[f(X)|X = k+1]P(X = k+1)$$

This can be simplified to:

$$f(k+1) = 1 + E[f(X)|X \leq k]P(X \leq k) + f(k+1)\frac{1}{2^{k+1}}$$

Now we'll bring the $f(k+1)$ term to the other side:

$$\frac{2^{k+1} - 1}{2^{k+1}} f(k+1) = 1 + E[f(X)|X \leq k]P(X \leq k)$$

And because the events $X \leq k$ and $X = k+1$ have complementary probabilities we know that:

$$P(X \leq k) = 1 - P(X = k+1) = 1 - \frac{1}{2^{k+1}}$$

This can be substituted in the above equation:

$$\frac{2^{k+1}-1}{2^{k+1}} f(k+1) = 1 + E[f(X)|X \leq k] \left(1 - \frac{1}{2^{k+1}}\right)$$

Now we'll use that our formula holds for k:

$$E[f(X)|X \leq k] \quad \leq \quad E[\log_2(X) + 2|X \leq k] \quad = \quad E[\log_2(X)|X \leq k] + 2$$

Now we use Jensen's inequality for conditional expectations [6] where we should notice that $\log_2(X)$ is a concave function.

$$E[\log_2(X)|X \leq k] + 2 \quad \leq \quad \log_2\left(E[X|X \leq k]\right) + 2$$

Our question is now: What is $E[X|X \leq k]$. We should notice that X is $Bin\left(k+1, \frac{1}{2}\right)$ distributed and use conditional probabilities to compute this conditional expectation:

$$E[X|X \leq k] = \sum_{j=0}^{k} jP(X = j|X \leq k) = \sum_{j=0}^{k} j\frac{P(X = j)}{P(X \leq k)}$$

Now we can take $\frac{1}{P(X \leq k)}$ out of the summation because it doesn't depend on j:

$$E[X|X \leq k] = \frac{1}{P(X \leq k)} \sum_{j=0}^{k} jP(X = j)$$

And because

$$E[X] = \sum_{j=0}^{k+1} jP(X = j) = \frac{k+1}{2}$$

we can take:

$$\sum_{j=0}^{k} jP(X = j) = E[X] - (k+1)P(X = k+1)$$

And so,

$$E[X|X \leq k] = \frac{1}{P(X \leq k)} \left(\frac{k+1}{2} - (k+1)P(X = k+1)\right)$$

Which makes:

$$E[X|X \leq k] = \frac{1}{\frac{2^{k+1}-1}{2^{k+1}}} \left(\frac{k+1}{2} - \frac{k+1}{2^{k+1}}\right) = \frac{2^{k+1}}{2^{k+1}-1} \left(\frac{k+1}{2} - \frac{k+1}{2^{k+1}}\right) = \frac{(k+1)\left(2^k - 1\right)}{2^{k+1}-1}$$

If we put these results together we get:

$$\frac{2^{k+1}-1}{2^{k+1}} f(k+1) \leq 1 + \left(\log_2\left(\frac{(k+1)\left(2^k-1\right)}{2^{k+1}-1}\right) + 2\right)\left(1 - \frac{1}{2^{k+1}}\right)$$

We now multiply both sides with $\frac{2^{k+1}}{2^{k+1}-1}$ and use that $\log_2(xy) = \log_2(x) + \log_2(y)$:

$$\begin{aligned} f(k+1) &\leq \log_2\left(\frac{(k+1)\left(2^k-1\right)}{2^{k+1}-1}\right) + 2 + \frac{2^{k+1}}{2^{k+1}-1} \\ &\leq \log_2(k+1) + \log_2\left(\frac{2^k-1}{2^{k+1}-1}\right) + 2 + \frac{2^{k+1}}{2^{k+1}-1} \end{aligned}$$

We want to prove that:

$$f(k+1) \leq \log_2(k+1) + 2$$

To prove this we'll need that:

$$\log_2\left(\frac{2^k-1}{2^{k+1}-1}\right) + \frac{2^{k+1}}{2^{k+1}-1} \leq 0$$

So:

$$\begin{aligned} \frac{2^{k+1}}{2^{k+1}-1} &\leq -\log_2\left(\frac{2^k-1}{2^{k+1}-1}\right) \\ \frac{2^{k+1}}{2^{k+1}-1} &\leq \log_2\left(\frac{2^{k+1}-1}{2^k-1}\right) \end{aligned}$$

Now we can simplify both sides:

$$\frac{2^{k+1}}{2^{k+1}-1} = \frac{2^{k+1}}{2^{k+1}-1} - \frac{1}{2^{k+1}-1} + \frac{1}{2^{k+1}-1} = 1 + \frac{1}{2^{k+1}-1}$$

and

$$\log_2\left(\frac{2^{k+1}-1}{2^k-1}\right) = \log_2\left(2\left(\frac{2^k-1/2}{2^k-1}\right)\right) = \log_2\left(2\left(\frac{2^k-1}{2^k-1}\right) + \frac{1}{2^k-1}\right)$$

which means that we get:

$$1 + \frac{1}{2^{k+1}-1} \leq \log_2\left(2 + \frac{1}{2^k-1}\right)$$

If we'll take $x = 1 + \frac{1}{2^{k+1}-1}$ and $y = 1 + \frac{1}{2^k-1}$ our equation becomes:

$$x \leq \log_2(1+y)$$

And because $k \geq 1$, it is clear that:

$$x < y$$

Using this and the fact that $1 \leq x \leq 2$, we see that $x \leq y \leq 2$. So it is enough to prove that

$$x \leq \log_2(1+y) \qquad \text{where} \qquad 1 \leq x < y \leq 2$$

By plotting $f(x) = x - 1$ and $g(x) = \log_2(x)$ we see that our inequality holds for $1 \leq x \leq 2$.

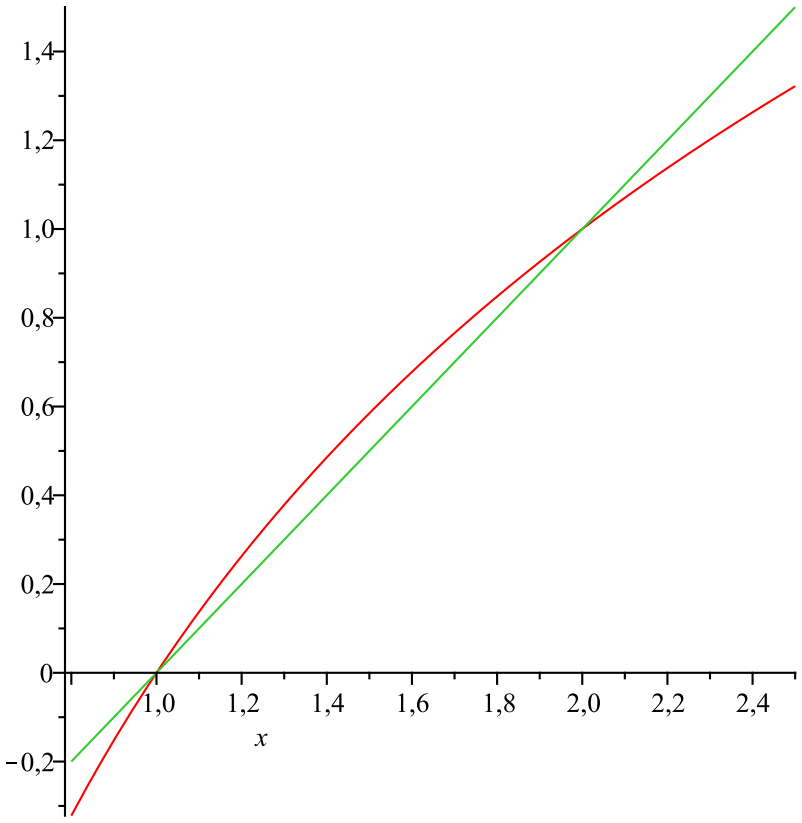


Figure 10: $f(x)$ and $g(x)$

We can also prove this by simple reasoning. The two formulas are equal whenever $x = 1$ and $x = 2$. Because $\log_2$ is a convex function, it is clear that all points on the straight line $x - 1$ are beneath $g(x)$. And with this result we proved our inequality. □

We 'prove' the other part of inequality 18 by simple reasoning:

Proof.

$$\log_2(k) \leq E[X_0^k]$$

$\log_2(k)$ could be seen as the guessing strategy where you always pick the number that is exactly in between the two boundaries. Our coin tossing strategy does something similar. The only difference between these two strategies is that with our coin tossing strategy there is a probability to guess the same number two (or ever more times) in a row. It is clear that the coin tossing strategy is less accurate. □

It seems obvious that we could make another inequality for the expected number of tosses when the secret number is 1. This expected value is also related to the base 2 logarithm. If we could find an upper- an an lower-bound for the expected value that is related to the base 2 logarithm ± a constant C (which would be negative on one side and positive on the other side) then we could easily prove that the relative difference of these two expected values converge to 1. Unfortunately there wasn't enough time to prove this.

5.2 Another guessing strategy

To come back at our guessing game: The guessing strategy that was based on the problem of Råde is not very efficient. It may happen that you guess the same number twice in a row. Of course, the probability is very low if n is large. Only when you get close to guessing the secret number this problem occur. We could make up a new strategy that is almost the same as the first one but takes out the probability to guess a number twice. We do this by turning one coin after each guess as follows; Suppose the secret number is 0 and we are tossing with 100 coins. After the first toss we've 50 coins that show heads. We hear that the secret number is lower and turn one of the head coins to become a tail coin. After that we'll toss the 49 coins that are left once more. This way it is impossible to toss 50 head coins another time. Now suppose the secret number is 1. After some guesses we've 4 coins left. We'll toss all of them and get 4 coins that show tails, we guess 0 and the 'Hider' tells us that the secret number is higher. Now we'll turn one coin from tails to heads and toss the other 3 coins again. From now one the lowest possible secret number is 1. This way it is impossible to guess 0 once more. Just as in the above section we'll compute the first 2 expected numbers by direct calculation and after that we'll make an direct formula for higher numbers of total coins.

5.2.1 With one coin

If the secret number is 0 and we'll toss with 1 coin, so the possible secret numbers are 0 and 1, we'll get as an expected number of tosses:

In words; With probability 1/2 we'll guess 0 at once and with probability 1/2 we'll guess 1 and turn that single coin to tails. The next toss is done with 0 coins which means the second guess will be 0. If we take these two values together we'll get:

$$E[X_0^1] = 1/2 + 1/2(1+1) = 3/2$$

And for the secret number 1:

$$E[X_1^1] = 1/2(1+1) + 1/2 = 3/2$$

Now we'll do the same computations for the case that we have 2 coins:

5.2.2 With two coins

If the secret number is equal to 0 we guess 0 with probability $(1/2)^2 = 1/4$. With probability 1/2 we toss one head and one tail coin. The 'Hider' tells that the secret number is lower after which we turn the head coin to tails. Now we toss with no coins and guess 0 as a second guess. With probability $(1/2)^2$ we'll guess 2. We turn one of the head coins to tails after which we're back in the case where the secret number is equal to 0 and we have a total of 1 coins. If we take the sum of those three cases we'll get:

$$E[X_0^2] = 1/4 + 1/2(1+1) + 1/4(1 + E[X_0^1]) = 15/8$$

If the secret number is equal to 0 we guess 0 with probability 1/4. After that we turn one of the tail coins to heads. We'll toss the last coin until it shows tails. This is the same as tossing with 1 coin where the secret number is 0. With probability 1/2 we guess 1 and with probability 1/4 we'll guess 2. In that case we turn one of the head coins to tails and are in the same situation as $E[X_1^1]$:

$$E[X_1^2] = 1/4(1 + E[X_0^1]) + 1/2 + 1/4(1 + E[X_1^1]) = 7/4$$

5.2.3 With n coins

Now we make the recursion formulas for both $E[X_0^n]$ and $E[X_1^n]$. We have:

$$E[X_0^n] = \frac{1}{2^n} + \frac{n}{2^n}2 + \sum_{k=2}^{n} \frac{\binom{n}{k}}{2^n}(1 + E[X_0^{k-1}]) \tag{19}$$

And for the secret number 1 we get:

$$E[X_1^n] = \frac{1}{2^n}(1 + E[X_0^{n-1}]) + \frac{n}{2^n} + \sum_{k=2}^{n} \frac{\binom{n}{k}}{2^n}(1 + E[X_1^{k-1}]) \tag{20}$$

We see that we need the values of $E[X_0^n]$ for the values of $E[X_1^n]$. Just as before we can make a table with the expected values for different numbers of coins, we include the results of our old method:

	'New' method		**'Old' method**	
n	$E[X_0^n]$	$E[X_1^n]$	$E[X_0^n]$	$E[X_1^n]$
1	1.5000	1.5000	2.0000	2.0000
2	1.8750	1.7500	2.6667	2.0000
3	2.1719	2.0156	3.1429	2.3333
4	2.4170	2.2617	3.5048	2.6667
5	2.6258	2.4768	3.7942	2.9556
6	2.8079	2.6626	4.0348	3.2000
7	2.9694	2.8250	4.2408	3.4085
8	3.1146	2.9695	4.4211	3.5894
9	3.2465	3.1003	4.5813	3.7494
10	3.3672	3.2203	4.7256	3.8932
50	5.4401	5.2934	6.9910	6.1582
100	6.4055	6.2588	7.9838	7.1511

Table 6: Expected number of tosses to find the secret numbers 0 and 1 with our new and old guessing strategy

It is clear that our new method is way more accurate then the old one. Especially for the secret number 0 we see that it needs at least 1 guess less then the old method. And a figure that shows the relative difference between these two expected values for the first 100 values of n:

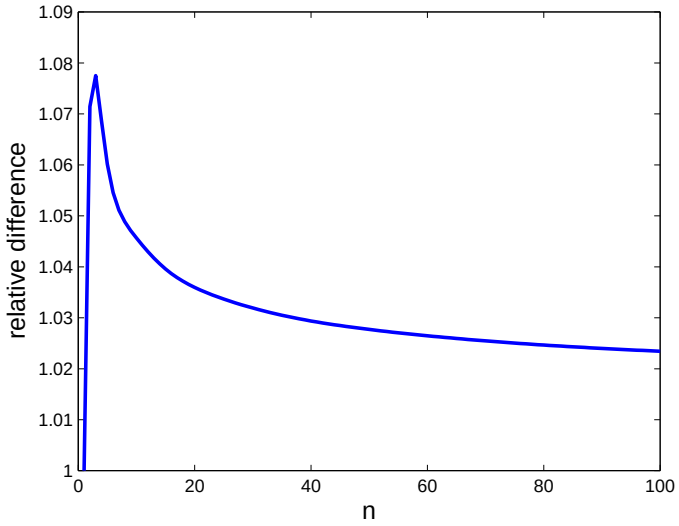


Figure 11: $E[X_0^n]$ divided by $E[X_1^n]$

It seems that the relative difference will converge to one again. This isn't that hard to understand. Suppose that the secret number is picked between 0 and 10^{100} then it is understandable that the secret numbers are equally hard to guess (the real problem is where to get 10^{100} coins).

5.3 A 'perfect' guessing strategy

We'll try to make up a perfect strategy for this game. We define a perfect strategy as a strategy that has the same expected value for every possible secret number:

Suppose we can change the probability for tossing heads after each toss. Is is possible to:

1. Even the expected number of times we need to toss coins for each possible outcome.
2. Find the lowest expected number of times we need to toss coins for each possible outcome.

We'll make use of the following notation:

$p_{ij}^{(k)}$ the probability that, when in state i, to go to state j when tossing with k coins

And,

$E[X_i^n]$ the expected number of tosses when the secret number is i with n coins

Because of the fact that we don't have any information before we've made any guess (we have absolutely no clue whether the secret number will be either high or low) it is clear that we do the first toss with probability $\frac{1}{2}$.

5.3.1 With one coin

First we'll do the computations for $n = 1$, i.e with one coin. We use the same strategy as before, where we turn one coin from heads to tails (of from tails to heads) after each toss.

If we work with 1 coin, the solution is the following: There are two possible choices for the secret number; 0 and 1. We need to match the following expected values: $E[X_0^1]$ and $E[X_1^1]$. We can write out expressions for both: If the output is "LOWER" one coin that was heads is turned to tails. If the output is "HIGHER" one coin that was tails is turned to heads. This way it is impossible to return to the same stage. Using this it is easy to solve the problem for $n = 1$:

$$E[X_0^1] = \frac{1}{2} + \frac{1}{2}(1+1) = \frac{3}{2}$$
$$E[X_1^1] = \frac{1}{2} + \frac{1}{2}(1+1) = \frac{3}{2}$$

With this result the problem for $n = 1$ is solved. Now we solve the problem for 2 coins.

5.3.2 With two coins

There are 3 possible secret numbers: 0,1 and 2. We write out expressions for the expected number of tosses, first for the event that the secret number is equal to 0:

$$E[X_0^2] = \left(\frac{1}{4}\right) + \left(\frac{1}{2}\right) \cdot 2 + \left(\frac{1}{4}p_{20}^{(1)}\right) \cdot 2 + \left(\frac{1}{4}p_{21}^{(1)}\right) \cdot 3$$

When the secret number is equal to 2 we get an almost similar expression:

$$E[X_2^2] = \left(\frac{1}{4}\right) + \left(\frac{1}{2}\right) \cdot 2 + \left(\frac{1}{4}p_{02}^{(1)}\right) \cdot 2 + \left(\frac{1}{4}p_{01}^{(1)}\right) \cdot 3$$

The expression for the event that the secret number is equal to 1 is slightly different:

$$E[X_1^2] = \left(\frac{1}{2}\right) + \left(\frac{1}{4}p_{01}^{(1)} + \frac{1}{4}p_{21}^{(1)}\right) \cdot 2 + \left(\frac{1}{4}p_{02}^{(1)}1 + \frac{1}{4}p_{20}^{(1)}1\right) \cdot 3$$

Now we should notice that $p_{201} + p_{211} = 1$ and the same holds for $p_{011} + p_{021} = 1$. We should also notice that because of symmetry we have $p_{20}^{(1)} = p_{02}^{(1)}$,$p_{21}^{(1)} = p_{01}^{(1)}$ and $E[X_0^2] = E[X_2^2]$. Now we can solve this problem by direct calculation.

$$p_{02}^{(1)} = p_{20}^{(1)} = \frac{2}{3} \qquad p_{01}^{(1)} = p_{21}^{(1)} = \frac{1}{3}$$

When we fill in these probabilities in our formulas for the expected value:

$$E[X_0^2] = E[X_1^2] = E[X_2^2] = \frac{11}{6}$$

5.3.3 With three coins

For $n = 3$ we encounter another problem. We'll have to even 4 expected values:

$$E[X_0^3] = E[X_1^3] = E[X_2^3] = E[X_3^3]$$

We can write down the exact formulas for these expectations. Note that these are just the sum of all the possible events times the probability for that particular event.

$$\begin{aligned}
(\mathbf{1}) \quad E[X_0^3] =& \frac{1}{8} + \frac{3}{8} \cdot 2 + \frac{3}{8} \cdot p_{20}^{(1)} \cdot 2 + \frac{3}{8} \cdot p_{21}^{(1)} \cdot 3 + \frac{1}{8} \cdot p_{30}^{(2)} \cdot 2 + \frac{1}{8} \cdot p_{31}^{(2)} \cdot 3 + \frac{1}{8} \cdot p_{32}^{(2)} \cdot p_{20}^{(1)} \cdot 3 \\
&+ \frac{1}{8} \cdot p_{32}^{(2)} \cdot p_{21}^{(1)} \cdot 4 \\
(\mathbf{2}) \quad E[X_1^3] =& \frac{1}{8} \cdot p_{01}^{(2)} \cdot 2 + \frac{1}{8} \cdot p_{02}^{(2)} \cdot 3 + \frac{1}{8} \cdot p_{03}^{(2)} \cdot p_{31}^{(1)} \cdot 3 + \frac{1}{8} \cdot p_{03}^{(2)} \cdot p_{32}^{(1)} \cdot 4 + \frac{3}{8} + \frac{3}{8} \cdot p_{21}^{(1)} \cdot 2 \\
&+ \frac{3}{8} \cdot p_{20}^{(1)} \cdot 3 + \frac{1}{8} \cdot p_{31}^{(2)} \cdot 2 + \frac{1}{8} \cdot p_{30}^{(2)} \cdot p_{01}^{(1)} \cdot 3 + \frac{1}{8} \cdot p_{30}^{(2)} \cdot p_{02}^{(1)} \cdot 4 + \frac{1}{8} \cdot p_{32}^{(2)} \cdot p_{21}^{(1)} \cdot 3 \\
&+ \frac{1}{8} \cdot p_{32}^{(2)} \cdot p_{20}^{(1)} \cdot 4 \\
(\mathbf{3}) \quad E[X_2^3] =& \frac{1}{8} \cdot p_{02}^{(2)} \cdot 2 + \frac{1}{8} \cdot p_{01}^{(2)} \cdot p_{12}^{(1)} \cdot 3 + \frac{1}{8} \cdot p_{01}^{(2)} \cdot p_{13}^{(1)} \cdot 4 + \frac{1}{8} \cdot p_{03}^{(2)} \cdot p_{32}^{(1)} \cdot 3 + \frac{1}{8} \cdot p_{03}^{(2)} \cdot p_{31}^{(1)} \cdot 4 \\
&+ \frac{3}{8} \cdot p_{12}^{(1)} \cdot 2 + \frac{3}{8} \cdot p_{13}^{(1)} \cdot 3 + \frac{3}{8} + \frac{1}{8} \cdot p_{32}^{(2)} \cdot 2 + \frac{1}{8} \cdot p_{31}^{(2)} \cdot 3 + \frac{1}{8} \cdot p_{30}^{(2)} \cdot p_{02}^{(1)} \cdot 3 \\
&+ \frac{1}{8} \cdot p_{30}^{(2)} \cdot p_{01}^{(1)} \cdot 4 \\
(\mathbf{4}) \quad E[X_3^3] =& \frac{1}{8} \cdot p_{03}^{(2)} \cdot 2 + \frac{1}{8} \cdot p_{01}^{(2)} \cdot p_{13}^{(1)} \cdot 3 + \frac{1}{8} \cdot p_{01}^{(2)} \cdot p_{12}^{(1)} \cdot 4 + \frac{1}{8} \cdot p_{02}^{(2)} \cdot 3 + \frac{3}{8} \cdot p_{13}^{(1)} \cdot 2 \\
&+ \frac{3}{8} \cdot p_{12}^{(1)} \cdot 3 + \frac{3}{8} \cdot 2 + \frac{1}{8}
\end{aligned}$$

We know that some probabilities have to add up to one:

$$\begin{aligned}
(\mathbf{5}) \quad & p_{30}^{(2)} + p_{31}^{(2)} + p_{32}^{(2)} = 1 \\
(\mathbf{6}) \quad & p_{01}^{(2)} + p_{02}^{(2)} + p_{03}^{(2)} = 1 \\
(\mathbf{7}) \quad & p_{01}^{(1)} + p_{02}^{(1)} = 1 \\
(\mathbf{8}) \quad & p_{12}^{(1)} + p_{13}^{(1)} = 1 \\
(\mathbf{9}) \quad & p_{20}^{(1)} + p_{21}^{(1)} = 1 \\
(\mathbf{10}) \quad & p_{31}^{(1)} + p_{32}^{(1)} = 1
\end{aligned}$$

And because of symmetry we now that:

$$
\begin{aligned}
(\mathbf{11}) \quad & p_{01}^{(1)} = p_{32}^{(1)} \\
(\mathbf{12}) \quad & p_{02}^{(1)} = p_{31}^{(1)} \\
(\mathbf{13}) \quad & p_{03}^{(2)} = p_{30}^{(2)} \\
(\mathbf{14}) \quad & p_{02}^{(2)} = p_{31}^{(2)} \\
(\mathbf{15}) \quad & p_{01}^{(2)} = p_{32}^{(2)} \\
(\mathbf{16}) \quad & p_{12}^{(1)} = p_{21}^{(1)} \\
(\mathbf{17}) \quad & p_{13}^{(1)} = p_{20}^{(1)}
\end{aligned}
$$

We should notice that of these 17 equations there are 3 that cancel out. The symmetry tells us that equations 5 and 6, 7 and 10 and the equations 8 and 9 are the same. And because we want that $E[X_0^3] = E[X_1^3], E[X_1^3] = E[X_2^3]$ and $E[X_2^3] = E[X_3^3]$. This means that equations 1, 2, 3 and 4 give us 3 equation that we want to solve. This means that we have a total of 17-3-1=13 equations. The number of variables is 14. This means that we need another equation to solve this system. It may be hard to understand this so we try to look at it more general.

Because of symmetry we now that the equations for $E[X_0^3]$ and $E[X_3^3]$ (and the same holds for $E[X_1^3]$ and $E[X_2^3]$) are equivalent. This means that we actually have only one equation:

$$E[X_0^3] = E[X_1^3]$$

Now we can count the number of times we can adjust the probability for tossing heads. The first toss (with 3 coins) is done with probability $\frac{1}{2}$. When we are tossing with 2 coins we can adjust the probability for tossing heads. The same holds for tossing with 1 coin. This means we have 2 degrees of freedom and only 1 equation. In other words; there is no unique solution.

5.3.4 With n coins

If we want to solve the problem for n coins we have the following: There are $n+1$ possible secret numbers: $0, 1, \ldots, n+1$. Because of symmetry we'll have that $E[X_0^n] = E[X_n^n]$, $E[X_1^n] = E[X_{n-1}^n], E[X_2^n] = E[X_{n-2}^n], \ldots$ etc. This means that we have $\lfloor \frac{n}{2} \rfloor$ equations:

$$
\begin{array}{rcl}
E[X_0^n] & = & E[X_1^n] \\
E[X_1^n] & = & E[X_2^n] \\
\vdots & \vdots & \vdots \\
E[X^n_{\lfloor \frac{n}{2} \rfloor}] & = & E[X^n_{\lceil \frac{n}{2} \rceil}]
\end{array}
$$

The number of variables is equal to $n-1$; only the first toss has a fixed probability. This means that we can solve this problem for $n = 1$ and $n = 2$. In these cases we have respectively 1 equation without variables and 2 equations with 2 variables. But when $n = 3$ or even bigger we need more equations to solve our problem. We'll need to think of another way so solve this problem but unfortunately there wasn't enough time to do this. This means that the following problem:

Suppose we can change the probability for tossing heads after each toss. Is is possible to:

1. Even the expected number of times we need to toss coins for each possible outcome.
2. Find the lowest expected number of times we need to toss coins for each possible outcome.

stays unanswered in this paper due to a lack of time.

References

[1] J.J.A.M Brands, F.W. Steutel, R.J.G. Wilms, *On the number of maxima in a discrete sample*, **Journal of Statistics Probability Letters 20**, 209-217, 1994.

[2] Råde, L., *Problem E3436*, **Amer. Math Monthly**, 366, 1991.

[3] Barndorff-Nielsen, O., *On the rate of growth of the partial maxima of a sequence of independent identically distributed random variables*, **Math. Scand**, 383-394, 1961.

[4] Ross S.M., *Stochastic Processes 2nd Edition*, **John Wiley & Sons**, 4-5, 1996.

[5] Schilling Ren L., *Measures, Integrals and Martingales*, **Cambridge University Press**, 89, 2007.

[6] Shreve Steven E., *Stochastic Calculus for Finance ii*, **Springer**, 70, 2004.

6 Matlab codes

The Matlab codes that were used for this report can be found in this section.

Exact formula for $P(K_n = 1)$

```
%This program is used to find an 'exact' value for the probability that the
%last toss is with one coin for various values of n and p.

clear all
clc

%Constants:
    %number of coins
    n=127;
    %infinity look-a-like
    infty=1000;
    %probability of landing on heads
    p=3/4;


%lambda
lambda=-log(1-p);
%theta
theta_n=((1/lambda)*log(n))-floor(((1/lambda)*log(n)));
X=[-infty:infty];
for l=-infty:infty
    X(l+infty+1)=exp(-lambda*(l-theta_n))*exp(-exp(-lambda*(l-theta_n)));
end
PKn_k=p*sum(X);
```

The value of θ_n against $P(K_n = 1)$

```
% In this progam we variete the value of theta between 0 and 1 to see the
% influence on the probability for ending up with 1 coin in the last toss
% the results are represented in a plot
clear all
clc
%The number of steps between 0 and 1 we take for theta:
m=1000;
%Make a matrix in which we denote the different values of theta
theta=[0:1/m:1-1/m];
%Parameters that won't change
infty=200; p=3/4;  lambda=-log(1-p);
%Empty matrix in which the different oucomes will come
PKn_k=zeros(m,1);
X=[-infty:infty];
for i=1:m
    for l=-infty:infty
```

```
        X(l+infty+1)=exp(-lambda*(l-theta(i)))*exp(-exp(-lambda*(l-theta(i))));
    end
    PKn_k(i)=p*sum(X);
end
Mean=mean(PKn_k)
Max=max(PKn_k)
Min=min(PKn_k)
amplitude1=abs(Mean-Max)
amplitude2=abs(Mean-Min)
factor=amplitude1/amplitude2

%plot the probability against the value of theta
plot(theta,PKn_k)
xlabel('\theta')
ylabel('PKn=1')
%title('\theta versus the probability
       for ending up with 1 coin in the last toss')
%now we give the values of theta where there is a minimum/maximum
i=1;
while(PKn_k(i)<max(PKn_k))
    i=i+1;
end
maximum_valtheta=theta(i)

i=1;
while(PKn_k(i)>min(PKn_k))
    i=i+1;
end
minimum_valtheta=theta(i)
```

Monte Carlo methods

```
%In this program we make 95% confidence intervals using Monte Carlo methods
%This has been done to compute the probability that the final toss is done
%with one coin
tic
%clc
clear all
%totaal aantal munten
N=round(63);
P=20;
p=0.75;
%aantal keer dat er herhaald wordt
herhaal=10000000;
herhaal2=1;
%tellers maken
i_1=1;
i_2=1;
i_3=1;
```

```
%chrismatrix maken
%chrismatrix=zeros(P,1+herhaal2+1);
for(i=1:herhaal2)
    %herhaal2-i
    %for(i_2=1:P)
    %    chrismatrix(i_2,1)=i_2;
    %end
    aantalworpen=zeros(herhaal,1);
    muntenlaatsteworp=zeros(herhaal,1);
    for(i_1=1:herhaal)
        %herhaal-i_1
        worp=geornd(p,N,1)+1;
        sortworp=sort(worp);
        sort1worp=(sortworp==sortworp(N,1));
        aantalworpen(i_1,1) = sortworp(N,1);
        muntenlaatsteworp(i_1,1) = sum(sort1worp);
    end
    %aantalworpen;
    %muntenlaatsteworp;
    %gemaworpen=mean(aantalworpen)
    gmlworp=mean(muntenlaatsteworp);
    stdmlworp=std(muntenlaatsteworp);
    kansop1laatsteworp=sum(muntenlaatsteworp==1)/herhaal
    %for(i_3=1:herhaal)
    %    chrismatrix(muntenlaatsteworp(i_3),i+1)=
       %chrismatrix(muntenlaatsteworp(i_3),i+1)+1;
    %end
end
%for(j=1:P)
%chrismatrix(j,1+herhaal2+1)=round(mean(chrismatrix(j,2:herhaal2+1)));
%end
%chrismatrix
%PKn=chrismatrix(:,herhaal2+2)/herhaal
btbhint95=[kansop1laatsteworp-1.96*stdmlworp/sqrt(herhaal),
kansop1laatsteworp+1.96*stdmlworp/sqrt(herhaal)]
kansop1laatsteworp
toc
```

Expected number of tosses coin strategy 1

```
%This program makes a table with the expected number of tosses to find the
%secret number 0 and 1 for the strategy where you can guess the same number
%twice

clc
clear all
%hoeveel waardes bepalen?
M=100;
```

```
%verwachtingen matrix
EX0=zeros(M,1);
%de eerste waarde vullen we alvast in
EX0(1)=2;
for i=2:M
    clear somrij
    %somrij maken + een extra vakje voor het extra deel
    somrij=zeros(1,i+1);
    %de eerste en de laatste waarde liggen vast:
    somrij(1)=(1/2)^i;
    somrij(length(somrij))=(1/2)^i;
    for i_2=2:(i)
        somrij(i_2)=((factorial(i)/(factorial(i_2-1)
        *factorial(i-(i_2-1))))/(2^i))*(1+EX0(i_2-1));
    end
    somrij;
    %we vermenigvuldigen met een bepaalde constante om de recursie eruit te
    %krijgen
    EX0(i)=((sum(somrij))*2^i)/(2^i-1);
end
EX0;

%nu doen we hetzelfde voor EX1:
EX1=zeros(M,1);
%de eerste waarde vullen we alvast in
EX1(1)=2;
EX1(2)=2;
for i=2:M
    clear somrij
    %somrij maken + een extra vakje voor het extra deel
    somrij=zeros(1,i+1);
    %de eerste en de laatste waarde liggen vast, nu ligt ook de 2e waarde vast:
    somrij(1)=(1/2)^i;
    somrij(length(somrij))=(1/2)^i;
    somrij(2)=((factorial(i)/(factorial(1)*factorial(i-1)))/(2^i));
    for i_2=3:(i)
        somrij(i_2)=((factorial(i)/(factorial(i_2-1)
        *factorial(i-(i_2-1))))/(2^i))*(1+EX1(i_2-1));
    end
    somrij;
    %we vermenigvuldigen met een bepaalde constante om de recursie eruit te
    %krijgen
    EX1(i)=((sum(somrij))*2^i)/(2^i-2);
end
EX1;

EX0en1=transpose([transpose(EX0);transpose(EX1);transpose(EX0./EX1)])

plot(EX0en1(:,3),'LineWidth',2)
```

```
xlabel('n','FontSize',14)
ylabel('relative difference','FontSize',14)
```

Expected number of tosses coin strategy 2

```
%This program makes a table with the expected number of tosses to find the
%secret number 0 and 1 for the strategy where you can't guess the same number
%twice

clc
clear all
%hoeveel waardes bepalen?
M=100;
%verwachtingen matrix
EX0=zeros(M,1);
%de eerste waarde vullen we alvast in
EX0(1)=3/2;
for i=2:M
    clear somrij
    %somrij maken + een extra vakje voor het extra deel
    somrij=zeros(1,i+1);
    %de eerste en de laatste waarde liggen vast:
    somrij(1)=(1/2)^i;
    somrij(2)=(factorial(i)/(factorial(i-1)))/(2^i)*(1+1);
    for i_2=3:(i+1)
     somrij(i_2)=((factorial(i)/(factorial(i_2-1)*
     factorial(i-(i_2-1))))/(2^i))*(1+EX0(i_2-2));
    end
    somrij;
    %we vermenigvuldigen met een bepaalde constante om de recursie eruit te
    %krijgen
    EX0(i)=(sum(somrij));
end
%EX0;


%nu doen we hetzelfde voor EX1:
EX1=zeros(M,1);
%de eerste waarde vullen we alvast in
EX1(1)=3/2;
for i=2:M
    clear somrij
    %somrij maken + een extra vakje voor het extra deel
    somrij=zeros(1,i+1);
    %de eerste en de laatste waarde liggen vast, nu ligt ook de 2e waarde vast:
    somrij(1)=(1/2)^i*(1+EX0(i-1));
    somrij(2)=(factorial(i)/factorial(i-1))/(2^i);
    for i_2=3:(i+1)
        somrij(i_2)=((factorial(i)/(factorial(i_2-1)*factorial(i-(i_2-1))))
```

```
        /(2^i))*(1+EX1(i_2-2));
    end
    somrij;
    %we vermenigvuldigen met een bepaalde constante om de recursie eruit te
    %krijgen
    EX1(i)=sum(somrij);
end
%EX1

EX0en1=transpose([transpose(EX0);transpose(EX1);transpose(EX0./EX1)])
plot(EX0en1(:,3),'LineWidth',2)
xlabel('n','FontSize',14)
ylabel('relative difference','FontSize',14)
```

Empirical version of the expected number of tosses 1

```
%A program that 'plays' the coin tossing strategy where there is a
%probability to guess the same number twice.

%een programma om te bepalen in hoeverre het geheime getal 0 beter is dan
%het geheime getal 1 voor verschillende aantallen munten (en dus
%verschillende aantallen intervallen waaruit gekozen mocht worden).

%doet er bij 100 munten en 100000 repititions ongeveer 5 minuten over.

clear all
clc
tic
%geheime getal matrix:
GG=[0,1];
%aantal keer herhalen om zo een betrouwbaarheidsinterval te kunnen maken
M=10;
for i_3=1:M
    for i_2=1:length(GG)
        clear i
        clear N
        clear gok

        %de teller definieren
        i=1;

        %Matrix waarin het aantal munten waarmee wordt
        %gegooid wordt bijgehouden:
        N(i)=100;

        %Matrix waarin het getal dat gegokt wordt wordt
        %bijgehouden. Dit betekent dus dat het laatste
        %element van gok gelijk moet zijn aan het geheime
        %getal. Deze matrix is dus even groot als matrix N.
```

```
        gok(i)=N(i);

        %Nu simuleren we het gooien. We gooien zolang de gok
        %niet gelijk is aan het geheime getal.
        while(gok(i)~=GG(i_2))
            %Hier doen we berekeningen voor het geval dat de gok
            %groter is dan het geheime getal
            if(gok(i)>GG(i_2))
                %we gooien met het aantal munten waarbij we
                %onze gok gelijk nemen aan het aantal munten dat 'kop' is.
                gok(i+1)=sum(round(rand(N(i),1)));
                %we tellen een worp bij het totaal aantal worpen op.
                i=i+1;
                %het nieuwe aantal munten is gelijk aan onze gok
                N(i)=gok(i);
                %wanneer we nul gokken krijgen we te horen
                % dat we klaarzijn of dathet geheime getal
                %hoger is. Daarom nemen we dan het aantal munten
                %gelijk aan het aantal munten in de vorige worp.
                if(N(i)==0)
                    N(i)=N(i-1);
                end
            %Hier doen we berekeningen voor het geval dat de gok
            %kleiner is dan het geheime getal
            else
                gok(i+1)=sum(round(rand(N(i),1)));
                i=i+1;
                N(i)=gok(i);
                if(N(i)==0)
                    N(i)=N(i-1);
                end
            end
        end
        aantal_pogingen(i_3,i_2)=i-1;
    end
end
gemiddelden=[mean(aantal_pogingen(:,1)),mean(aantal_pogingen(:,2))]
betrouwbaarheidsintervallen=
[gemiddelden(1)-1.96*std(aantal_pogingen(:,1))/sqrt(M),
gemiddelden(1)+1.96*std(aantal_pogingen(:,1))/sqrt(M) ;
      gemiddelden(2)-1.96*std(aantal_pogingen(:,2))/sqrt(M),
      gemiddelden(2)+1.96*std(aantal_pogingen(:,2))/sqrt(M)]
toc
```

Empirical version of the expected number of tosses 2

```
%The same program but now for the other case

%een programma om te bepalen in hoeverre het geheime getal 0 beter is dan
```

```
%het geheime getal 1 voor verschillende aantallen munten (en dus
%verschillende aantallen intervallen waaruit gekozen mocht worden).
clear all
clc
tic
%geheime getal matrix:
GG=[0,1];
%aantal keer herhalen om zo een betrouwbaarheidsinterval te kunnen maken
M=100000;
for i_3=1:M
    for i_2=1:length(GG)
        clear i
        clear N
        clear gok

        %de teller definieren
        i=1;

        %Matrix waarin het aantal munten waarmee wordt
        %gegooid wordt bijgehouden:
        N(i)=100;

        %Matrix waarin het getal dat gegokt wordt wordt bijgehouden.
        %Dit betekent dus dat het laatste element van gok gelijk
        %moet zijn aan het geheime getal. Deze matrix is dus
        %even groot als matrix N.
        gok(i)=N(i);

        %Nu simuleren we het gooien. We gooien zolang
        %de gok niet gelijk is aan het geheime getal.
        while(gok(i)~=GG(i_2))
            %Hier doen we berekeningen voor het geval
            %dat de gok groter is dan het geheime getal
            if(gok(i)>GG(i_2))
                %we gooien met het aantal munten waarbij we onze gok
                %gelijk nemen aan het aantal munten dat 'kop' is.
                gok(i+1)=sum(round(rand(N(i),1)));
                %we tellen een worp bij het totaal aantal worpen op.
                i=i+1;
                %het nieuwe aantal munten is gelijk aan onze gok
                N(i)=gok(i);
                %wanneer we nul gokken krijgen we te horen
                %dat we klaarzijn of dat
                %het geheime getal hoger is.
                %aarom nemen we dan het aantal munten
                %gelijk aan het aantal munten in de vorige worp.
                if(N(i)==0)
                    N(i)=N(i-1);
                end
```

```
            %Hier doen we berekeningen voor het geval dat
            %de gok kleiner is dan het geheime getal
            else
                gok(i+1)=sum(round(rand(N(i),1)));
                i=i+1;
                N(i)=gok(i);
                if(N(i)==0)
                    N(i)=N(i-1);
                end
            end
        end
        aantal_pogingen(i_3,i_2)=i-1;
    end
end
gemiddelden=[mean(aantal_pogingen(1,:)),mean(aantal_pogingen(2,:))]
betrouwbaarheidsintervallen=
                [gemiddelden(1)-1.96*std(aantal_pogingen(1,:))/sqrt(M),
                gemiddelden(1)+1.96*std(aantal_pogingen(1,:))/sqrt(M) ;
                gemiddelden(2)-1.96*std(aantal_pogingen(2,:))/sqrt(M),
                gemiddelden(2)+1.96*std(aantal_pogingen(2,:))/sqrt(M)]
toc
```